A CONDITIONAL COVERAGE PATH PLANNING METHOD FOR AN AUTONOMOUS LAWN MOWER

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ARDIÇ KAROL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
MECHANICAL ENGINEERING

MAY 2016

Approval of the thesis:

A CONDITIONAL COVERAGE PATH PLANNING METHOD FOR AN AUTONOMOUS LAWN MOWER

submitted by **ARDIÇ KAROL** in partial fulfillment of the requirements for the degree of **Master of Science in Mechanical Engineering Department, Middle East Technical University** by,

Prof. Dr. Gülbin DURAL ÜNVER
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Tuna BALKAN
Head of Department, **Mechanical Engineering** _____

Assoc. Prof. Dr. E. İlhan KONUKSEVEN
Supervisor, **Mechanical Engineering Dept., METU** _____

Assist. Prof. Dr. A. Buğra KOKU
Co-supervisor, **Mechanical Engineering Dept., METU** _____

Examining Committee Members:

Assoc. Prof. Dr. Yiğit YAZICIOĞLU
Mechanical Engineering Department, METU _____

Assoc. Prof. Dr. E. İlhan KONUKSEVEN
Mechanical Engineering Department, METU _____

Assist. Prof. Dr. A. Emre TURGUT
Mechanical Engineering Department, METU _____

Assist. Prof. Dr. Kıvanç AZGIN
Mechanical Engineering Department, METU _____

Assist. Prof. Dr. K. Bilge ARIKAN
Mechatronics Engineering Department, Atılım University _____

Date: May 3rd, 2016

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : Ardıç KAROL

Signature :

ABSTRACT

A CONDITIONAL COVERAGE PATH PLANNING METHOD FOR AN AUTONOMOUS LAWN MOWER

KAROL, Ardıç

M.S., Department of Mechanical Engineering

Supervisor : Assoc. Prof. Dr. E. İlhan KONUKSEVEN

Co-Supervisor : Assist. Prof. Dr. A. Buğra KOKU

May 2016, 183 pages

Randomized and deterministic coverage path planning methods are widely used in autonomous lawn mowers. Random planning cannot guarantee a complete coverage, whereas, many deterministic techniques are not solely eligible for unstructured outdoor environments, since they highly suffer from wheel slippage or numerical drift. Besides, complete coverage techniques either demands high computational power or expensive sensor hardware.

A genuine, **C**onditional **C**overage **P**ath **P**lanning (CCPP) method, which satisfies complete coverage with a comparably low computational requirement, is developed in this study. CCPP is created from a motivation that the border information must be taken into account for a better coverage performance, since the working environments for autonomous lawn mowers are all bordered.

For the implementation of this developed coverage technique, a state-of-art autonomous lawn mower is designed and produced. Besides being an implementation platform, the robot is designed to satisfy market demands, where it became a ready-

to-sell commercial product at the end of this work. Moreover, a unique simulation environment is developed for CCPP method performance evaluation.

In order to validate the CCPP technique, many simulations and outdoor tests are performed to visualize its advantages and disadvantages over the widely used coverage methods. Test results revealed that the CCPP method significantly increased coverage performance, when compared with conventional coverage algorithms.

Although this method is implemented to an autonomous lawn mower in this study, it is concluded that CCPP can be a beneficial coverage path planning alternative for many commercial domestic robots, since it provides a decent coverage without necessity for expensive hardware or intensive computations.

Keywords: Conditional Coverage Path Planning, Complete Coverage Method, Autonomous Lawn Mower, Autonomous Outdoor Navigation

ÖZ

BİR OTONOM ÇİM BİÇME ROBOTU İÇİN
KOŞULLANDIRILMIŞ ALAN KAPSAMA YÖRÜNGE PLANLAMA METODU

KAROL, Ardıç

Yüksek Lisans, Makina Mühendisliği Bölümü

Tez Yöneticisi : Doç. Dr. E. İlhan KONUKSEVEN

Ortak Tez Yöneticisi : Yrd. Doç. Dr. A. Buğra KOKU

Mayıs 2016, 183 sayfa

Rastlantısal ve deterministik kapsama güzergâhı planlaması, çim biçme robotlarında yaygın olarak uygulanmaktadır. Rastlantısal yol planlaması yönteminin tam kapsama garanti edemeyeceği bilindiği gibi, birçok deterministik teknik tekerlek patinajı ve numerik kayma sebepleri ile dış ortam koşullarında başarılı olamamaktadır. Buna ek olarak tam kapsama teknikleri yüksek işlem gücü veya pahalı algılama donanımları gerektirmektedir.

Bu çalışma kapsamında özgün olarak geliştirilen **K**oşullandırılmış **A**lan **K**apsama **Y**örünge **P**lanlama (KAKYP) metodu, görece düşük işlem yükü gereksinimine sahip bir tam kapsama tekniğidir. Çim biçme robotları çalışma alanlarının sınırlandırıldığı gerçeğinden yola çıkılarak KAKYP, daha iyi bir kapsama için sınır bilgisinin seyrüsefer hesaplarına dahil edilmesi motivasyonu ile oluşturulmuştur.

Geliştirilen bu kapsama tekniğinin uygulanması adına yeni ve eşsiz bir otonom çim biçme robotu tasarlanmış ve üretilmiştir. Bir uygulama platformu olma özelliğinin yanı sıra bu robot, pazar talepleri gözetilerek tasarlanmış ve çalışmanın sonunda satışa

hazır bir ticari ürün haline gelmiştir. Bunun yanı sıra, KKGP performansının belirlenmesi adına yeni bir simülatör geliştirilmiştir.

KAKYP tekniğinin doğrulanması, bu tekniğin yaygın kullanılan güzergâh planlama algoritmalarına karşı avantaj ve dezavantajlarının belirlenmesi adına birçok simülasyon ve dış ortam testi gerçekleştirilmiştir. Test sonuçları, konvansiyonel kapsama algoritmaları ile karşılaştırıldığında KAKYP'nin kapsama yüzdesini belirgin bir oranda arttırdığını göstermiştir.

Bu çalışma içerisinde KAKYP'nin bir otonom çim biçme robotuna uygulanmasına rağmen pahalı donanımlara veya yoğun işlemlere ihtiyaç duymadan iyi bir kapsama sunması sebebi ile birçok ticari evcil robot için faydalı bir kapsama güzergâh planlaması alternatifi olabileceği sonucuna varılmıştır.

Anahtar Kelimeler: Koşullandırılmış Alan Kapsama Yörünge Planlaması, Tam Kapsama Yöntemi, Otonom Çim Biçme Robotu, Otonom Dış Ortam Seyrüseferi

To my lovely family

ACKNOWLEDGEMENTS

I want to express my sincere appreciation to my supervisor Assoc. Prof. Dr. E. İlhan KONUKSEVEN and my co-supervisor Assist. Prof. Dr. A. Buğra KOKU for their guidance, advice, criticism, encouragements and understanding throughout this work.

The greatest thanks go to my partner, colleague and dear friend Serkan ÇİÇEK for his relentless efforts and for all that he had built up from nothing.

Dedicated hard work of Çağın ÖZDEMİR, Anıl SERDAR, Kutay KÖK and all of my colleagues are gratefully acknowledged.

I would also like to appreciate Serdar GÜNEY for his valuable hands-on technical supports.

Special thanks to all of our contractors, whom manufactured our robot.

Lastly, I would like to express my deepest gratitude to my lovely wife Ayçin, for her understanding, encouragement and for withstanding by me against all of the things that we have been through.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF SYMBOLS

A	Area
D	Distance
d	Wheel diameter
e_p	Heading error of robot
e_p	Position error of robot
F_{net}	Net force acting on the robot
$F_{rolling}$	Rolling force acting on the robot
H_{area}	Rectilinear area height
i	Total reduction ratio
L	Wheelbase of the robot
L_t	Length of the trajectory
m	Mass
n_p	Number of pass in the coverage pattern
P	Power required for drive motors
R	Drive wheel radius of robot
R_r	Rolling resistance
t	Time
T	Torque required for drive motors
ϑ_L	Left wheel velocity
ϑ_R	Right wheel velocity
W	Weight of the robot
W_{area}	Rectilinear area width
W_b	Mowing blade width of robot
x	X coordinate of robot
$\dot{x}$	Robot velocity in X direction
y	Y coordinate of robot

$\dot{y}$	Robot velocity in Y direction
θ	Heading angle of robot
$\dot{\theta}$	Angular velocity of robot

LIST OF ABBREVIATIONS

ADC	Analog to Digital Converter
ALM	Autonomous Lawn Mower
CCPP	Conditional Coverage Path Planning
CCW	Counter Clock-Wise
CNC	Computer Numerical Control
CPP	Coverage Path Planning
CPR	Counts per Revolution
CPU	Central Processor Unit
CW	Clock-Wise
dGPS	Differential Global Positioning System
DR	Dead-Reckoning
GC	Geometric Center
GLONASS	Global Navigation Satellite System
GPS	Global Positioning System
GUI	Graphical User Interface
HMI	Human-Machine Interface
I2C	Inter-Integrated Circuit
ICE	Internal Combustion Engine
IDE	Integrated Development Environment
IFR	International Federation of Robotics
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
IR	Infrared
KF	Kalman Filter
LCD	Liquid Crystal Display
LI-ION	Lithium-Ion
LI-PO	Lithium-Polymer

MEMS	Micro-Electromechanical Systems
MOSFET	Metal Oxide Semiconductor Field Effect Transistor
NI-CD	Nickel-Cadmium
NI-MH	Nickel-Metal Hydride
OPAMP	Operational Amplifier
PCB	Printed Circuit Board
PLA	Polylactic Acid
RF	Radio Frequency
RTK	Real-Time Kinematics
SCL	Serial Clock Line
SDA	Serial Data Line
SLAM	Simultaneous Localization and Mapping
SPI	Serial Peripheral Interface
STC	Spanning Tree Coverage
TOF	Time of Flight
TTL	Transistor-Transistor Logic
UART	Universal Asynchronous Receiver and Transmitter
UAV	Unmanned Aerial Vehicle
US	Ultrasonic
VB	Visual Basic
WMR	Wheeled Mobile Robot

CHAPTER 1

INTRODUCTION

1.1 On Mobile Robotics

The term “robota” is first appeared in 1920 with a science fiction play named as R.U.R. (Rossumovi Univerzální Roboti), written by a Czech writer Karel Čapek [1]. The word “roboti” etymologically means “servants” in Czech language. When, R.U.R. is translated into English in 1921, the corresponding word “robot” is introduced to English language and many others. In 1942, when Isaac Asimov introduced formulated “Three Laws of Robotics”, he inherently named this discipline as robotics.

Even though manipulator robots have started to been used in production and assembly lines in 1970’s, the history of mobile robotics is a little bit older. W. Grey Walter is known as the builder of the first mobile robot, which he named them as Elmer and Elsie. Both robots are three-wheeled, equipped with a light sensor. When they have sensed a light source, they move towards it by avoiding obstacles. These robots are then called as Machina Speculatrix because of their complex environment exploration abilities.

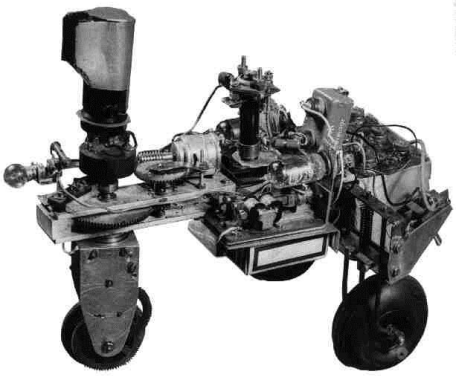


Figure 1.1: Elsie Robot [2]

Shakey was the first robot which utilizes vision sensors. It was developed from 1966 through 1972, by Stanford University. A camera, a range finder sensor, and two bumping detectors were used in this robot. Two electric motors are used for traction, where it has an on-board computer for computations [3].

Hilare Mobile Robots created a breakthrough advancement in mobile robotics. The Hilare family was developed in LAAS (Laboratoire d'Automatique et d'Analyse des Systèmes) from 1977 to 1992. First Hilare robot, Hilare I was developed in 1977. Two traction wheels and a caster wheel are used in driveline. Multibus is used as the bus in that robot with four Intel 802286 processors. In communication, a 9600 baud serial radio modem is used. The robot had odometer, 16 US sensors, and a laser range finder as the sensor equipment.

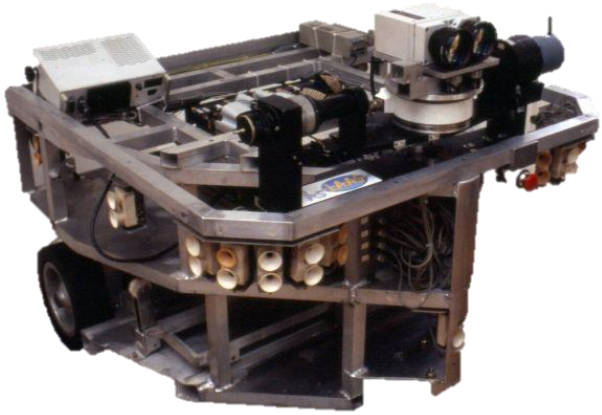


Figure 1.2: Hilare I Robot [4]

After Hilare I, LAAS was developed Hilare II robot with better actuation, processor, and operating system. Dual drive wheels with four idle wheels are introduced in the new robot. Four processors used in this robot are Motorola 68040 and Motorola PPC750. This robot has VxWorks 5.3.1 operating system. Odometric sensors are used in addition to 32 sonar range sensors, 2D laser range finder, a pan/tilt/zoom color camera, and two pan/tilt/zoom black and white cameras. Further developments are also made over Hilare II till 1992, with manipulator addition and weight reduction. Today, Hilare robots are accepted as the pioneers of mobile robotics with their perception and navigation capabilities.

Figure 1.3: Hilare II Robot [4]

With the advancements in technology, companies evolved those academic experience into industrial, military and domestic mobile robots in 1990's. Today, various types of domestic robots are available in the market. The sample taxonomy for domestic robots are presented in Table 1.1 [5].

Table 1.1: Taxonomy of Domestic Robots

Categories	**Applications**	**Examples**
Entertainment Robots	-	Mindstorms, Nao
Security Robots	-	Rovio, Spykee
Collaborative Robots	Vacuum Cleaning	Roomba, Navibot
	Lawn Mowing	Robomower, Automower
	Pool Cleaning	Verro, Aquabot
	Window Cleaning	Windoro
Personal Robots	Telepresence	MantaroBot
	Physical Assistance	Care-o-Bot

Entertainment robots which are actually toys, which are specially designed for interaction to human and objects. As the main aspect of these type of robots, children can enjoy and learn with these robots at the same time.

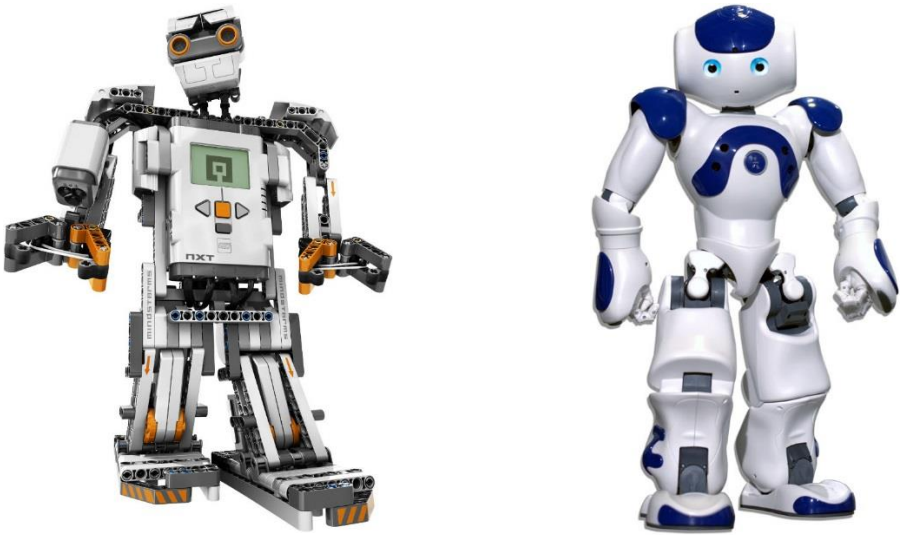


Figure 1.4: Entertainment Robots - Mindstorms [6] and Nao [7]

Main aspect of security robots is home surveillance. They continuously gather information from inside or outside of the house, check for any invasion or unauthorized entry situation.

Figure 1.5: Security Robots - Rovio [8]

Personal robots mainly provides aiding or caring services. They can assist especially to elder or disabled people for walking, feeding, load lifting and shop listing. They

also provide companionship by talking or by music. Some of them are used for therapeutic purposes.

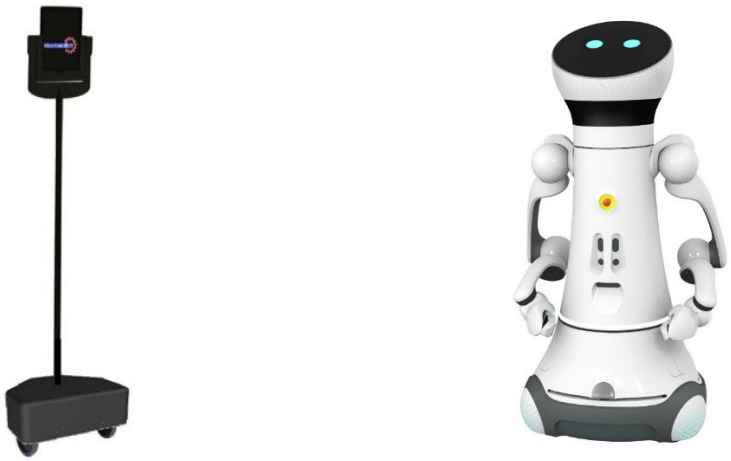


Figure 1.6: Personal Robots - MantaroBot [9] and Care-o-Bot [10]

It is a common dream of everyone for robots to do undesired house chores. For some futurists, this dream is appearing to become true today [11]. According to **I**nternational **F**ederation of **R**obotics (IFR) researches, 4.7 million civilian service robots were sold in 2014, where 3.3 million of it are domestic robots, which corresponds an annual increase of 24% [12]. IFR also estimates that 25.2 million domestic robots will be sold throughout 2015-2018.

Figure 1.7: Collaborative Robots - Navibot [13] and Robomower [14]

With the continuous advancements in research and technology, domestic robots can evaluate, manage and execute their tasks successfully. As it is seen that domestic robotics field is developing and emerging rapidly, the idea of having a robot in every home will not solely belong to futurists anymore.

1.2 On Coverage Path Planning

Motion planning is a vital task for an autonomous mobile robot's navigation performance. Motion planning algorithms originally considered the start-to-goal problem, whose solution requires a path between two points [15]. This path can be generated by sensor-based and/or map-based approaches. As being an old topic, there is a vast variety of motion planning algorithms for mobile robots available in literature.

Although they can be useful for finding a convenient way between two points, conventional start-to-goal, map-based and sensor-based path planning approaches are inadequate for applications that require a regional planning like floor/window cleaning, lawn mowing, demining, harvesting, etc. For those applications, a more sophisticated method - **C**overage **P**ath **P**lanning (CPP) must be considered.

CPP problem has numerous everyday variants like covering salesman problem, lawn mower problem, piano mover's problem, art gallery problem and watchman route problem. Each of them requires a complete coverage of the environment by means of travelling.

CPP is a relatively new discipline in the motion path planning literature. A lot of techniques have been proposed and applied for different tasks so far, yet, more will have to be introduced in close future in order to create robust, reliable and accurate mobile robots, performing complex coverage tasks.

1.3 Scope of the Thesis

It is tested, experienced and evaluated that some heuristic or randomized coverage path planning algorithms have serious inadequacies for a complete coverage, whereas some higher level techniques with expensive hardware demands, also suffers from outdoor conditions even if they theoretically provide a complete coverage for structured environments.

Starting from this point, this thesis aimed to develop a new coverage path planning technique to achieve a better coverage performance over conventional techniques that are used in commercial domestic mobile robots. This technique requires no expensive hardware (i.e. sensors or processors), which makes it easy to utilize for many forms of domestic mobile robots.

The second objective of this work is to exhibit a state-of-art autonomous lawn mower, created regarding user and market requirements. It is intended to make it a commercial product at the end of this work, therefore all details besides navigation performance have also been meticulously handled.

Within the content of the thesis, this autonomous lawn mower is designed, developed, manufactured, integrated and tested, starting from scratch. All the work except some manufacturing is built in-house.

Different coverage and coverage path planning techniques, including our proposal, have been extensively simulated and tested both in unstructured environments to reveal the true performance of proposed coverage path planning technique.

Finally, it is aimed to present an explanatory, well-written, understandable thesis to be beneficial for ones dealing with autonomous mobile robotics.

1.4 Outline of the Thesis

First chapter of the thesis briefly introduces mobile robotics discipline, history and pioneers in this field. After that, it gives core information for distinguishing coverage path planning from a general path planning.

In the second chapter, this mobile robotics and coverage concepts are elaborated. Literature survey of localization and coverage methods, which are the key factors of autonomous navigation performance are presented. A small section which remarks the academic and commercial value for autonomous lawn mowers is also given.

Chapter 3 approaches this subject from a system engineering point of view. It makes the problem definition, which prominences the key ideas about proposed coverage technique and presents system requirements. It also denotes selected concepts which shapes the autonomous lawn mower design.

Chapter 4 reveals majority of work in which mathematical modelling, hardware and software designs are presented in detail.

Simulation environment, scenarios and results are given detailed in Chapter 5.

Chapter 6 is dedicated for manufacturing details of the autonomous lawn mower. Moreover, brief information for the system integration also took its place.

Chapter 7 shows test environment, scenarios and test results. Comparisons for different coverage techniques for outdoor environment have been made. In addition, evaluations and conclusions of test results are also exhibited.

The final chapter summarizes and concludes the work done in thesis. Future works are also discussed.

CHAPTER 2

LITERATURE SURVEY

2.1 Mobile Robot Positioning Methods

Although so many definitions are available in literature, a robot can be briefly described as a machine, designed to execute single or multiple tasks repeatedly, precisely and rapidly. As there are many different types of tasks desired from robots to perform, many different types of robots are also available.

An important subgroup - mobile robot or mobile platform can be defined as a machine who is able to locomote within an environment. This environment can be terrestrial, underwater, celestial or spatial.

In general understanding, there are three important functional characteristics of a mobile robot. A mobile robot has to have;

- Mobility so it can freely move within the working environment
- A Certain Level of Autonomy to limit human interaction
- Perception Ability in order to sense and react within the working environment

A **W**heeled **M**obile **R**obot (WMR) is defined as an autonomous wheeled vehicle that can operate with no human assistance. WMR's are usually equipped with a set of motorized actuators and an array of sensors for the sake of mobility and perception requirements from mobile robots.

Navigation is known as a field of study that focuses on the process of monitoring and controlling the movement of a vehicle from one place to another. In order to move freely in a working environment, navigation is a crucial task for WMR mobility.

The general problem of WMR navigation can be summarized with three questions: “Where am I?”, “Where am I going?” and “How should I get there?” This section is aimed to answer the first question - WMR positioning.

As stated in [16], to date, there exists no truly exact solution for the positioning problem for WMR’s. Approximate solutions can be categorized as *Relative* and *Absolute* positioning. Sticking to the Borenstein’s classification, subgroups of position measurements are defined as;

Relative Positioning

- Odometry
- Inertial Navigation

Absolute Positioning

- Active Beaconing
- Landmark Recognition
- Map/Model Matching

A WMR navigation can rely on one or several methods from above categories. The rest of this section surveys state-of-art techniques for mobile robot positioning in detail.

2.1.1 Dead-Reckoning

In navigation, **D**ead-**R**eckoning (DR) is known as a technique to estimate one’s current position from a previously calculated (or determined) position, by means of velocity and time measurements. For that reason, DR is classified as a relative positioning method. It blindly calculates present from known past. If previous knowledge is correct, this the most powerful and computationally cheapest navigation technique. But if this knowledge contains uncertainties, DR can be seriously erroneous.

DR has a long history in civilization and has been widely used for terrestrial, underwater and celestial navigation purposes. In robotics, DR can be evaluated into two groups.

2.1.1.1 Odometric Navigation

Etymologically, the word “Odometry” is composed of Greek words “Hodos” (way) and “Metron” (measure). In robotics, odometry uses sensor data to estimate position change over time. Those sensors for WMR’s are Wheel Encoders.

Encoders are mechanical to electrical transducers whose output is derived by reading a coded pattern on a rotating disk or a moving scale. They can be classified as rotary or linear depending on the type of the motion. Wheel encoders are rotary encoders which are directly or indirectly attached to a traction or idle wheel of the system. For this reason, only rotary encoders, which are frequently used in WMR’s, are covered in this section.

Encoders are classified by the

- Method used to read the coded element: contact or non-contact
- Type of output: absolute digital word or series of incremental pulses
- Physical phenomenon employed to produce the output: electrical conduction, magnetic, optical, capacitive

Within a vast set of choice, optical encoders are commonly used for WMR acceleration, velocity and position calculations.

Figure 2.1: A Miniature Optical Encoder and Illustration

A simple illustration of an optical encoder is given in the Figure 2.1. It is basically a light chopper which counts the discontinuities on the sensor by emitting and receiving the light beams. The changes on the receiver side are counted in operation and this count gives the rotation in a discrete time interval. Resolution of these sensors is measured in terms of **C**ounts **P**er **R**evolution (CPR). Multiplication of the counts with the value of CPR gives the angular displacement within a specific time interval.

Quadrature encoders are mostly used in robotics. This type of encoders provide both the direction and the number of counts since they produce two square-waves with an appropriate phase difference. This procedure is schematically expressed in Figure 2.2.

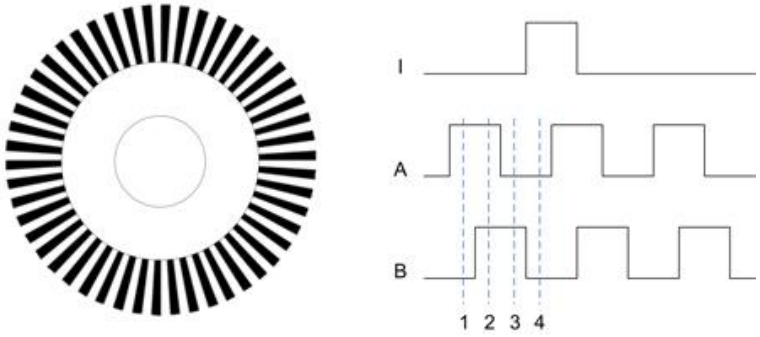


Figure 2.2: Quadrature Encoder Working Principles

Knowing the channel characteristics, the direction of revolution can be known by counting both channels. Counting ticks, the angular speed of the joint attached to the encoder can be calculated. The combinations of this logic is given in Table 2.1.

Table 2.1: Encoder State Table

State	Channel A	Channel B
S1	High	Low
S2	High	High
S3	Low	High
S4	Low	Low

Wheel encoders are inexpensive, simple and easy-to-implement sensors which makes them useful for WMR's. However, odometry is based on the assumption that wheel

revolutions can be translated into linear displacement relative to the floor. This is logical but a naive assumption with limited validity.

One of the practical examples that ruins this assumption is wheel slippage. If one wheel was to slip, the associated encoder would register wheel revolutions even though these revolutions would not correspond to a linear displacement of the wheel.

Besides slippage, there are other inaccuracies in the translation of wheel encoder readings into linear motion. All of those error sources can be classified into two groups: Systematic and Non-Systematic errors.

Borenstein and Feng classified odometry errors and their reasoning as [17];

Systematic Errors

- Unequal wheel diameters
- Average of both wheel diameters differs from nominal diameter
- Misalignment of wheels
- Uncertainty about the effective wheelbase (due to non-point wheel contact with the floor)
- Limited encoder resolution
- Limited encoder sampling rate

Non-Systematic Errors

- Traveling over uneven floors
- Traveling over unexpected objects on the floor
- Wheel-slippage (due to slippery floors, over-acceleration, external forces, etc.)

Systematic odometry errors, which are usually caused by imperfections in the design and mechanical implementation, are vehicle-specific and don't usually change during an operation. However non-systematic odometry errors are caused by interaction of the robot with unpredictable features of the environment (such as irregularities, bumps and slippery surfaces).

Borenstein and Feng introduced a measurement and correction procedure (UMBmark experiment) for systematic odometry errors, since they are predictable and recursive [17]. However, non-systematic odometry errors are very difficult to analyze and compensate. There are methods available over the literature for non-systematic odometry error correction, nevertheless, they ensure a very limited improvement since it is impossible to propose a general method for varying environments.

Odometry is the primary technique for WMR positioning but it is still inherently erroneous that jeopardizes pose estimation of a robot in practical applications. In addition to odometric error compensation, complementary navigation techniques are frequently used (cascaded with odometry) in WMR applications. Such applications are usually named as "Sensor Fusion Navigation".

2.1.1.2 Inertial Navigation

Inertial navigation, which is also a dead-reckoning technique, utilizes accelerometer, gyroscope or compass sensor measurements to estimate rate of rotation and acceleration of the system. Those measurements are integrated over time (once or twice) to result position.

Accelerometers are used to measure linear accelerations, whereas gyroscopes are used to determine heading of a system. Compasses (or Magnetic Compass) act as a pointer to "magnetic north".

Inertial **N**avigations **S**ystems (INS) may consist of one or all of those sensors. Like odometric sensors, inertial sensors have both mechanic and electronic types but today, mechanical ones has been disfavored since they are expensive and geometrically larger.

Inertial **M**easurement **U**nit (IMU) generally consist of three accelerometers, three gyroscopes and three compasses for the relevant Cartesian axes are widely used in the field of robotics. Mobile robots, working in an unbounded environment like UAV's, Multi-Rotor drones and WMR's usually benefits from IMU for localization. A sample MEMS IMU is presented in Figure 2.3.

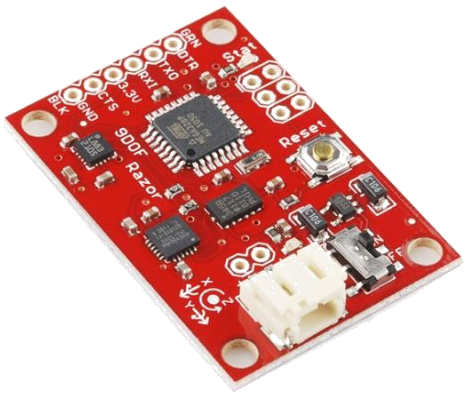


Figure 2.3: A MEMS IMU [18]

As the position calculation over inertial navigation sensor inherently depends on double integration, any small measurement error or noise grows unbounded over time. Therefore it can be said that IMU localization drifts with time.

Recent fiber-optic gyros (laser gyros) are very accurate but they are highly expensive. With the recent advances in micro-electromechanical systems (MEMS), small and lightweight inertial navigation systems are available for robotic applications today.

Like odometry, inertial navigation cannot be successfully utilized solely for robotic applications because of its erroneous nature. Today, inertial navigation technique and its sensors are usually combined with other navigation methods in order to improve localization performance.

2.1.2 Active Beacon Navigation

Active Beacon Navigation, which is an absolute positioning technique in Borenstein's dissection, computes the absolute position of the robot from measuring the direction of incidence of three or more actively transmitted beacons. The transmitters, usually transmit light or electromagnetic signals (IR or RF) where the receivers capture them.

In this technique, transmitters are located at known sites in the environment. These locations and environment can be terrestrial, underwater, celestial or spatial. The

transmitters are named as beacons. Since the beacons are active (i.e. transmitting signals) this technique is called as active beacon navigation or active beaconing.

Active beaconing uses Triangulation or Trilateration/Multi-Lateration techniques in order to compute one's position in the environment. Triangulation is named for the process of determining the location of a point by measuring angles to it from known fixed baseline, whereas trilateration/multi-lateration is known as the process of determining location of a point by measuring distances from fixed references.

Both of the processes use circle (or sphere) geometry to calculate distances or angles. For both, at least three circles must be drawn in order to create an intersection region. Therefore, active beaconing requires three or more transmittance in order to compute one's position in the environment. The "Tri" and "Multi" prefixes on those processes denotes the number of circles.

The three circles algorithm used in the trilateration procedure is schematically shown in Figure 2.4.

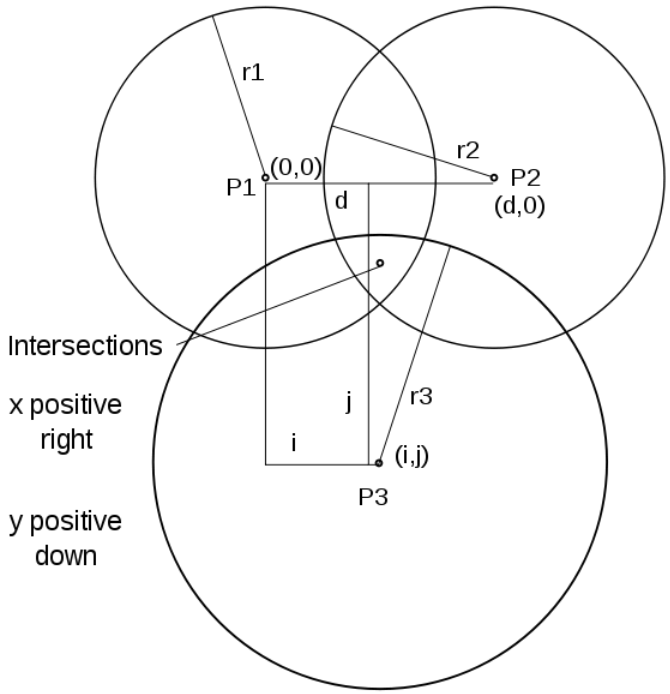


Figure 2.4: Trilateration Technique Schematic

The trilateration equations in 3D space is given in Equations 2.1 - 2.3.

$$r_1^2 = x^2 + y^2 + z^2 \tag{2.1}$$

$$r_2^2 = (x - d)^2 + y^2 + z^2 \tag{2.2}$$

$$r_3^2 = (x - i)^2 + (y - j)^2 + z^2 \tag{2.3}$$

Obtaining x from (2.1) - (2.3);

$$x = \frac{(r_1^2 - r_2^2 + d^2)}{2d} \tag{2.4}$$

By using the expression of x coordinate, y and z coordinate can be found as;

$$y = \frac{r_1^2 - r_3^2 - x^2 + (x - i)^2 + j^2}{2j} \tag{2.5}$$

$$z = \sqrt{r_1^2 - x^2 + y^2} \tag{2.6}$$

Active beaconing is a navigation technique as old as humanity. People observed stars for navigation for centuries; whereas lighthouses can be considered as the early human-build beacons. Today, star observations are replaced by **G**lobal **P**ositioning **S**ystems (GPS) and lighthouses are derived to sophisticated beacons for fundamental navigation purposes.

Satellite localization or satellite navigation is commonly used method for global positioning systems. **G**lobal **P**ositioning **S**ystem (GPS) and **GLO**bal **NA**vigation **S**atellite **S**ystem (GLONASS) usually uses 24 satellite for an earth-fixed positioning. In satellite localization method, satellites acts as active transmitters and send signals to GPS or GLONASS receivers. In the satellite signals, time and satellite position

information at the time of signal transmissions exists. With this information, GPS/GLONASS receivers computes its global position via **T**ime **o**f **F**light (TOF) computations within a bounded accuracy. This accuracy heavily depends on weather conditions, tall obstacle existence in the receiver environment and number of active satellites.

Today, this technique is widely used and GPS/GLONASS receivers can be found in many devices like cell phones, car navigation systems, tracking systems, etc. Since it provides an absolute measurement, GPS/GLONASS receivers are also frequently used in outdoor mobile robotic applications. A sample GPS/GLONASS receiver is presented in Figure 2.5.

Figure 2.5: A GPS/GLONASS Receiver for Robotic Applications [19]

The cost and utilization of a GPS/GLONASS receiver is comparably inexpensive. However, Byrne showed that the commercial devices for civilian use estimates one's position within a circle of 10-30 m radius [20]. This type of accuracy is not desirable for many WMR applications. Besides poor accuracy, Ohno et.al. showed that satellite navigation results in large position errors when there are high buildings or trees in the working environment [21].

GPS accuracy can be improved by an extension technique called **D**ifferential **G**lobal **P**ositioning **S**ystem (dGPS). DGPS uses a network of fixed, ground-based reference stations to broadcast the difference between the positions indicated by the satellite systems and the known fixed positions. These broadcasted signals are called correction signals. The illustration of dGPS method is given in Figure 2.6

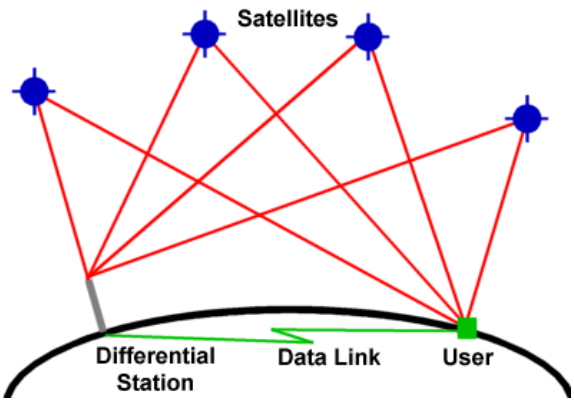


Figure 2.6: dGPS Schematic

With the aid of earth-fixed positioned beacons and correction signals, dGPS systems provides a localization accuracy of 1-5 m, which makes them more powerful and reliable compared with GPS systems. On the other hand, the valuable correction signals are not free to use. Commercial companies provides this correction services with a respectable price, which makes dGPS systems very expensive for many robotic applications. Also earth-fixed beacons are not globally available, therefore many countries could not able to benefit from those services.

A further improvement has been developed for dGPS which is called dGPS with **R**eal-**T**ime **K**inematics (RTK). In this technique, many fixed beacons send real time correction signals to the receiver. With dGPS-RTK, localization accuracy is improved to 1-80 cm, depending on the weather conditions and existing obstacles. A sample dGPS - RTK module is presented in Figure 2.7.

Figure 2.7: dGPS-RTK System [22]

Even though the dGPS-RTK is highly accurate, use of this system is very expensive. The initial hardware costs are about 15,000 USD. Moreover, 1,500 USD must be paid annually in order to get quality real time correction services in addition to subscription fee.

As a summary, GPS/GLONASS systems have poor accuracy and great deficiency for weather and environment conditions. dGPS and dGPS-RTK systems are very expensive for many robotic applications. They are mainly used for freight tracking. For those reasons, use of satellite navigation solely is disadvantageous for WMR applications, but it can act as a correction mechanism for relative positioning techniques like dead-reckoning.

Beacon localization, which can be considered as a micro-scale satellite navigation, is another commonly used technique, utilizing triangulation, trilateration or multi-lateration. This method also utilizes three or more transmitting fixed beacons and TOF computations in order to calculate one's position.

Among different types, RF Beaconing and **U**ltra**s**onic (US) Beaconing are the most common methods, used for WMR localization. Electromagnetic waves are sent and received in RF beaconing, whereas high speed sound waves are used in ultrasonic beaconing.

In both methods, the time passed between departure from transmitter and arrival to receiver is measured. The electromagnetic radio signals travel at the speed of light and ultrasonic waves travel at the speed of sound. Measuring the time between departure and arrival, multiplication with constant velocity leads as the distance. This is also mentioned earlier as TOF computation.

This technique also has some limitations. Both RF and US signals are affected by weather conditions. Moreover, localization accuracy changes inversely proportional to the distance between transmitters and receiver. For those reasons, this technique is commonly used in indoor environment.

2.1.3 Landmark Navigation

Like many other mentioned, landmark navigation techniques determine one's relative position to an external reference. Landmarks are geometrical objects, which have unique features that a robot can recognize from its sensors. They can be natural or artificial. Once landmark locations are known, this information can be conveniently used for localization, correction or for various aspects.

Landmark navigation can also be called as Vision-Based Navigation since this technique need a vision for identification. This vision is provided mainly by optical and magnetic sensors.

A landmark navigation system generally has those characteristics;

- A vision sensor for landmark detection and for comparison against background image
- A method for matching observed features with a map of known landmarks.
- A method of localization, computed from matches.

An artificial landmark creation is usually easier than natural type, because one can adjust necessary landmark contrast (compared with background), which increases accuracy of identification.

Figure 2.8: A LIDAR [23] and a Stereo-Camera Sensor [24]

One simple example of landmark navigation is path guidance technique. Line following robots can be shown as a basic implementation of path guidance technique.

In this method, electromagnetic wires, reflecting tapes, walls, thermal or metallic markers can be used as guides. This is a cheap and easy-to-implement technique but it needs a prior way (guide) to follow.

General disadvantage of landmark navigation lies on its nature. As mentioned, the critical limitation for this technique is that landmarks must have fixed and known location. It is very hard to use this technique solely for localization since the exact locations of external references are very hard to be known in many applications. For that reason, landmark navigation generally used for correction, mapping, and obstacle avoidance tasks.

RADARs (Radio Detection and Ranging) and LIDARs (Light Detection and Ranging) are widely used for mapping and obstacle avoidance tasks in WMR applications. A sample LIDAR, used for robotic applications is presented in Figure 2.8.

2.1.4 Map-based Navigation

Map-based navigation, also called "map matching", is a technique in which the robot utilizes sensory information to create a map of its local environment. This local map is then compared with a prior map previously stored in its memory. If this comparison results with a match, then the robot can compute its relative position and orientation in this local environment.

The prior map can be a geometric or phenomenological model of the environment, or it can be constructed from prior sensor data. Talluri and Aggarwal defined the basic requirements for a map as [25];

- It should provide a way to incorporate consistently the newly sensed information into the existing world model.
- It should provide the necessary information and procedures for estimating the position and pose of the robot in the environment.
- Information to do path planning, obstacle avoidance, and other navigation tasks must also be easily extractable from the prior map.

As other navigation techniques, map-based navigation also has advantages and disadvantages. The main advantages of map-based navigation can be listed as;

- Localization without modifying the environment.
- Can be used to generate and updated map of the environment.
- Allows a robot to learn a new environment and to improve positioning accuracy through exploration.

On the other hand, disadvantages of this technique;

- There may not be enough stationary, unique features that can be used for map matching.
- Sensory data must be accurate to be useful.
- Real time computations require huge amount of computational power.
- Required sensors may need to be expensive.

With all this pros and cons, map-based navigation technique is generally considered as useful for structured environments (like laboratories), yet it has an inadequacy for real, unstructured environments.

2.2 Coverage Methods for Mobile Robots

CPP is the task of determining a path that passes over all points of an area or volume of interest while avoiding obstacles [26].

Cao et al. defined the requirements from CPP as [27];

- Robot must move through all the points in the target area covering it completely.
- Robot must fill the region without overlapping paths.
- Continuous and sequential operation without any repetition of paths is required.
- Robot must avoid all obstacles.
- Simple motion trajectories should be used for simplicity in control

- An optimal path is desired under available conditions

Those requirements are essential for a quality coverage, however, it may not be possible all the time to satisfy all the criterion. Therefore a priority assignment has to be made in most of times.

Coverage algorithms can be classified as heuristic or complete depending on whether or not they provably guarantee complete coverage of the working environment. Independently, they can be classified as either off-line or on-line [15].

This section covers the state-of-art coverage path planning methods in detail.

2.2.1 Heuristic and Randomized Coverage Methods

In contrast to deterministic methods, heuristic or randomized approaches do not plan search or coverage paths, they rather select their actions when needed.

Behaviors like line following, wall following and obstacle avoidance are generally considered as heuristics. Even though these heuristics are commonly used in deterministic methods, they are essential and vital for random coverage algorithms. For example; a randomly operating robot requires a border to limit the working environment. This border may be a landmark, a wall or some other distinguishable element. Heuristic behavior provides robot not to cross that border.

There is no commonly used random coverage algorithm template in the literature. It will vary for different applications and working environments. A primitive random CPP approach is to change orientation of robot when some heuristic action is needed. Nearly all of the random operating robots orient their heading in a random angle when they reach to a border. In some, this rotation angle is bounded regarding robot's previous actions and knowledge.

Since each action for random CPP results with a random response, it can be said that random CPP does not guarantee a complete (full) coverage. Although random CPP is inadequate to cover entire area, there are many advantages of this method. Balch [28]

and Gage [29] have analyzed randomized CPP from a cost/benefit perspective. Balch argues that random CPP neither require expensive sensory hardware, nor consume computational resources for localization. He also showed over simulations that a random CPP architecture can be built up with one-fifth cost with equivalent coverage performance, when compared with some deterministic methods.

With its low cost/benefit ratio, randomized CPP methods are widely used for various applications like floor cleaning, window cleaning, lawn mowing, de-mining, etc. Even though it has some inadequacies, randomized CPP methods are efficient, beneficial, and provide better overall performances compared with many deterministic CPP methods.

2.2.2 Complete Coverage Methods

Complete CPP methods are deterministic methods that ensure the full coverage of entire workspace. Even though there are additional techniques like Morse-Based Coverage, Landmark-Based Coverage, Graph-Based Coverage, etc. this section reveals some common complete coverage techniques.

2.2.2.1 Cellular Decomposition Methods

Cellular decomposition methods divides workspace into simple non-coincident sub regions. Those regions, which also name the method are called cells. The union of all cells partially or completely fills the workspace and each cell is assumed to be "obstacle free".

The main idea in cellular decomposition is to break overall area into small pieces, which are easy to cover by simple motions i.e. zig-zag, back-forth, spiral, etc.

In cellular decomposition, two cells are thought to be adjacent if they share a common boundary. Generally, an adjacency graph is used to represent the cellular decomposition, where a node represents a cell and an edge represents an adjacency relationship between two cells.

Cellular decomposition algorithms can be approximate, semi-approximate and exact. In an approximate decomposition, cells are all identical (same height and length) and the union of all only approximates the entire workspace.

Semi-approximate cellular decomposition relies on a partial discretization of space, where cells are fixed in width, but top and bottom lines may have any shape. Semi-approximate method provides a better approximation of workspace. As it might be expected from its name, exact cellular decomposition techniques create cells and their union completely cover the workspace.

Two well-known exact cellular decomposition approaches are discussed below.

Trapezoidal Decomposition

One of the simplest exact cellular decomposition techniques which can yield a complete coverage path is the trapezoidal decomposition. This method is applicable only 2D polygonal spaces.

In the trapezoidal decomposition, each cell is a trapezoid and simple back-and-forth motions can be used to cover each cell. Complete coverage is guaranteed by finding an exhaustive walk through the adjacency graph associated to the decomposition. As a result, a specific zigzag path to cover each cell is generated. Trapezoidal decomposition schematic is presented in Figure 2.9.

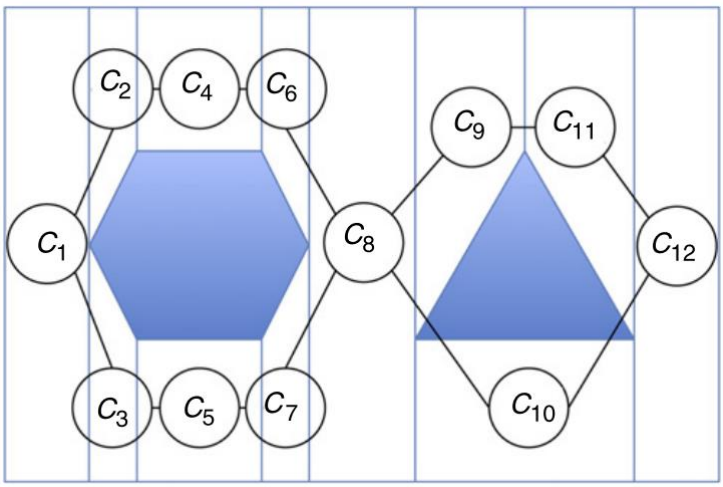


Figure 2.9: Trapezoidal Decomposition with Adjacency Graph.

Boustrophedon Decomposition

One disadvantage of the trapezoidal decomposition is that it generates many cells that, intuitively can be merged together to form bigger cells. This is clearly inconvenient, since as number of cells increases, final coverage path becomes longer. The main reason is that trapezoidal decomposition creates only convex cells. However, non-convex cells can also be completely covered by simple motions.

To overcome this limitation, Choset and Pignon proposed the boustrophedon cellular decomposition [30]. Boustrophedon means “the way of ox” in English. When an ox draws a plow in a field, it crosses the full length of the field in a straight line, turns around and then traces a new straight line path adjacent to previous one [15].

The boustrophedon decomposition is similar to the trapezoidal decomposition, but it only considers vertices where a vertical segment can be extended both above and below the vertex. The vertices where this occurs are called critical points.

By adhering to this strategy, the boustrophedon decomposition effectively reduces the number of cells in trapezoidal decomposition. Hence, shorter coverage paths are obtained. This method assumes polygonal obstacles and the borders to be known.

Figure 2.10 shows the advantage of boustrophedon decomposition. Extra strips are not needed in boustrophedon decomposition, therefore it results with less cells and shorter coverage path.

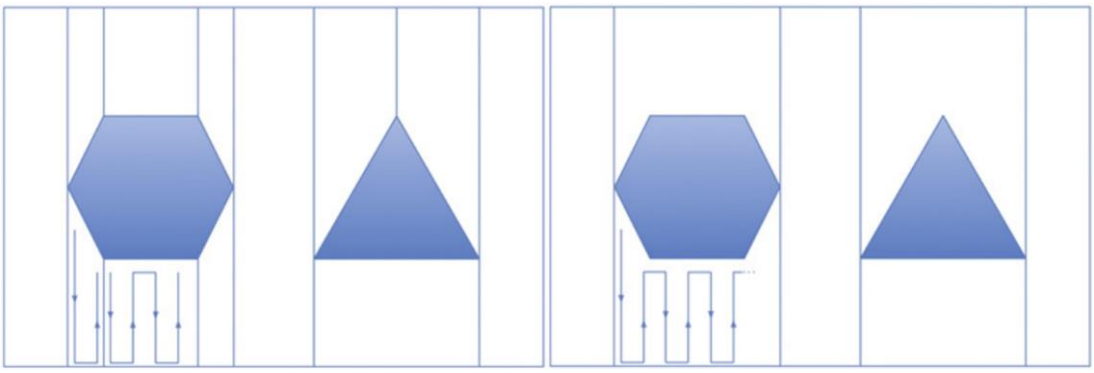


Figure 2.10: Trapezoidal (Left) and Boustrophedon (Right) Decomposition

2.2.2.2 Grid - Based Methods

Grid-based methods decomposes working environment with identical sized, uniform grid cells. In this representation, each grid cell has an associated value stating whether an obstacle is present or if it is rather free space. The value can be either binary or a probability. Typically, each grid cell is a square, but also different grid cell shapes can be used, such as triangles. As grid representations only approximate the shape of the target region and its obstacles, these methods are called as approximate cellular decompositions. As a result of this approximate representation their completeness depends on the resolution of the grid map. Sample grid map is presented in Figure 2.11.

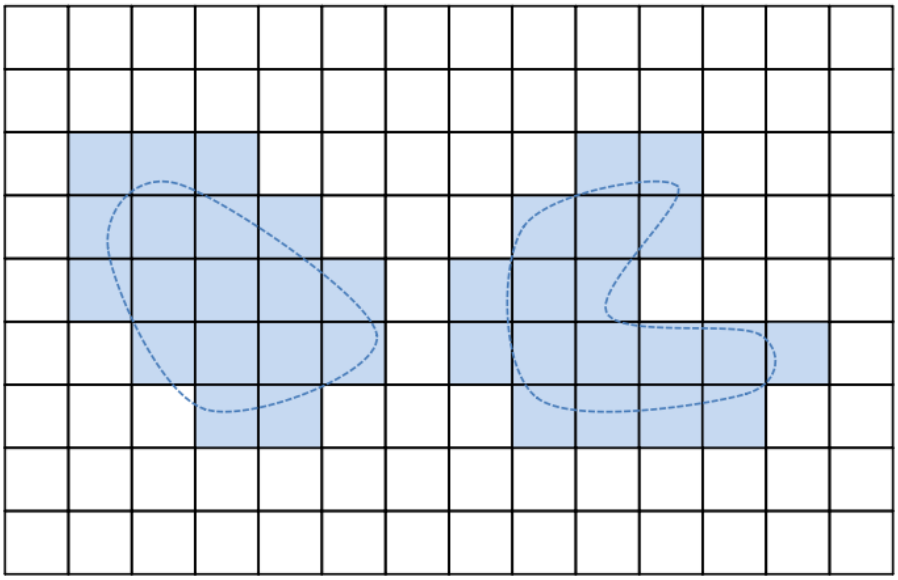


Figure 2.11: Grid-Based Decomposition Map

Due to its easy-to-use nature and simplicity, the grid-based representations are the most widely used coverage algorithms. Nonetheless, grid maps suffer from exponential growth of memory usage because the resolution remains constant regardless of the complexity of the environment. Also, they require accurate localization to maintain the map's coherency [26].

For these reasons, grid-based coverage methods are suited for indoor mobile robot operations, where the size of the area to be covered is relatively small.

2.3 Autonomous Lawn Mowers

ALMs are popular domestic robots and being studied globally for both academic and commercial purposes. This section introduces ALMs from both academic and commercial point of views.

2.3.1 Academic Value

ALMs are perfect research platforms, when outdoor navigation is considered. Mapping of unknown environments, accurate SLAM, path-planning and obstacle avoidance are the major topics for these types of researches.

Institute **of N**avigation (ION) Satellite Division organizes an ALM competition in US annually [31]. The purpose of this competition is to design and operate a fully autonomous lawn mower using the state-of-art navigation techniques to mow working environment accurately [32].

ION autonomous lawn mower competition is held in two classes; basic and advanced mowing competitions. Both competitions are aimed to award the winner who performs the "best cut". The "best cut" definition here stands for a minimum 75% grass-cut of the given field.

As mentioned before, the purpose of this competition is to use art and science of navigation. Therefore, participant robots capabilities are different compared to the conventional ALMs. As mentioned in the previous section, all of the commercial ALMs use perimeter wire in order not to pass to neighbor’s yard or not to ruin any flowerbed. Almost all of them operate with random coverage patterns. None of them use any external correction reference. Many of them have just bump-switches for obstacle avoidance.

All participant robots for ION competition use wheel encoders for dead-reckoning. Many of them use expensive LIDAR equipment for mapping and obstacle avoidance challenges. Perimeter wire type bordering is not allowed in this competition, so the

robots must calculate border positions throughout its operation; otherwise the team will get a penalty.

All participant robots use GPS for position correction. Moreover, the teams also allowed to use visionary sensors for corrections. With this expensive hardware usage, the average budget of a team is about 20,000 USD. Most of the budget is spent for hardware in this competition.

ION competition rules and descriptions are:

- Lawnmowers shall be autonomous and unmanned and shall not be remotely controlled during the competition.
- Lawnmowers shall have a maximum speed of 10 km/h for safety reasons.
- Lawnmowers shall not exceed 2 meters in any dimension.
- Lawnmower movement shall be accomplished through direct contact with the ground.
- Lawnmowers shall demonstrate the ability to mow a predetermined path void of any obstacles.
- Teams shall have a maximum of 20 minutes to cut the field.
- Mowers shall be designed to operate in any weather condition.

As mentioned before, there are two types; basic and advanced mowing competitions.

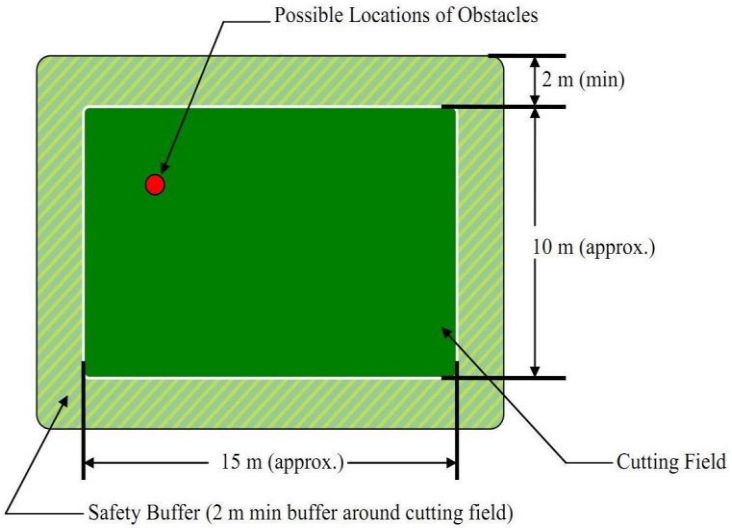


Figure 2.12: ION Competition Basic Field

The basic field is 10 x 15 m^2 with 2 m buffer zone in each side. Field contains a static obstacle such as a standard size plastic container. The drawing of the basic field is given in Figure 2.12.

The advanced field has an irregular shape with at least one non-perpendicular side, a flowerbed and a dynamic obstacle. The drawing of the advanced field is given in Figure 2.13.

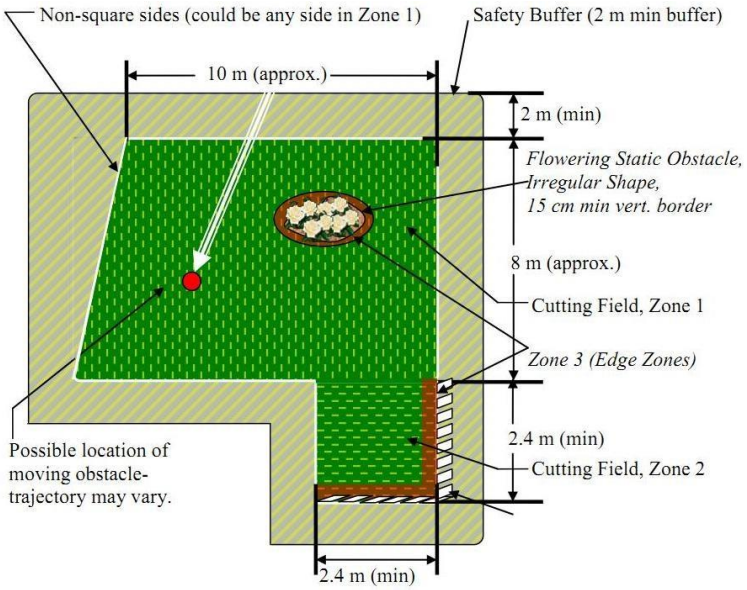


Figure 2.13: ION Competition Advanced Field

Precise localization requires expensive sensors as mentioned. For to visualize, navigation hardware configuration of the winner team in 2012 ION autonomous lawn mower competition is given in Table 2.2 [33] [34].

Table 2.2: Sensor Pricing for ION 2012 ALM Competition Winner

Sensor Hardware	**Price [USD]**
dGPS	30,000.00
Encoder	170.00
IMU	2,400.00
LIDAR	5,000.00

Although being a sponsored event, sensor budget is too high as seen in Table 2.2. It is impossible to convert such platforms into commercial products.

2.3.2 Commercial Products and Specifications

Starting from the 1990's, a necessity arise for robots to take care of the house chores as the time spent on those chores decreases inevitably. First commercial autonomous products sold globally for house chores are lawn mowers. Major commercial ALMs are investigated in this section from a broad perspective. Navigation techniques, used equipment and specifications of these robots are compared in detail.

Approximately 10 main ALM manufacturers, with more than 25 products, are available on the global market today. A detailed comparison that reveals the top products of the top 6 manufacturers are given in Table 2.3.

All of these commercial robotic mowers are powered by internal batteries. Almost all of them are differentially driven by 2 DC motors and utilize two castor wheels for balancing. In order to lower the manufacturing costs, almost all of the body components are made by plastic materials with injection molding or vacuum forming processes.

It is not a cost effective solution to gather clipping for an ALM. All of the commercial products use mulching technique for lawn mowing. In mulching, grass is cut into very small clippings - size about a millimeter - which are buried in the roots of the lawn, where they decompose and act like a natural fertilizer. This technique results in a healthier and better-looking lawn, and major advantage is that it eliminates the need to collect and remove the clippings.

On contrary to the conventional man-powered (push behind) lawn mowers, most of the ALMs do not have a single large cutting blade. Instead, they use multiple smaller blades mulching.

The cutting height is manually adjustable in all ALMs. In average, lawn height can be kept around 20-100 mm, depending on the ALM model.

Company	HUSQVARNA	Friendly Robotics	BOSCH	John Deere	HONDA
Models	220 AC	RS630	Indego	Tango E5	Miimo 500
Image					
Coverage [m2]	1800	2900	1000	1800	3000
Max. Inclination [°]	19	20	19	19	15
Battery Type	NiMH	LI-ION	LI-ION	LI-ION	LI-ION
Charging Time [min]	45	120	50	90	60
Mowing Time [min]	45	60	50	60	45
Cutting System	3 Blades	3 Blades	3 Centrifugal Blades	1 x 4 Sided Blade	1 x 3 Sided Blade
Cutting Height [mm]	20-61	20-81	20-60	19-102	20-60
Cutting Width [mm]	221	560	260	-	220
Sound [dB]	64	74	75	69	70
Dimensions [mm]	711x533x305	736x660x305	700 x 520 x 300	775 x 535 x 360	645x550x275
Weight [kg]	10	20	11,1	15,3	11,8
Charge Station	Yes	Yes	Yes	Yes	Yes
Remote Control	No	No	No	No	No
Online Price [$]	2700	2500	2100	2250	2500

Table 2.3: Top Commercial ALM Comparison Chart

The navigation techniques of all the commercial ALMs are quite similar. They all have a charging station from where they start their operations. There is a limited IR communication between the charging station and the robot used for the robot to find charge station when the batteries have nearly consumed. They have no expensive visionary sensors like LIDARs or cameras to inspect the environment. Even though some of them use inertial sensors to make the localization more accurate, they all operate by dead-reckoning with their odometric wheel encoders. For this reason, the robots to operate mainly on random patterns which jeopardizes a complete coverage.

In order to border working are, all of the commercial ALMs use the same method. This method consists of a perimeter wire and an inductive sensor. A perimeter wire is simply a harmless low-current passing wire. The user connects both ends of this wire to the charging station to obtain a closed current loop. The wire is buried only a few millimeters deep in the ground along the perimeter of the desired area. During their operations, robots sense the borders with the aid of the inductive sensors located at the bottom of their body. When they arrive to a border, they act with respect to their predefined motion algorithms.

Static or dynamic obstacle avoidance is achieved by using US, IR sensors or bumper switches in all commercial products.

The average cost of an ALM is about 3,000 USD without shipping fees and taxes. This price may be seemed a little expensive for some people but sales records for autonomous lawn mowers definitely shows the opposite.

ALM technology has proceeded and the market has emerged within the last decade, yet, there is a lot to do in order to make these robots more intelligent and affordable.

CHAPTER 3

DESIGN CONSIDERATIONS

Chapter 2 presents a detailed literature survey of mobile robot positioning and coverage path planning methods. In this chapter, these methods are comprehensively discussed and design arguments for ALM are revealed.

3.1 Problem Definition and System Requirements

There are four major and indispensable requirements for a commercial autonomous lawn mower. An autonomous lawn mower;

- Must mow in good quality and care the lawn
- Must be mobile
- Must be autonomous
- Must be affordable

Lawn mowing and lawn care are basically related with mechanical design of the structure. This task will be covered in further sections. The others can be generalized as a requirement of autonomous mobility for affordable prices.

As mentioned earlier, the main performance metric for an ALM is its navigation success. Since ALMs are outdoor robots, the main problem to be solved can be summarized as autonomous outdoor navigation. The concept is kept as the foundation of the design decisions throughout this work.

Considerations and selections for mobility, navigation technique, obstacle avoidance technique and lawn mowing technique are presented in following sections.

3.2 Concept Selection

3.2.1 Mobility

Conventional man-powered lawn mowers usually utilizes Internal Combustion Engines (ICE) to drive their mowing blades. ICEs can create large amounts of power, however they are inefficient and have high running costs. The user must also refill the fuel almost before every operation, which is an undesired task.

Besides ICE, there are also electricity powered conventional lawn mowers, however their energy requirements are very high so that they usually cannot operate with internal batteries. Nearly in all of them, electric is supplied from an external source with a power cable. This cable always rambles to the mower or to the user, which is also undesired.

When mobility concerned, an internal source of energy is compulsory for an ALM. LI-ION or LI-PO batteries have superior advantages over NI-MH and NI-CD batteries when their power-to-weight ratios are considered. In our ALM, LI-ION and LI-PO types are compared and LI-ION batteries are selected to be used for drive and mowing motor, since they are safer and more reliable.

3.2.2 Navigation Technique

Localization and coverage path planning techniques are the key features for an ALM navigation. Many different positioning and CPP methods are presented in Chapter 2.

For localization, active beaconing is evaluated. It is known that trilateration technique is successful in indoors, whereas RF and other signals can be influenced from weather conditions, when outdoor operation is considered. Moreover, the installation of beacons to working environment can be serious problem. As this ALM is intended to be a commercial product, it needs to be plug-and-play. Customers usually do not want artificial beacons on their gardens. Also this beacons will inherently increase price of the product. For those reasons, beacon localization is found disadvantageous.

Although beacon localization is determined not to be used in ALM, another active beaconing technique - satellite navigation is decided to be used. Since its localization accuracy is poor, a low-cost, commercial GPS receiver is added to the system for error compensation in comparably large working areas.

Landmark navigation brings some difficulties with itself inherently. For sole localization, exact positions of the external references must be known. It is almost impossible to apply this method to every single garden for the same reasons, considered for beacon localization. Creating some artificial landmarks seemed a little inappropriate for a domestic product. In addition, landmark navigation requires comparably expensive sensors like LIDARs or high definition stereo cameras and demands great computational power. Landmark navigation technique is also found inapplicable for a commercial ALM. Lastly, map-based navigation techniques are found irrelevant, since prior map information of every operation environment is impossible to be known.

The localization for ALM has chosen to be performed by dead-reckoning technique. Advantages of both odometric and inertial DR techniques are determined to be utilized. For this purpose, wheel encoders and an IMU is decided to be used on ALM.

When it comes to the coverage path planning, technique selection heavily depends on whether the prior border information in known or not. Almost all of the commercial ALMs operates in relatively small sized gardens. Those gardens are inherently limited by roads, flowerbeds, trees or fences, so that they are somehow bordered. It is found convenient to border the working environment of ALM by means of a perimeter wire. Perimeter wire is an electric wire that is drawn over the desired perimeter of the working area. It is a safe, accurate and low cost solution for bordering, but it naturally needed to be sensed by robot for not to override. In our ALM, perimeter wire sensing is determined to be performed by an inductive sensor.

CPP methods can be heuristic, randomized and complete. Heuristic CPP techniques needs a prior way to follow. This way could be a wire, a wall or a paint mark. None of them are applicable for a commercial ALM, since the whole are must be marked in advance for to robot finds its way.

Randomized CPP gives the decent coverage results in many environments, where it does not need any sophisticated algorithms or expensive sensors to navigate. In randomized ALM applications, all the robot does is to rotate with a random heading angle when reached to a border.

Although randomized CPP is advantageous for its simplicity and low-cost requirements, this technique cannot guarantees a complete coverage. It may works fine over simple rectangular environments but most of the areas can be left unmowed, especially for “U-shaped” or “O-shaped” complex garden geometries.

For to overcome uncertainties of random coverage, deterministic CPP methods has also been utilized in commercial ALMs. Those deterministic methods mainly consist of geometric patterns. Parallel swath and spiral-shaped coverage algorithms are widely used in commercial ALMs solely or cascaded with randomized algorithms.

Deterministic CPP methods are complete coverage methods, when operating environment border information is known. For that reason they seem advantageous over randomized algorithms in first glance, however, they either suffer from wheel slippage or numerical drift. As a result, instantaneous pose estimation of robot can deviate to actual pose.

The key advancement in navigation technique of ALM is in its CPP method. In this work, a unique CCPP method is proposed for to overcome disadvantages of randomized and complete CPP techniques. Mathematical and implementation details of this technique is presented in Chapter 4. In addition to CCPP, deterministic geometric patterns and randomized path planning techniques are also implemented to ALM for to reveal theoretical and practical coverage success of each.

3.2.3 Obstacle Avoidance Technique

Every single garden may have many static or dynamic obstacles. In order not to collide with these obstacles, some sensory equipment such as IR sensors, US sensors, LIDARs, RADARs, etc. can be used for mobile robotics. LIDAR and RADAR sensors are highly accurate, but they are extremely expensive for our purpose.

US sensors are advantageous for their capability of detecting transparent objects, but low-cost commercial US sensors usually produce noisy outputs, which may mislead the location of obstacles.

In this ALM, low-cost IR sensors are decided to be used. An IR sensor skirt, which covers ALM's entire frontal area has decided to be built up. IR sensors details, layout and details of obstacle avoidance algorithm are presented in Chapter 4.

3.2.4 Lawn Mowing Technique

Mulching is chosen as the lawn mowing technique for the ALM. Mulching is an agricultural term, in which, grass is cut into very small clippings and buried in the roots of the lawn. Mulched lawn clippings decompose in soil and act like a natural fertilizer in time. This technique results in a healthier and better-looking lawn. The major advantage of this technique is that it eliminates the need to collect and remove the lawn clippings.

In a conventional man-powered lawn mower that does not use mulching, lawn clippings are collected in a basket. The volume of a conventional basket is about 40 L. It can carry the clippings of an approximately 50 m^2 lawn yard. After mowing this area, the user must empty the basket. The mass of a basket full of lawn clippings is about 6 kg.

It is found inconvenient for an ALM to collect lawn clippings since they add extra weight to the system, which means a decrease on the operation time inefficiently. Moreover, the main idea of autonomous lawn mowing is regaining the time spent on

lawn mowing to the user. This goal cannot be achieved if the user periodically empties the lawn basket of a robot.

Therefore, mulching technique is selected for lawn mowing in this study. A single, three-sided cutting blade has determined to be used for energy efficiency and easy-cleaning purposes.

CHAPTER 4

SYSTEM DEVELOPMENT

4.1 Hardware Design

4.1.1 Selection of System Components

4.1.1.1 Motors and Motor Drivers

It has been stated in previous chapter that the ALM is decided to be differentially driven by two DC electric motor, whereas the lawn mowing operation to be performed over a single DC motor. For selection of drive motors, power, torque, and speed requirements are analytically calculated. Free body diagram that provides a foundation for calculations is presented in Figure 4.1.

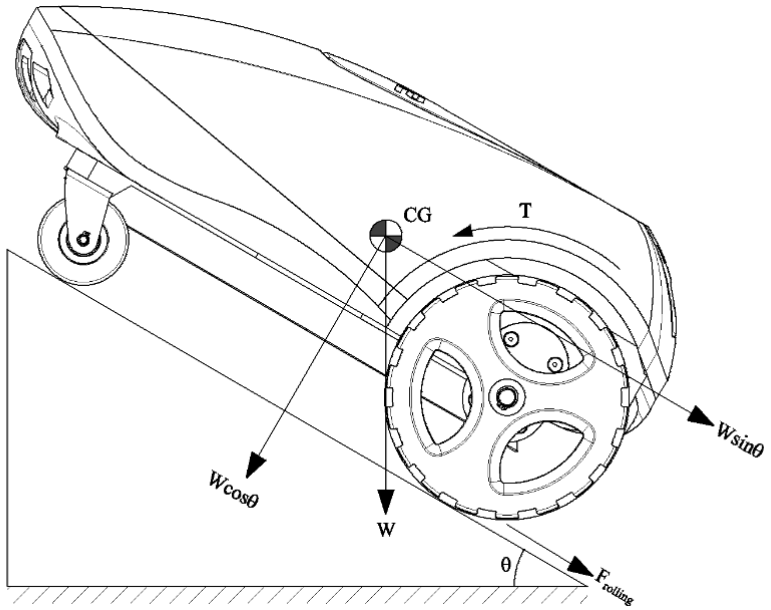


Figure 4.1: Free Body Diagram for the Autonomous Lawn Mower

Calculations are made for a single wheel with no slip condition. Parameters for initial calculations are given in Table 4.1.

Table 4.1: Parameters Used For Drive Motor Selection

Parameter	**Value**
System Mass, m [kg]	20.0
Rolling Resistance, R_r	0.1
Max. Inclination, θ_{max} [deg]	20
Max. Velocity Over Inclined Surface, $\vartheta_{max,inc}$ [km/h]	1
Max. Velocity Over Flat Surface, $\vartheta_{max,flat}$ [km/h]	1
Wheel Diameter, d [mm]	200.0

The net force acting on direction of motion for a single wheel;

$$F_{net} = \frac{Wsin(\theta_{max})}{2} + F_{rolling} \tag{4.1}$$

where, W is the gross weight of the system, θ_{max} is the maximum inclination angle of the robot and $F_{rolling}$ is the rolling resistance (force) acting on a single wheel. Rolling resistance is denoted as;

$$F_{rolling} = \frac{Wcos(\theta_{max})R_r}{2} \tag{4.2}$$

The minimum power and torque requirements for a single traction motor are calculated using;

$$T_{min} = F_{net}\frac{d}{2} \tag{4.3}$$

$$P_{min} = T_{min}\omega \tag{4.4}$$

where, ω is the maximum angular speed of a wheel, derived from ϑ_{max} and d.

Using the parameters given in Table 4.1, minimum power, torque and angular velocity requirements from a drive motor are calculated as;

$$P_{min} = 11.9 \text{ W} \tag{4.5}$$

$$T_{min} = 4.3 \text{ Nm} \tag{4.6}$$

$$\omega_{min} = 26.5 \text{ RPM} \tag{4.7}$$

A relatively high gradeability requirement leads the torque demand to be high. Therefore a gear-motor is decided to be used for traction of the ALM. The selected gear motor and its properties are given in Table 4.2.

Table 4.2: Properties of Selected Drive Motor

Producer / Model	Shenzhen Dongshun / 37RS5550246000
Rated Voltage [VDC]	24.0
Rated Power [W]	12.0
Rated Torque [Nm]	2.0
Rated Current [A]	1.0
Stall Torque [Nm]	12.5
Rated Speed [RPM]	56.5
Maximum Speed [RPM]	67.8
Gearbox Type	37 D, 4 Stage Spur Gear
Reduction Ratio	1:90

Figure 4.2: Selected Drive Motor

Since the resistance of grass depends on its type and weather conditions, it is hard to determine the power and torque requirements from the mower motor. Several test and comparisons were made for the selection of mower motor. Specifications for the selected mower motor is presented in Table 4.3.

Table 4.3: Properties of Selected Mower Motor

Producer / Model	RW-ML-1212
Rated Voltage [VDC]	24.0
Rated Power [W]	72.0
Rated Current [A]	3.0
Peak Current [A]	3.0
Rated Torque [Nm]	0.1
Stall Torque [Nm]	0.5
No Load Speed [RPM]	6200

Figure 4.3: Selected Mower Motor

For traction motors, a dual DC motor driver has decided to be used. A powerful and robust conventional motor driver is selected for this purpose. The properties of the selected drive motor driver is given in Table 4.4.

Table 4.4: Properties of Selected Drive Motor Driver

Producer / Model	Sabertooth / 2 x 25 A
Input Voltage [VDC]	6.0 - 24.0
Output Voltage [VDC]	6.0 - 24.0
Continuous Output Current [A]	12.0 per channel
Peak Output Current [A]	25.0 per channel
Rated Torque [Nm]	0.1
Stall Torque [Nm]	0.5
No Load Speed [RPM]	6200

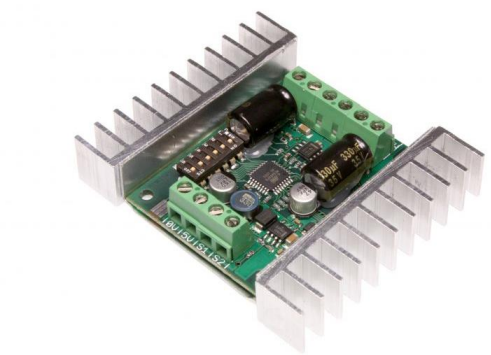


Figure 4.4: Selected Drive Motor Driver

Since no speed adjustments are needed for the mower motor, a simple on/off control is found sufficient. For that purpose a MOSFET transistor is used to control 24V mower motor from TTL level controller side.

Table 4.5: Properties of Selected Mower Motor Transistor

Producer / Model	IRLIZ44N
Continuous Current [A]	30.0

Power Dissipation [W]	45.0
Temperature Range [°C]	-55 to + 175

Figure 4.5: Selected Mower Motor Transistor

4.1.1.2 Batteries

The power consumption and desired operation time are considered for battery selection phase. Consumptions for mower motor, drive motors and other electronics are summed up to develop power budget. Rated consumptions for electric devices in the ALM is given in Table 4.6.

Table 4.6: ALM Rated Power Consumptions

Component	**Rated Power Consumption [Wh]**
Drive Motors	48.0
Mower Motor	72.0
Electronics (sensors, processors, etc.)	4.8

It is concluded that a total of 125 W is needed to operate ALM for one hour. The maximum duration of operation for ALM is defined as 1.5 h. Therefore a minimum of 187.5 Wh energy must be provided from batteries in order to meet operational requirements.

Besides being durable and powerful enough to supply current demands, lightweight batteries are intended to be used for to reduce overall weight of the ALM. For this

reason LI-ION and LI-PO battery types have been investigated. No satisfactory ready-to-use products can be found in the market so that, a lumped battery block is determined to be produced from single cells.

The selected Panasonic LI-ION cell is presented in Figure 4.6. This cell provides 3400 mAh for 3.7 V output.

Figure 4.6: Selected LI-ION Cell

Both drive and mower motors of ALM is chosen to be operated over 24 VDC. Therefore battery block is intended to be built as 24 VDC for not to waste energy by transformation. A total of 18 cells are used to produce 22.2 V, 10.2 Ah battery block (6S, 3P). This block provides 226.44 Wh power, which approximately leads to a 1.8 h operation time. The schematic and the produced battery block is given in Figures 4.7.

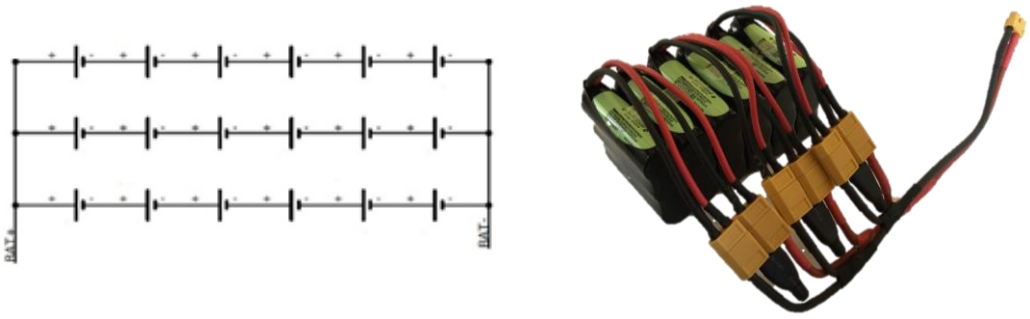


Figure 4.7: Battery Block Schematic and Produced LI-ION Battery Block

4.1.1.3 Sensors

4.1.1.3.1 Encoders

Encoder, coupled with the selected gear-motors are decided to be used for odometric calculations. Two identical optical, hallow-shaft type, quadrature wheel encoders are used in the system. The device provides one, two or two plus index incremental encoding square wave signal outputs. Properties of the selected encoder are presented in Table 4.7.

Table 4.7: Properties of Selected Wheel Encoders

Producer / Model	Shenzhen Dongshun
Type	Optical, Hollow Shaft
Output Channels	Quadrature + 1 Index Channel
Encoder Resolution [CPR]	13
Input Voltage [VDC]	5

4.1.1.3.2 Inertial Sensors

In order to make pose estimation of ALM more accurate, inertial sensors are intended to be used with wheel encoders in dead-reckoning procedure. A 9 DOF MEMS IMU is utilized where it has three accelerometers, three gyroscopes and three compasses on board for corresponding coordinate axes. The selected IMU and its specifications are given in Table 4.8.

Table 4.8: Properties of Selected Inertial Sensor

Producer / Model	Sparkfun / Razor 9 DOF
Type	MEMS
On Board Processor	ATmega328

Acceleration Limit [g]	±16
Input Voltage [VDC]	3.5 - 16.0
Sensors	ITG-3200 gyroscope, ADXL345 accelerometer, HMC5883L magnetometer
Dimensions [mm]	28 x 41 mm

Figure 4.8: Selected Inertial Sensor

4.1.1.3.3 Infrared Sensors

For obstacle avoidance, IR sensors are evaluated since they are extremely inexpensive and easy-to-utilize. A very common IR sensor has chosen to be used in ALM, whose specification are given in Table 4.9. The measurement range of IR is determined by considering robot's speed and sensor location on ALM.

Table 4.9: Properties of Selected IR Sensor

Producer / Model	Sharp / GP2Y0A21YK0F
Output	Analog
Measurement Range [cm]	10 - 80
Input Voltage [VDC]	4.5 - 5.5

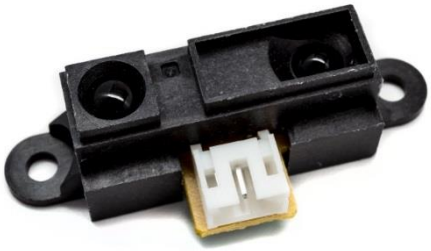


Figure 4.9: Selected IR Sensor

4.1.1.3.4 GPS

The ALM is designed to operate in comparably large areas like soccer fields or parks. For large area coverage, localization errors are intended to be corrected and compensated by a GPS sensor. Details of selected GPS sensor are listed in Table 4.10.

Table 4.10: Properties of Selected GPS Sensor

Producer / Model	Sparkfun / LS20031
Number of Satellites	22 Active, 66 Searching
Patch Antenna Size [mm]	15 x 15 x 4
Frequency [Hz]	10
Position Accuracy [m]	< 3.0

Figure 4.10: Selected GPS Sensor

4.1.1.3.5 Inductive Sensors

As been stated in previous chapter, ALM needs a bordered environment to work on. In this context, borders are electric cables, in which a low voltage current passes through it. Therefore a sensor that perceives the magnetic field on the wire is needed.

There are no commercial products in the market for this application. Thereby a unique, hand-made sensor has been designed, developed and produced. This sensor consists of a coil, which is inducted by the magnetic field wire, an amplifier used to amplify the magnitude of received signal, and other necessary integrated electronic components. The electrical circuit schematic and finished inductive sensor are presented in Figures 4.11 - 4.12.

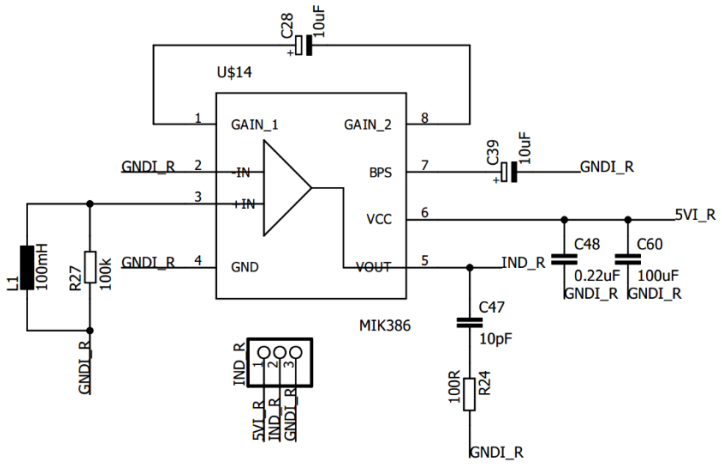


Figure 4.11: Electrical Circuit Schematic of Inductive Sensor

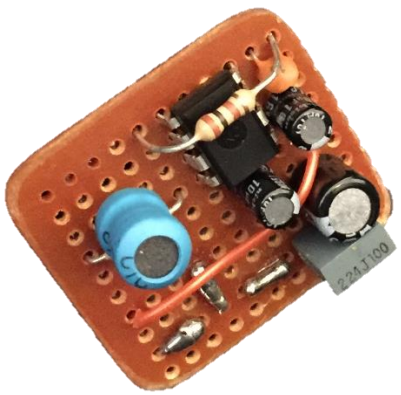


Figure 4.12: Produced Inductive Sensor

4.1.1.3.6 Current Sensors

In order to protect drive motors, mower motor and batteries, simple low-cost current sensors are used in ALM. A hall effect-based linear current sensor is selected, whose properties are given in Table 4.11.

Table 4.11: Properties of Selected Current Sensor

Supply Voltage [VDC]	8
Overcurrent Tolerance [A]	30
Bandwidth [kHz]	80

Figure 4.13: Selected Current Sensor

4.1.1.4 Controllers

Since the ALM is developed as an application platform regarding its early phase, controller architecture is selected from easy-to-utilize, low-cost hardware. Open-source Arduino platforms are found to be the best for this purpose when availability, interchangeability, compactness and computation capabilities are sought.

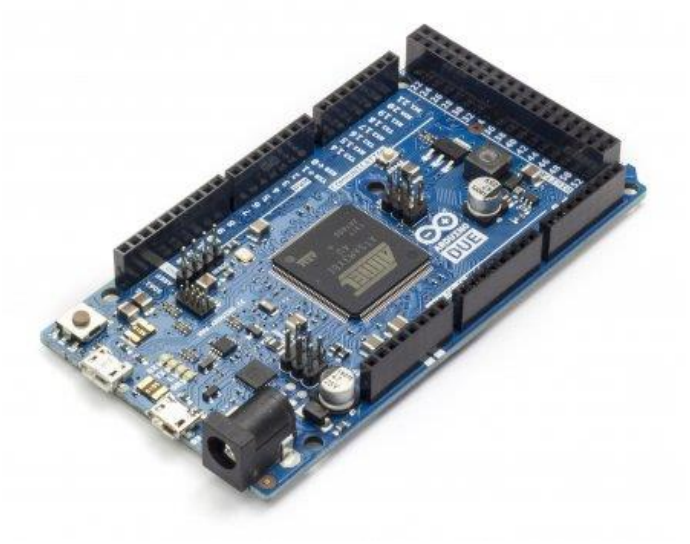


Figure 4.14: Arduino Due

Arduino is a single-board microcontroller, consisting of a physical programmable circuit board and an **I**ntegrated **D**evelopment **E**nvironment (IDE), used to write and upload the code to the physical board.

Even though some Arduino microprocessors are eligible to perform all the necessary computations required in ALM, it is experienced that a single microprocessor usage can easily suffer from nested hardware interrupts occurred in the system. When this happens, microprocessor misses some precious information coming from sensors.

For this reason, controller architecture is designed in hierarchical master-slave configuration. The master controller is selected as Arduino Due. All high level codes such as navigation, trajectory planning and control algorithms run on this board. Specifications of Arduino Due is listed in Table 4.12.

Table 4.12: Properties of Selected Master Controller

Producer / Model	Arduino / Due
CPU	32 bit ARM, AT91SAM3X8E
Number of I/O pins	54
Clock Speed [MHz]	84

Operating Voltage [V]	3.3

Seven slave controllers are connected to master. Slave controllers are selected from the same product family as Arduino Micro. Slave controllers are used to;

- Control of drive motors
- Encoder tick count and data process for left drive motor
- Encoder tick count and data process for right drive motor
- Data read and process for IR and inductive sensors
- Data read and process for current sensors, used for motors
- Data read and process for current sensors, used for battery block
- Control of user interface hardware (LCD display, switches, etc.)

in ALM. The selected slave controller is given in Figure 4.15.

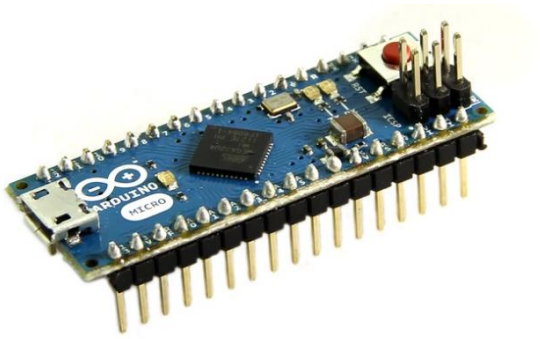


Figure 4.15: Arduino Micro

The properties of selected slave controller is presented in Table 4.13.

Table 4.13: Properties of Selected Slave Controller

Producer / Model	Arduino / Micro
CPU	ATmega32U4
Number of I/O pins	20
Clock Speed [MHz]	16

Operating Voltage [V]	5.0

4.1.1.5 Power Management

In ALM, operating voltage for drive motors and mower motor are consistent with battery block supply voltage which is 24 VDC. Therefore no voltage conversion is needed to power motors.

However, master controller, slave controller and sensors nominally operate in different voltages. Master and slave controllers demands 12 VDC, whereas sensors and some other equipment requires 3.3 or 5 VDC.

Another problem in power distribution is grounding. Since motors drains comparably high currents than other microelectronics, using a common ground for all components, inherently produces noisy communication.

For to overcome this problem and to regulate supply voltage, a 24-12 V DC-DC converter with isolated ground output is decided to be used for separating low current TTL level equipment, from the motors which work with high current values.

Figure 4.16: DC-DC Converter with Isolated Ground

Specifications of selected DC-DC converter is listed in Table 4.14. This converter is used to supply power for main and slave controllers. Electrical circuit scheme is also given in Figure 4.17.

Table 4.14: Properties of Selected DC-DC Converter

Producer / Model	Traco Power / TEN40-1210
Power [W]	40
Input Voltage, Nominal [VDC]	24
Output Voltage Range [VDC]	3.3 - 15.0
Output Max. Current [A]	10.0 - 1.35

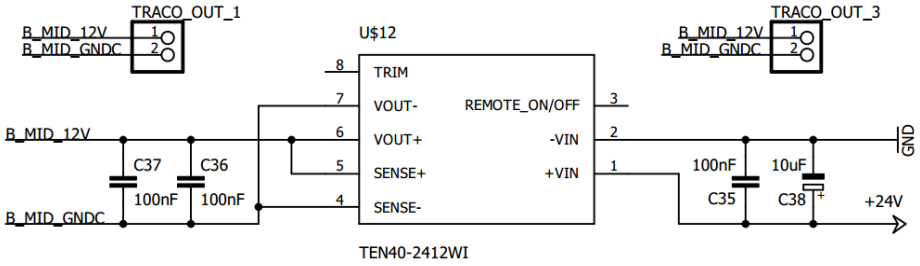


Figure 4.17: Electrical Circuit Scheme for DC-DC Converter

From controllers to sensors, another conversion is needed. Inductive, current and IR sensors operate over 5 V, whereas IMU, GPS and RF communication modules demands 3.3 VDC.

Conversion of 12-5 V and 12-3.3V is performed over another regulator. An adjustable integrated DC-DC step-down regulator is utilized for those conversions.

Figure 4.18: DC-DC Regulator

Table 4.15: Properties of Selected DC-DC Step-Down Regulator

Producer / Model	Texas Instruments / LM2596
Output Voltage Range [VDC]	1.23 - 37.0
Output Current [A]	3.0

Corresponding 12-3.3 and 5 V and 24-12 V regulator circuit schemes are presented in Figure 4.19.

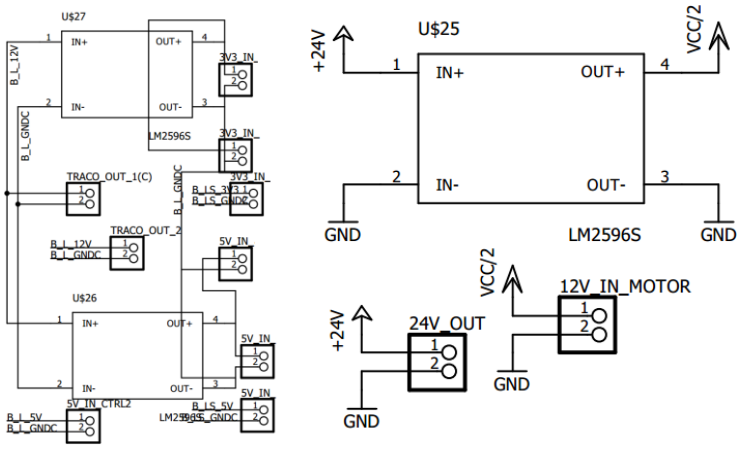


Figure 4.19: Electrical Circuit Schemes for DC-DC Regulator

4.1.2 Mechanical Design

This section presents the final mechanical design of key components and subsystems of the ALM, which is formed after numerous revisions.

4.1.2.1 Overview

Performance, functionality and visual attraction are mainly considered taken during the mechanical design process of the ALM. Robot's bounding box is intended to be kept as minimum as possible without renunciation from operational requirements.

Overview renders of finalized mechanical design are given in Figures 4.20 - 4.22.

Figure 4.20: ALM Isometric View - Front

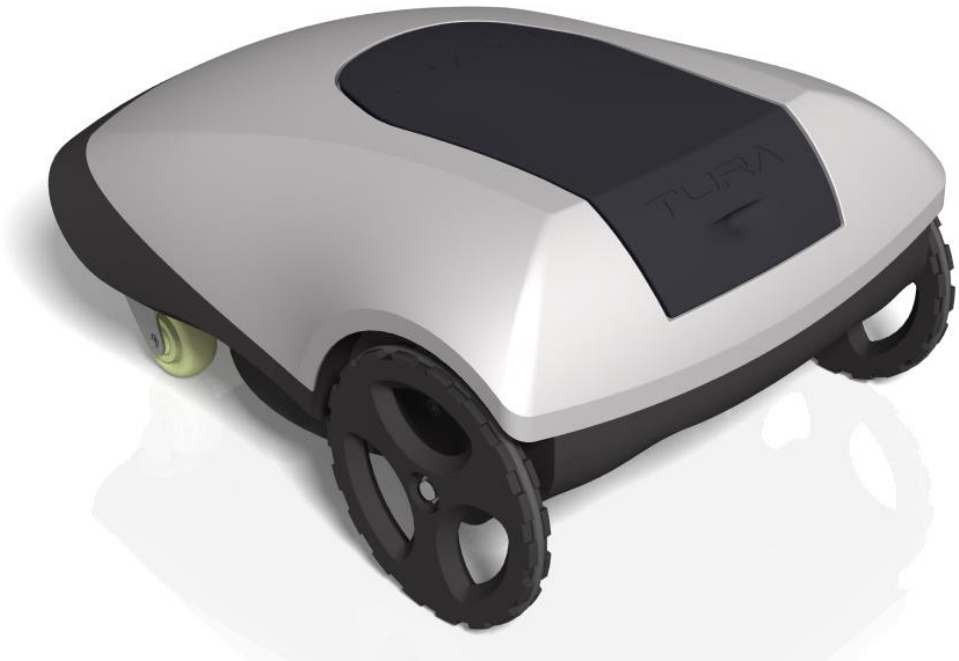


Figure 4.21: ALM Isometric View - Rear

Figure 4.22: ALM Isometric View - Bottom

Compactness is also another important parameter, considered in the mechanical design. The general dimensions of the ALM are presented in Figures 4.23 - 4.25.

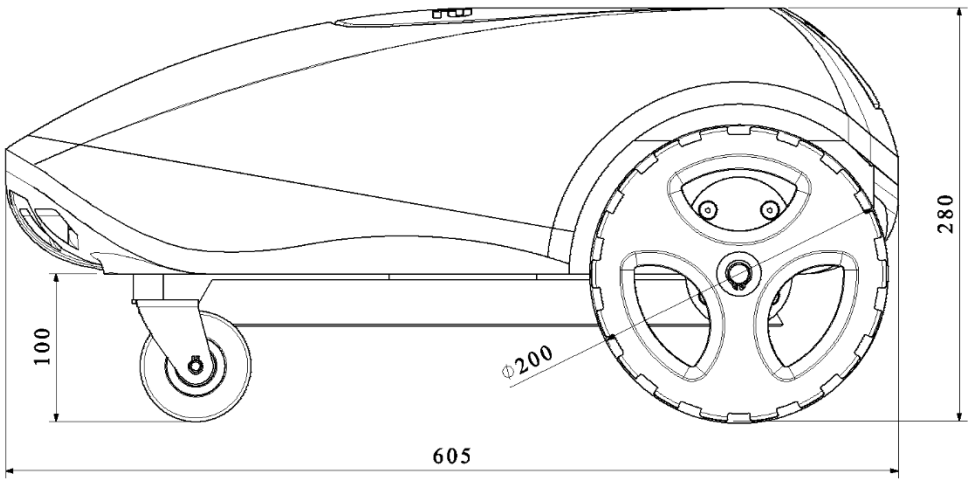


Figure 4.23: General Dimensions of ALM - Side View

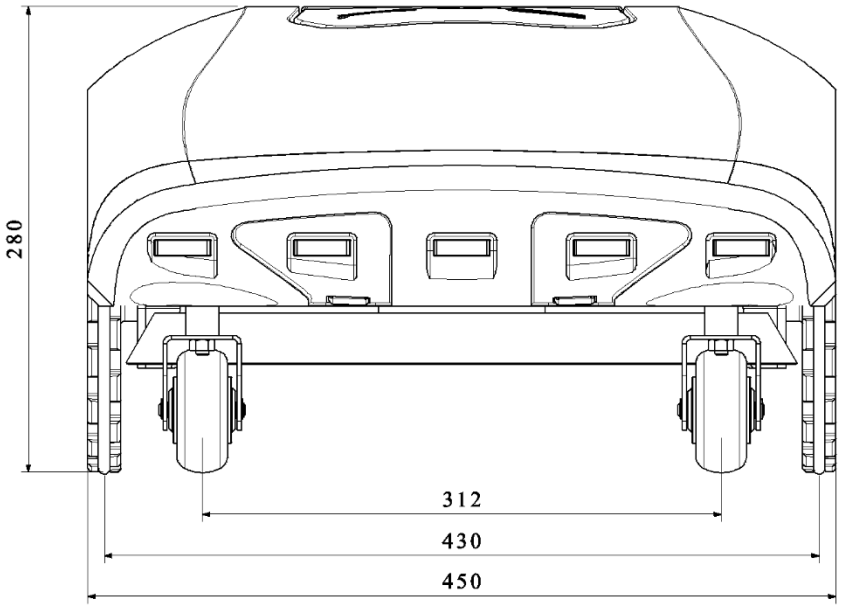


Figure 4.24: General Dimensions of ALM - Front View

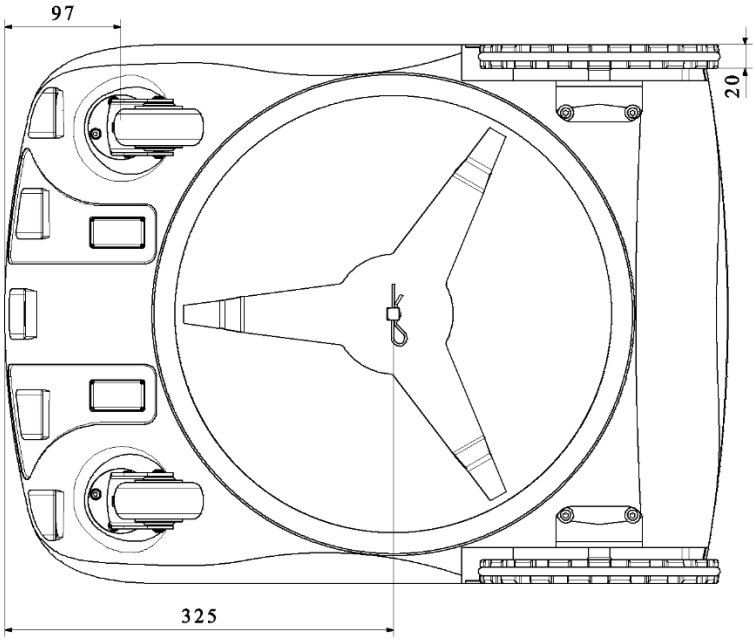


Figure 4.25: General Dimensions of ALM - Bottom View

4.1.2.2 Layout

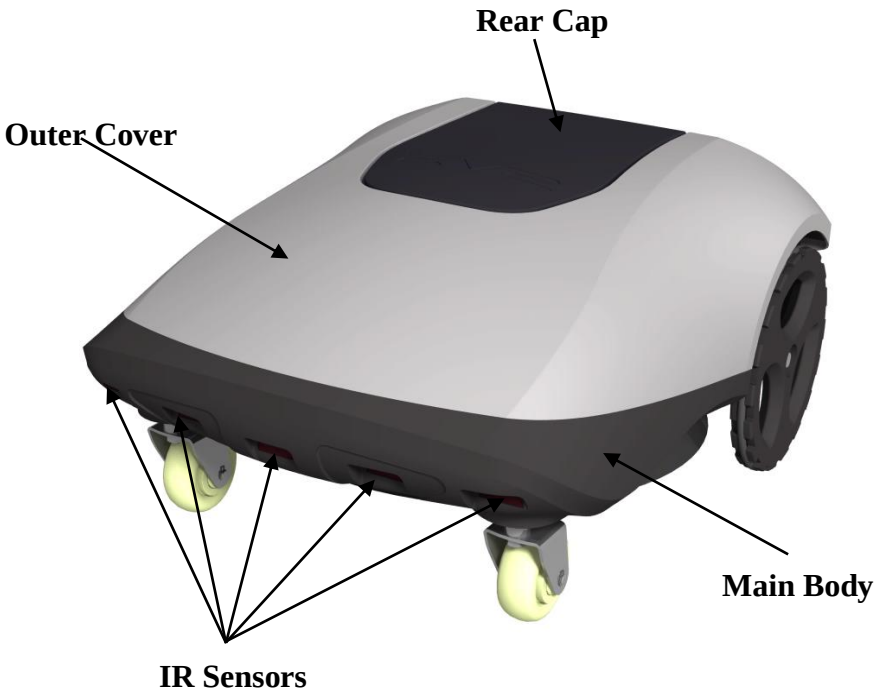


Figure 4.26: External Layout of ALM

The external layout of ALM is designed to be lean and chic to create visual attraction. All of the components lies on the main body, where an outer cover which supplies water and dust protection is assembled on the top of main body. The external layout of ALM is presented in Figure 4.26.

The outer cover is designed to be produced by vacuum forming manufacturing method, whereas the main body is decided to be manufactured by conventional CNC milling operation. For to be lightweight and durable, materials for main body and outer cover are chosen to be Delrin and polyethylene plastic materials.

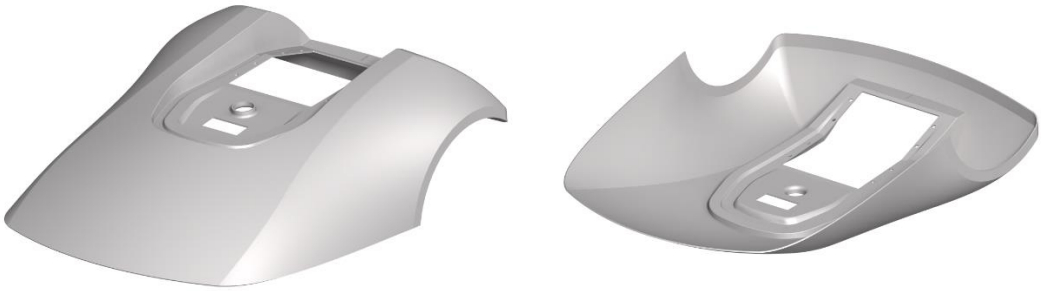


Figure 4.27: Outer Cover of ALM

A rear cap which provides access to user panel, is placed on outer cover. This cap is hinged onto outer cover. Due to its complex geometry, rear cap is designed to be manufactured by 3D printing. The material is selected to be PLA.

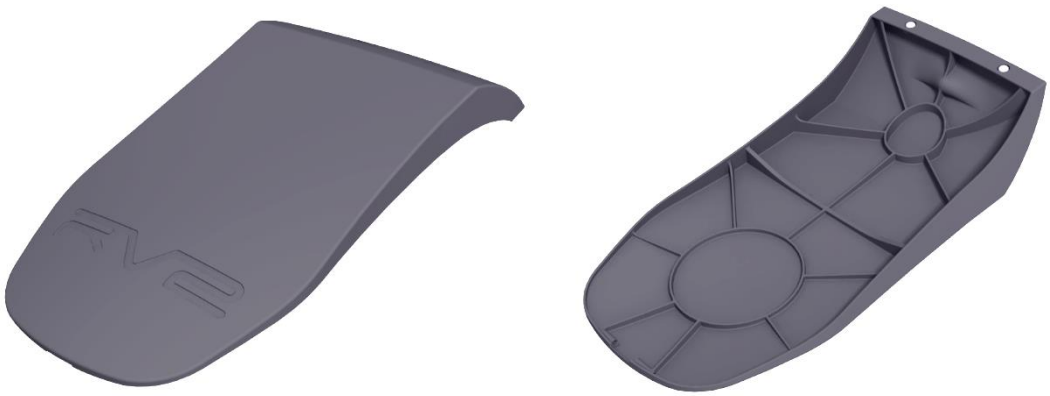


Figure 4.28: Rear Cap of ALM

The opened position of rear cap and the user panel features is given in Figure 4.29. Behind the rear cap, cutting height adjustment roller, battery access cover and user panel takes place.

Figure 4.29: User Panel of ALM

Wheel designs in ALM is presented in Figure 4.30.

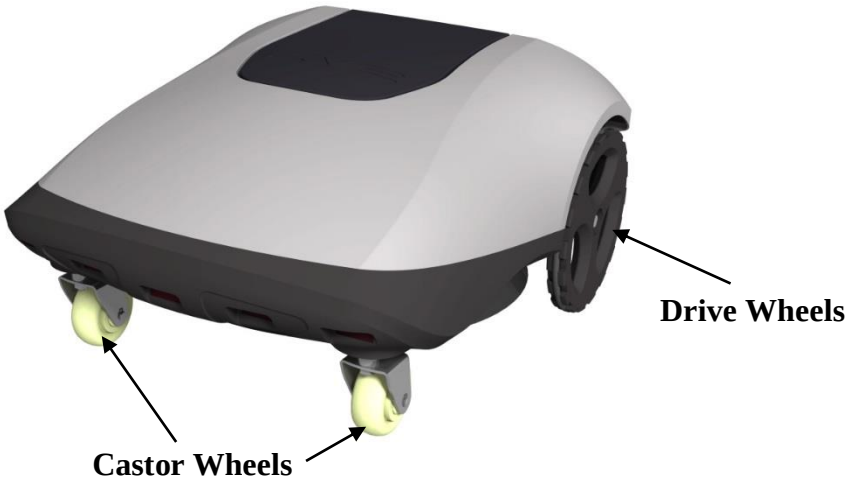


Figure 4.30: Wheel Design of ALM

Two traction wheels and two castor wheels are supported by main body. In order to decrease the effect of slippage, a suspension mechanism is used to castor wheels. This suspension system increases the ground traction of ALM, thereby makes its operation over uneven terrains easier.

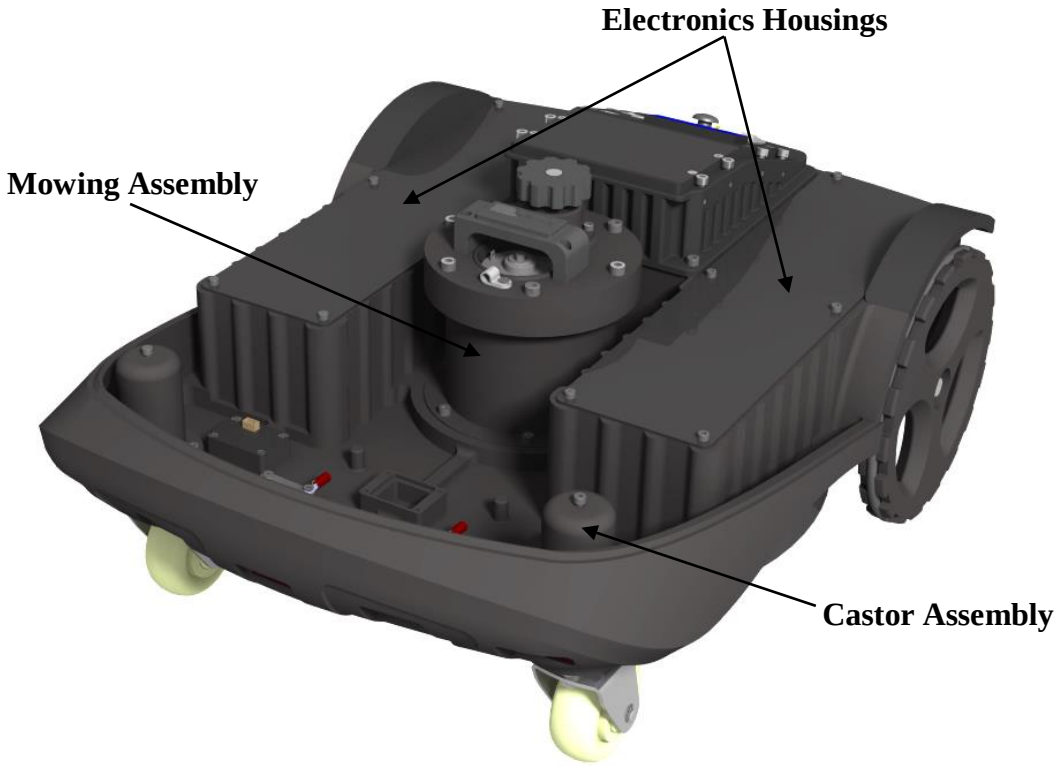


Figure 4.31: Internal Layout of ALM.

The internal layout of ALM is given in Figure 4.31. Mowing assembly consists of mower motor, mower blade and cutting height adjustment mechanism. Suspended castor wheels and electronics housings are other internal mechanical components of ALM.

External charge pins, inductive sensors and IR sensor are also assembled onto main body. All of the interior components are designed to be dust proof for tough outdoor conditions.

User panel of ALM, which is accessed from rear cap, is presented in Figure 4.32.

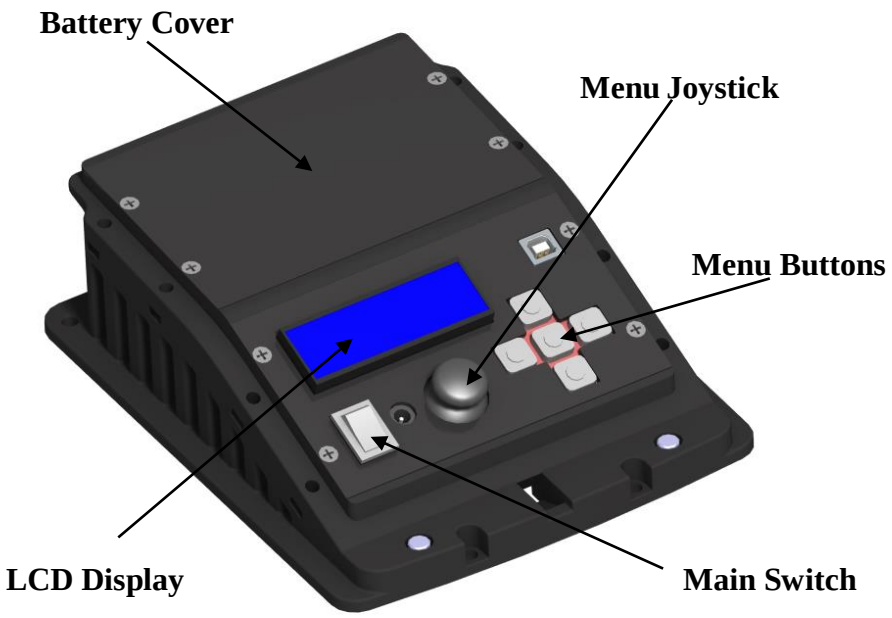


Figure 4.32: User Panel of ALM

The user panel acts as a HMI for ALM. Main switch, menu controls and LCD display are the key components of user panel. ALM is designed to operate with autonomous, RF controlled and Bluetooth controlled modes. User can select those operation modes over user panel.

In the autonomous mode, ALM operations can be set by a timer. User can select periodical operation hours from user panel. Battery voltage and some error information can also be viewed over LCD display.

4.1.2.3 Drive System

Drive systems are important sub-assemblies of ALM. In this design, drive wheels are not directly attached to motor, whereas they are attached to drive system body in which, a 1:2 reduction is implemented in addition to gear-motor reduction rate. Exploded view of drive system are presented in Figures 4.33.

Figure 4.33: External View of Drive System

Rated torque output of a drive motor is 2.0 Nm. The ALM is designed to operate continuously on a 20° slope, therefore an additional 1:2 gear reduction is needed. A belt-pulley mechanism is designed for this purpose. Exploded view of drive system is given in Figure 4.34.

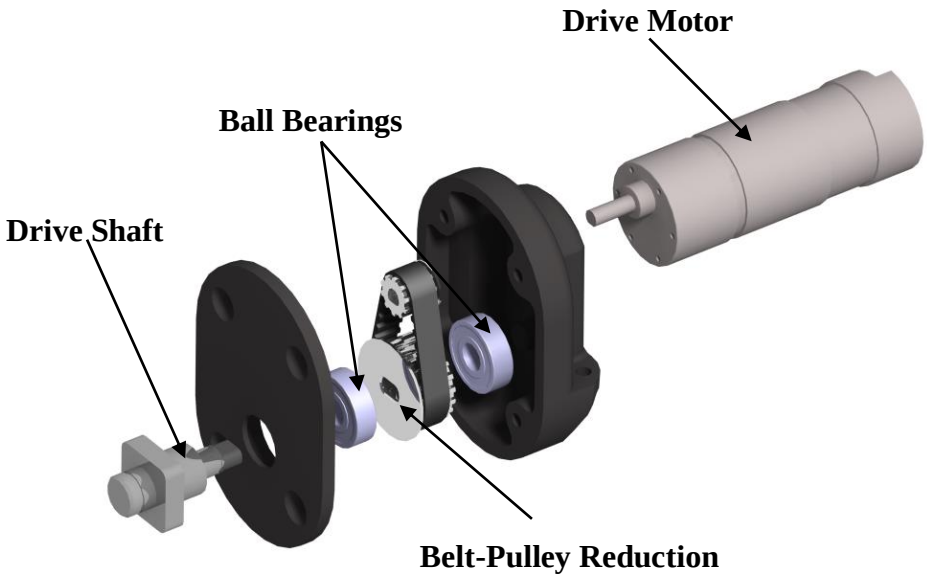


Figure 4.34: Exploded View of Drive System

4.1.2.4 Mowing System

Mowing system lies at the center of ALM. A three-sided cutting blade which is enclosed by a blade cover is attached to the mower shaft. Small cutting edges are designed for to lower shear forces by grass. Exploded view of mowing system sub-assembly is revealed in Figure 4.35.

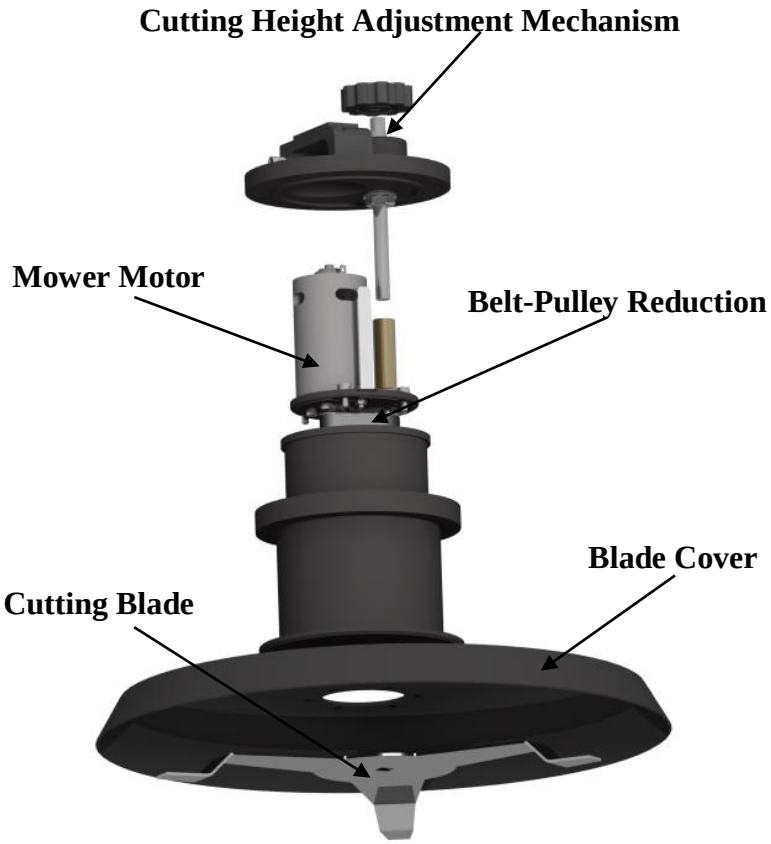


Figure 4.35: Exploded View of Mowing System

Since the torque output of mower motor is comparably insufficient, a 1:1.83 gear reduction is performed to make mowing operation easier. Rated speed of the mower motor is 6,200 RPM. This angular speed is degraded to 3,400 RPM by reduction, which is found satisfactory for a cutting diameter of 350 mm.

Cutting height of the ALM can be adjusted in 3-7 cm bandwidth, measured from ground level. The cutting height adjustment mechanism is very easy to use; the user only has to rotate the knob and the whole mechanism translates with it in vertical axis.

4.1.3 Electrical Design

4.1.3.1 Overview

The main electrical circuit scheme, in which all of the electronic components of ALM is included is presented in Figure 4.36.

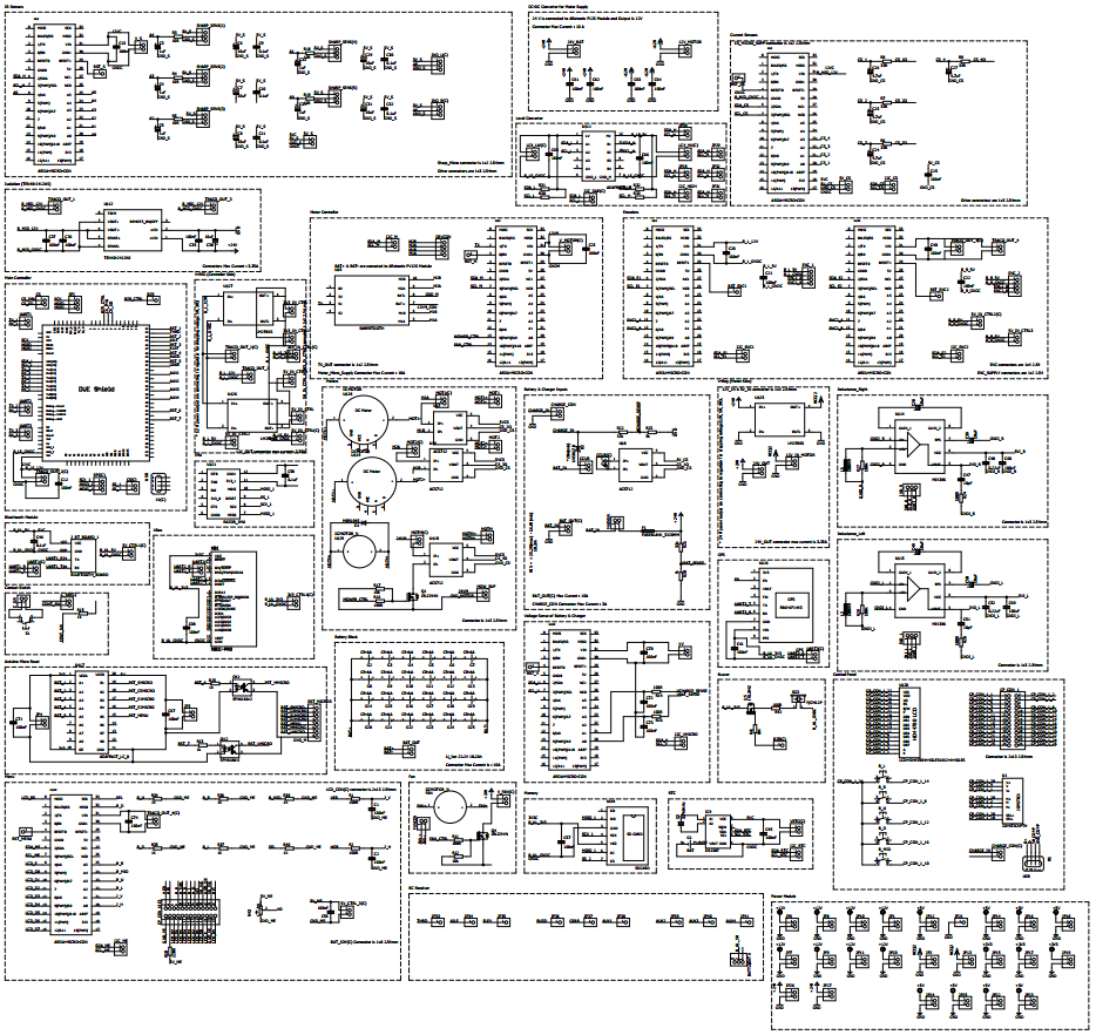


Figure 4.36: Electrical Circuit Schematic of ALM

Components, Connections and Grounding

Electrical consumption of each individual in overall circuit is inherently different. Since motors drain comparably high current than other microelectronics, use of common ground for all components inevitably introduces high level of noise for communications. For overcoming this problem, a DC-DC converter with isolated ground output is decided to be used for separating motors and other equipment inputs. This converter separates whole circuit into two sections because due to electrical isolation.

One of those isolated parts can be called as a "controller" side and the other part is "motor" side. Controller side consists of low power electronic devices such as microcontrollers, encoders, IR sensors, inductive sensors, isolated current sensors and other modules. However, motor side consists of DC Motors, motor driver, battery block and charging circuit. Furthermore, there are extra two microcontrollers in the motor side. One of them is used for communicating with the motor driver. Motor driver supports UART as a communication protocol which is explained in detail below. UART is working with referencing ground. Because of ground isolation, motor driver in the motor side cannot directly connected to the main controller, which is in controller side. Thus, there is an Arduino Micro as a slave microcontroller that communicate with the main controller and motor controller. Since no position control are needed for the mower motor, a simple on/off control is found sufficient. For that purpose, MOSFET is connected to same Arduino Micro to control mower motor. Furthermore, speed of the mower motor can be adjusted by using PWM signals. Other Arduino Micro as a slave is used for controlling battery and charging condition.

Encoder, coupled with the selected gear-motors are decided to be used for odometric calculations. In quadrature encoder, there are two channels which are coded ninety degrees out of phase. For ALM, sensing direction of motors is required. Microcontrollers can determine direction of movement based on the phase relationship between two channels. For avoiding data losses, changing in pulses of two channels should be read by using hardware interrupt in microcontroller. Because of that reason, there is Arduino Micro as a slave microcontroller for each encoder in order to connect

channels of encoders to the main controller directly. In default condition, ground of encoders are separated from grounds of motors. Thus, there is no need for extra isolation.

As mentioned on previous sections, an IMU is used to enhance pose estimation, and a GPS is used as a correction mechanism for large areas. The communication interface of both IMU and GPS is UART. Hence, these low power modules are directly connected to the main controller.

IR sensors are used for obstacle detection. With respect to the distance between object and sensors, analog outputs are generated from IR sensors. By using ADC interrupts in a microcontroller, sensor data are gathered. Also, IR data are filtered by using low pass filter to reduce noise. Moreover, ADC values is sampled twenty times. Arduino Micro is used as a slave microcontroller is used for obtaining and processing IR sensor data. Also, outputs of inductive sensors that are used for detecting the magnetic field inside a perimeter wire connected to ADC channels of this slave microcontroller.

In order to protect drive motors, mower motor and batteries, simple isolated current sensors are used in ALM. Similarly to the IR sensor, low pass filter is used and ADC values is sampled twenty times. Another Arduino Micro slave controller is used for data processing.

There is user panel in ALM. It consists of LCD and buttons. Furthermore, the main switch and charger input is also placed on this panel. User can select operation modes such as RF, RC, Bluetooth and autonomous from this panel. Battery voltage status can be tracked also from that panel. Moreover, system settings such as mower motor status, speed of mower motor, date and time can all be set from this control panel. To read button status, hardware interrupt is needed. An Arduino Micro slave microcontroller is used to overcome these problems. RF, Bluetooth and RC Receiver modules are directly connected to the main controller.

The main controller (or master) is selected as Arduino Due. With the help of slaves, all high level codes such as navigation, trajectory planning and control algorithms run on this master controller.

As a summary, the overall electrical circuit consists of some individual circuits. Controller architecture is designed in hierarchical master-slave configuration. When individual circuits are considered, they must share a common communication protocol for to exchange information. Regarding microcontrollers used in ALM, possible serial communication protocols are UART, SPI and I2C.

Device Communication Protocol

UART is a serial communication protocol that sends parallel data through a serial line. Being different from SPI and I2C, this protocol is asynchronous i.e. there is no clock data transmission. Hence, in UART, devices must agree ahead of time on a data rate. However, in an asynchronous protocol, data rate is generated according to the clock's data. So, the clock rate differences between two devices may cause garbled data. Furthermore, at least one start and stop bit is a part of each frame of data. In other words, 10 bits of transmission time are required for each 8 bits of data sent. Shortly, asynchronous serial ports are only suitable for communication between two devices without an external hardware solely.

SPI is a type of synchronous serial communication protocol that is suitable for short distance communications. Devices that can able to use this protocol communicate in full duplex mode by using master slave architecture. The main disadvantage of SPI is the number of pins required. Connecting a single master to a single slave with an SPI bus requires four lines; each additional slave that connected to the master requires one additional output pin on the master. Huge numbers of pin connections makes undesirable situations where lots of devices must be slaved to one master. Also, the rapid proliferation of pin connections for each device can make routing signals more difficult in tight PCB layout situations.

Regarding those disadvantages of SPI and UART, I2C is found as the most suitable serial communication protocol for the ALM. Similar to SPI, I2C is also intended to allow multiple slaves to communicate with one master in short distance communication. Contrary to SPI, there can be more than one master in I2C protocol. Moreover, I2C protocol only requires two signal wires like asynchronous serial

interfaces such as RS-232 or UART, to exchange information. Figure 4.37 shows the structure of I2C bus protocols for ALM.

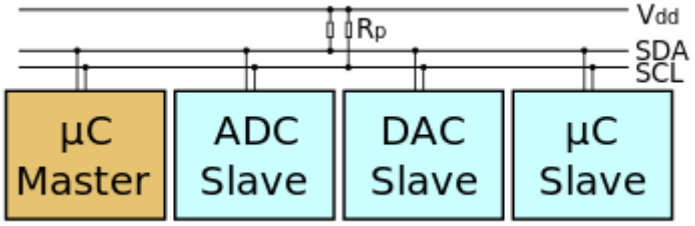


Figure 4.37: I2C Bus Protocol Schematic

I2C bus protocol consist of two wires - SDA and SCL. SCL is the master generated I2C clock that provides synchronization for all data transferred over the I2C bus. However, SDA is the bidirectional data line that transmit or receive data. Both SCL and SDA lines are open drain drivers. In other words, microcontrollers can drive their outputs low, but they cannot drive them high. Because of that reason, pull up resistors are connected to these lines.

4.1.3.2 Power Management and Charging

As it mentioned in previous section, a DC-DC converter with the isolated ground is used in ALM. Besides isolation, this converter provides a regulation of battery voltage to 12 VDC. Therefore, the master microcontroller and slave microcontrollers in controller side are supplied with the 12 VDC output of this DC-DC converter.

Even though microcontrollers can operate over 12VDC, some other modules nominally operates in different voltage levels. These are 3.3 VDC and 5 VDC. So, 12 VDC output of the DC-DC converter is regulated to 3.3 VDC and 5 VDC by using two adjustable DC-DC step down voltage regulator. However, there are two slave microcontrollers which are in the motor side. Thus, by using an additional DC-DC step down voltage regulator, battery voltage is regulated into 12 VDC to supply these slave microcontrollers. Drive motors and mower motor requirements are consistent with battery block supplement which is 24 VDC. Therefore no voltage conversion is needed to power motors.

Arduino Micro is a slave used for controlling battery and charging status. Battery block on ALM is 25.2 VDC at maximum and 16.8 VDC at minimum with the continuous current of 10 A. For overcurrent protection, there is a 10 A fuse at the battery input of the circuit.

To determine battery status (i.e. battery voltage), the 16.8-25.2 VDC interval is mapped between 0-5 VDC, since operating voltage of slave microcontroller is 5 VDC. For this mapping a voltage divider circuit is used. The critical level of battery voltage is defined 18 VDC. After this level, ALM is programmed to find its charging station.

The charger is designed to supply continuous 3 A at 24 VDC to LI-ION batteries. A Schottky diode and a current sensor are used to protect battery circuit.

4.1.3.3 Perimeter Wire

A perimeter wire is used to determine boundaries of mowing area for ALM. The aim of using perimeter wire is creating a magnetic field around a wire by sending some coded signals that ALM can detect when it reaches on this wire. Coded signals refers the polarity changes. In other words, positive and negative voltages reverse each other. The coded signals that generated a magnetic field around a wire is sensed by using two receiver coils. This part can be called as a receiver part.

In receiver part, by using **OP**erational **AMP**lifiers (OPAMP), coil signal is amplified to obtain a meaningful data between 0 to 5 VDC from the ADC channel of Arduino Micro. ADC values obtained from OPAMP is filtered by using a digital filter which is called as “Matched Filter” (also known as optimum filter). In this filter, signals from ADC channel of Arduino Micro are multiplied by the desired signal and summed with the previous correlation result to find new correlation. According to this correlation result of the matched filter, the state of ALM can be determined whether it has reached the border or not.

For the sender part, Arduino Micro is also used as a microcontroller. Regarding characteristics of coils which are used in the receiver part, these coils detect high frequency signals and give a reasonable data while the voltage level on the wire is

around 7 VDC. However Arduino Micro operating voltage is 5 VDC and it generates 20 mA DC current for I/O pins. It is known that magnitude of magnetic field changes directly proportional of current magnitude. So, 20 mA is a low current to generate a magnetic field that can be detected by coils. For amplification, a motor driver that operates over 5-28 VDC is used. A slave microcontroller is also used for direction determination of the output of the motor driver, to provide a coded signal for receiver coils. By changing direction of the motor driver output, positive and negative voltages reverses each other. Thus, the direction of the magnetic field is changed. In addition, a potentiometer is connected to the ADC channel of Arduino Micro to designate the duty cycle of square wave signal. By changing duty cycle, the strength of electromagnetic field generated on perimeter wire can be adjusted manually. Schematic of the sender part is illustrated in Figure 4.38.

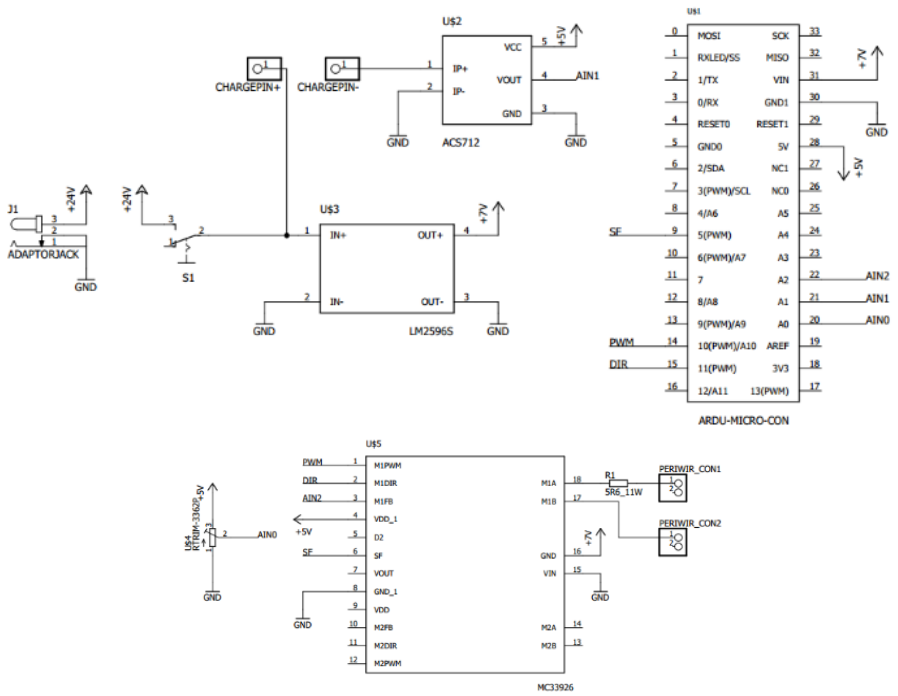


Figure 4.38: Perimeter Wire “Sender” Schematic

For the power management section of the sender part, an adjustable DC-DC step down voltage regulator is used for regulating 24 VDC on the adaptor to 7 VDC to supply motor driver. Also, a load of 5.6 Ω and 11 W resistor is used. By doing this, maximum current on the perimeter wire at 7 VDC is set to 1.25 A, which is determined as a safe current.

4.2 Mathematical Modelling

4.2.1 Introduction

Mathematical modelling is essential for mobile robots for controller design, path planning and even for hardware selection. The behavior of the robot can be modelled by introducing system kinematics or dynamics.

Different mobile robot platforms require different modelling approaches. For example lift and drag forces plays a vital role for a UAV control. For those types of systems, dynamics modelling is inevitable.

Dynamic modelling has some advantages over kinematic modelling, in which, one can determine the motion output more precisely by accounting the forces acting on the physical system. However dynamic modelling introduces complexity for computations. Equations of motion of the system and its solutions become inherently nonlinear. Moreover, dynamic modelling requires great identification of system parameters, such as inputs and material behaviors.

For those reasons, dynamic modelling is found as a good approach for structured environments like laboratories or artificial grounds. Many works in literature showed that dynamic modelling is unnecessary and erroneous for most of the unstructured environments, when parameter identification cannot be performed correctly.

For WMR applications, wheel slippage is the key factor that distinguishes the requirement of kinematic and dynamic modelling approaches. If wheel slippage has a dominant effect on the system behavior, dynamic modelling must be considered. For high speed vehicles like autonomous automobiles, dynamic modelling and nonlinear control are essential. However for the vehicles travelling under 10 km/h speed on a decent tractive ground, kinematic modelling provides very close results to dynamic modelling. For that reason, kinematic mathematical modelling is commonly used for WMR localization [35].

In this work, no dynamic models used and ALM is represented with kinematic equations. Among many others, two main kinematic modelling approaches are used with two traction wheeled mobile robots. These are differential drive and unicycle WMR models. These two methods are similar with minor differences.

In the differential drive WMR model, pose calculations are made from wheels, whereas unicycle model treats robot like a point mass. In this study, differential drive kinematic model is used.

Differential Drive Robot Model

Differential drive robot model and unicycle robot model were compared and their pros and cons were considered from almost all aspects. As a result, differential drive robot model it is chosen for ALM due to its flexibility for creating specific commands and suitability for software architecture. A two wheeled differentially driven WMR model is given in Figure 4.39.

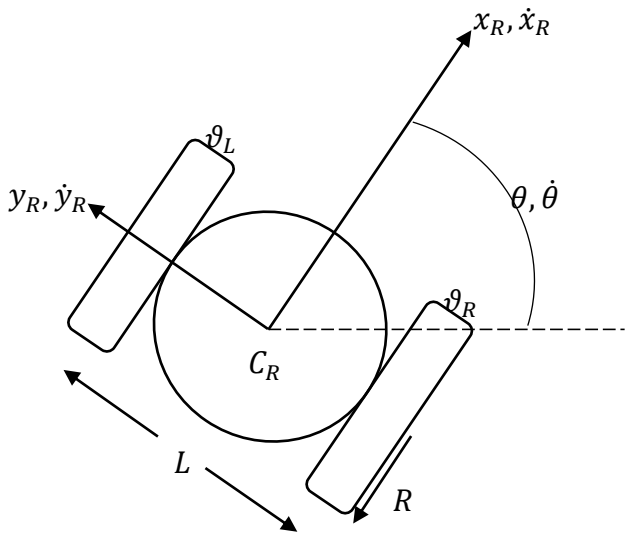


Figure 4.39: Differential Drive WMR Model

Well-known generalized kinematic equations for differential drive WMR model in 2D Cartesian plane;

$$\dot{x}_R = \frac{R}{2}(\vartheta_R + \vartheta_L)cos\theta \tag{4.8}$$

$$\dot{y}_R = \frac{R}{2}(\vartheta_R + \vartheta_L)sin\theta \tag{4.9}$$

$$\dot{\theta} = \frac{R}{L}(\vartheta_R - \vartheta_L) \tag{4.10}$$

where, ϑ_R and ϑ_L are right and left wheel velocities respectively, R is the drive wheel radius and L is the wheelbase.

Following sections gives detailed information for mathematical details of trajectory modelling, path modelling and some heuristics of ALM like obstacle avoidance.

4.2.2 Trajectory Planning

In robot motion planning, a path is defined as the whole route from point A to point B, whereas a trajectory defines some portion of this path. The trajectory may be as simple as a line or it can be different geometrical objects like arc, spline, etc.

In this work, straight lines and zero-radius rotations are used as basic trajectories to create complex paths. Following sections reveals mathematical models of those basic trajectories.

4.2.2.1 Line Trajectory Modelling

For a straight line trajectory, initial position and heading of the robot is known. To check whether robot is following its line trajectory or not one must know the position and the heading error of the robot.

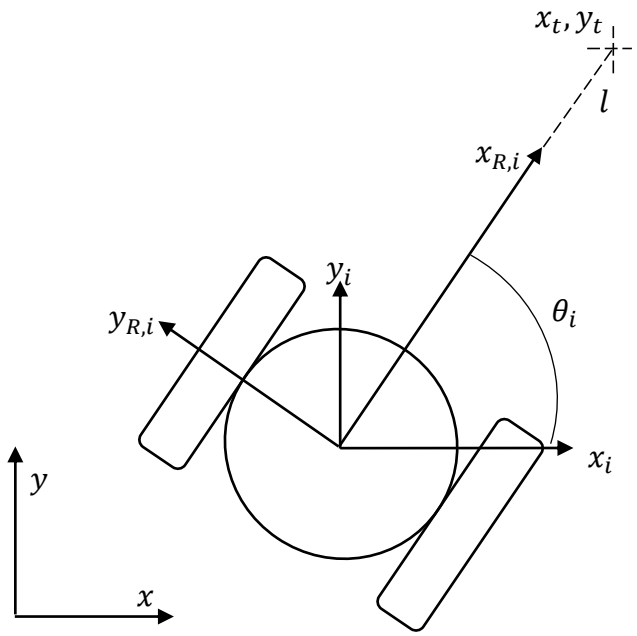


Figure 4.40: Target Point Calculation for Line Trajectory

Target coordinate with respect to robot's inertial frame can be computed with following procedure.

Translate the origin of the global coordinate axes $x - y$ to a new coordinate system called $x_i - y_i$. Then transform $x_i - y_i$ with the angle θ_i about the origin to obtain $x_{R,i} - y_{R,i}$. Now the coordinate system $x_{R,i} - y_{R,i}$ is exactly on the robot body-fixed frame $x_{R,i} - y_{R,i}$, it can be said for the coordinates of target point t, that;

$$\begin{bmatrix} x_{R,i,t} \\ y_{R,i,t} \end{bmatrix} = \begin{bmatrix} l \\ 0 \end{bmatrix} \tag{4.11}$$

Transforming the coordinate system $x_{R,i} - y_{R,i}$ back to the coordinate system $x_i - y_i$, coordinates of the target point t becomes;

$$\begin{bmatrix} x_{i,t} \\ y_{i,t} \end{bmatrix} = \begin{bmatrix} \cos(-\theta_i) & \sin(-\theta_i) \\ -\sin(-\theta_i) & \cos(-\theta_i) \end{bmatrix} \begin{bmatrix} x_{R,i,t} \\ y_{R,i,t} \end{bmatrix} \tag{4.12}$$

Substituting Equation 4.11 into Equation 4.12 leads;

$$\begin{bmatrix} x_{i,t} \\ y_{i,t} \end{bmatrix} = \begin{bmatrix} \cos(-\theta_i) l \\ -\sin(-\theta_i) l \end{bmatrix} \tag{4.13}$$

Translating the coordinate system $x_i - y_i$ back to the global coordinate system $x - y$, coordinates of target point t becomes;

$$\begin{bmatrix} x_t \\ y_t \end{bmatrix} = \begin{bmatrix} x_{i,t} + x_i \\ y_{i,t} + y_i \end{bmatrix} \tag{4.14}$$

Substituting Equation 4.13 into Equation 4.14 gives;

$$\begin{bmatrix} x_t \\ y_t \end{bmatrix} = \begin{bmatrix} \cos(-\theta_i) l + x_i \\ -\sin(-\theta_i) l + y_i \end{bmatrix} \tag{4.15}$$

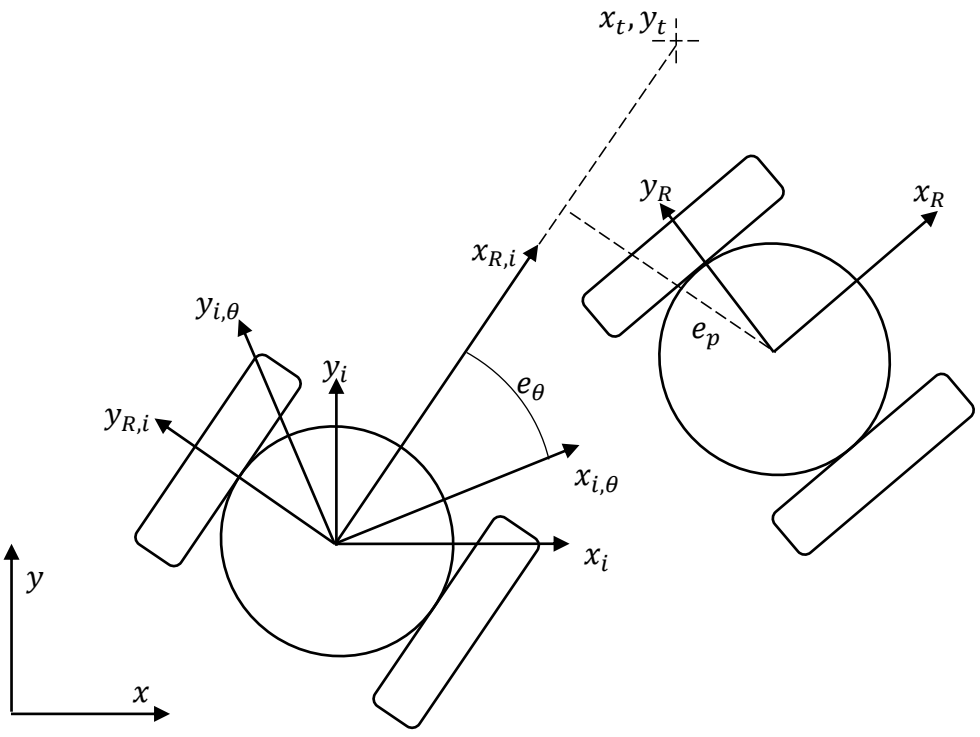


Figure 4.41: Error Representation for Line Trajectory

Error representation schematic for WMR is given in Figure 4.41. Skipping intermediate steps, position error e_p and heading error e_θ for line trajectory can be computed by using Equations 4.16 - 4.17.

$$e_p = -\sin(-\theta_i)\,(x - x_i) + \cos(\theta_i)\,(y - y_i) \tag{4.16}$$

$$e_\theta = atan2\big(-\sin(\theta_i)\,(x_t - x_i) + \cos(\theta_i)\,(y_t - y_i), \cos(\theta_i)\,(x_t - x_i) + \sin(\theta_i)\,(y_t - y_i)\big) \tag{4.17}$$

4.2.2.2 Rotation Trajectory Modelling

It is determined to use zero-radius rotations in the ALM. Mathematical modelling details of rotation action is briefly presented below.

At the end of a zero-radius rotation, target point calculations;

$$\begin{bmatrix} x_t \\ y_t \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \tag{4.18}$$

The position error for the rotation movement is calculated as;

$$e_p = \cos(\theta_i)\,(x_i - x) + \sin(\theta_i)\,(y_i - y) \tag{4.19}$$

4.2.3 Coverage Path Planning

Four CPP models are used in ALM. This section presents mathematical models and algorithmic workflow of those path planning methods in detail.

4.2.3.1 Parallel Swath Pattern Path Planning

Parallel swath pattern, which commonly known as zig-zag pattern utilizes low level line and rotation trajectories to fulfill entire workspace. In this application, parallel swath pattern path planning algorithm is used for rectilinear environments, in which prior border information is assumed to be known.

Each pass in this pattern consist of four consequent elements, i.e. line - rotation - line - rotation trajectories. In this context, lines are parallel to area borders and rotations are orthogonal. Each pass offsets ALM by mowing blade width magnitude. Magnitude of lines are either equal to area height/width or blade width. Schematic of a pass is shown in Figure 4.42.

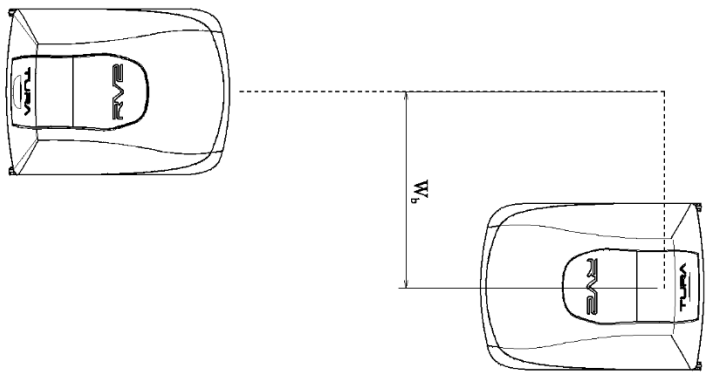


Figure 4.42: A "Pass" in Parallel Swath Pattern

W_b denotes blade width, which is 350 mm for ALM. The number of pass n_p for an area of $W_{area}\ x\ H_{area}$ is calculated by;

$$n_p = \frac{W_{area}}{W_b} - 1 \tag{4.20}$$

W_{area} can be replaced by H_{area} in Equation 4.20 depending on initial orientation of robot (i.e. initial orientation can be parallel to height or width).

The Figure 4.43 represents a parallel swath pattern of 6 passes.

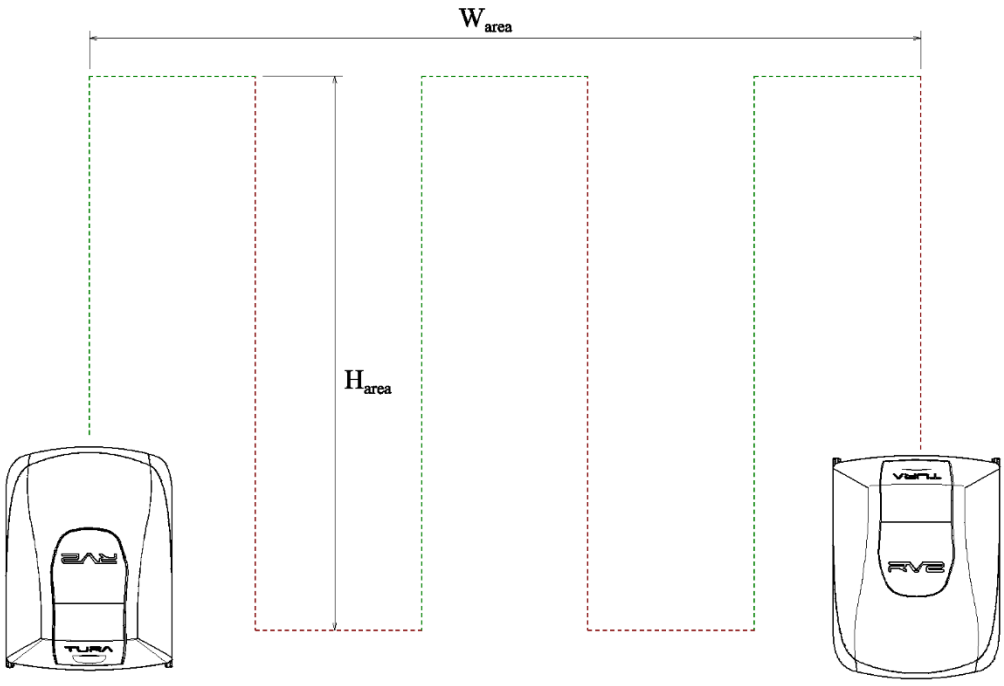


Figure 4.43: Parallel Swath Pattern Schematic

This path planning method is deterministic and it ends after desired number of pass achieved.

ALM has equipped with inductive sensors for border recognition (for ALM not to cross), therefore this algorithm still holds for "L Shaped" or "U Shaped" environments.

4.2.3.2 Spiral Pattern Path Planning

There are two main type of spiral pattern - inward and outward. For the inward spiral pattern, path starts from a corner of working environment and ends on the center line; whereas outward spiral is the opposite. In our application, only inward spiral pattern is concerned, since it not logical to start robot from the center of a garden.

Like other coverage patterns, inward spiral pattern utilizes line and rotation trajectories for “passes”. Union of all passes generates overall pattern and ensures a complete coverage.

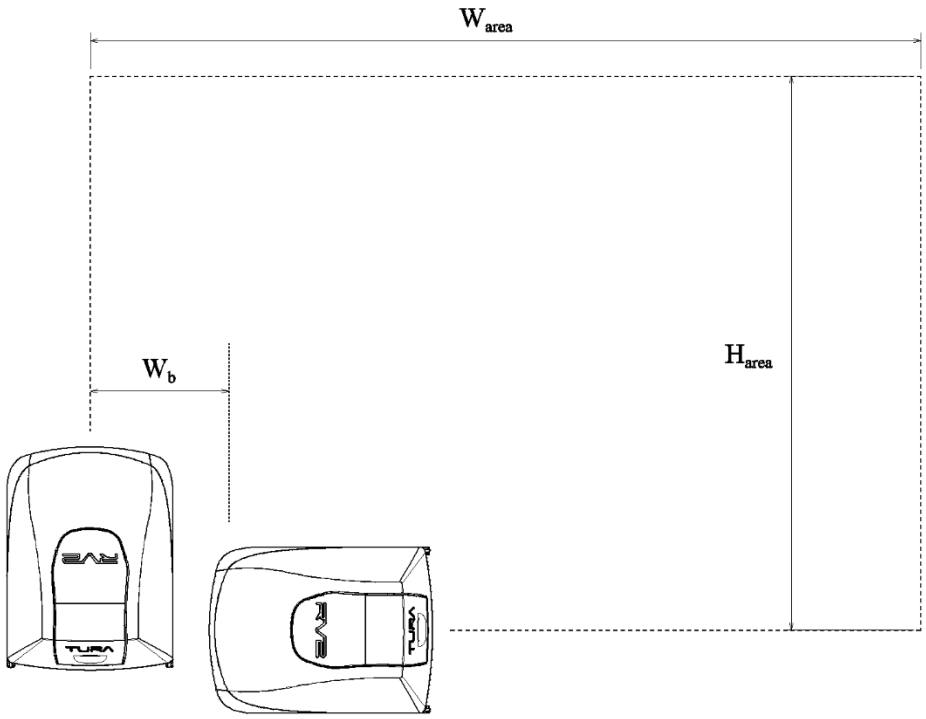


Figure 4.44: A “Pass” in Inward Spiral Pattern

Each pass in inward spiral pattern consist of eight consequent elements, i.e. line - rotation - line - rotation - line - rotation - line - rotation trajectories. The number of pass n_p for inward spiral pattern is calculated by;

$$n_p = min\left[\frac{H_{area} + W_b}{2W_b}, \frac{W_{area}}{2W_b}\right] \tag{4.21}$$

where $n_p \geq 1$.

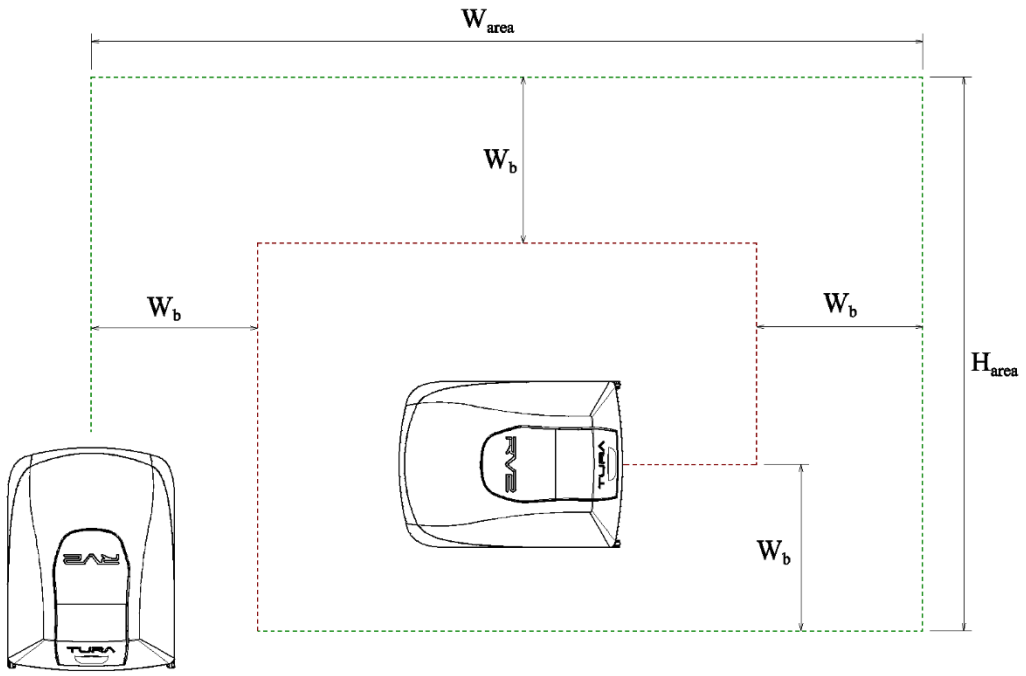


Figure 4.45: Inward Spiral Pattern Schematic

Figure 4.45 shows the inward spiral pattern. In each pass, line trajectories is offset by blade width. There are four lines in a pass, whose magnitudes are formulated in Equations 4.22 - 4.25.

$$Line_1 = H_{area} - W_b(2n_p - 3) \quad (4.22)$$

$$Line_2 = W_{area} - W_b(2n_p - 2) \quad (4.23)$$

$$Line_3 = H_{area} - W_b(2n_p - 2) \quad (4.24)$$

$$Line_4 = W_{area} - W_b(2n_p - 1) \quad (4.25)$$

Similar to parallel swath pattern, inward spiral pattern path ends when number of passes have reached. It is also suitable for varying rectilinear environments.

4.2.3.3 Randomized Path Planning

Randomized path planning algorithms are the mostly used motion planning algorithm in commercial autonomous lawn mowers. It is also used in our design.

Although there are many different variations for randomized path planning algorithms, it is kept as simple as possible in this work to make a decent comparison with other CPP techniques.

In our developed random path planning technique, robot can start from any arbitrary position on the border with any arbitrary heading. Each pass within overall path consists of one consecutive line - rotation trajectories.

With its initial position x, y and heading θ known, line intersection search with borders and heading line (infinite) is performed first. If an intersection points a border other than current border, heading is oriented by randomly created $\Delta\theta$. $\Delta\theta$ is between [0-2π] radians. After that, a straight line motion performed till a border encounter. This operation continues recursively. As randomized path planning is not deterministic, coverage performance and work completion remains unknown during the operation.

The workflow for randomized path planning algorithm is presented in Figure 4.46.

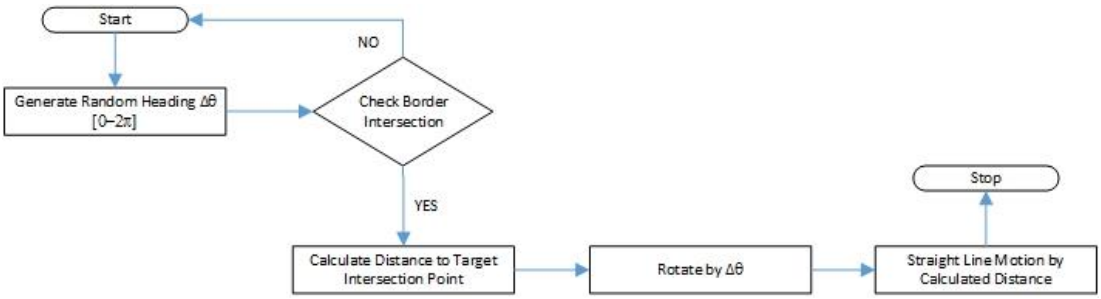


Figure 4.46: Randomized Path Planning Workflow

A sample schematic of randomized path planning is presented in Figure 4.47.

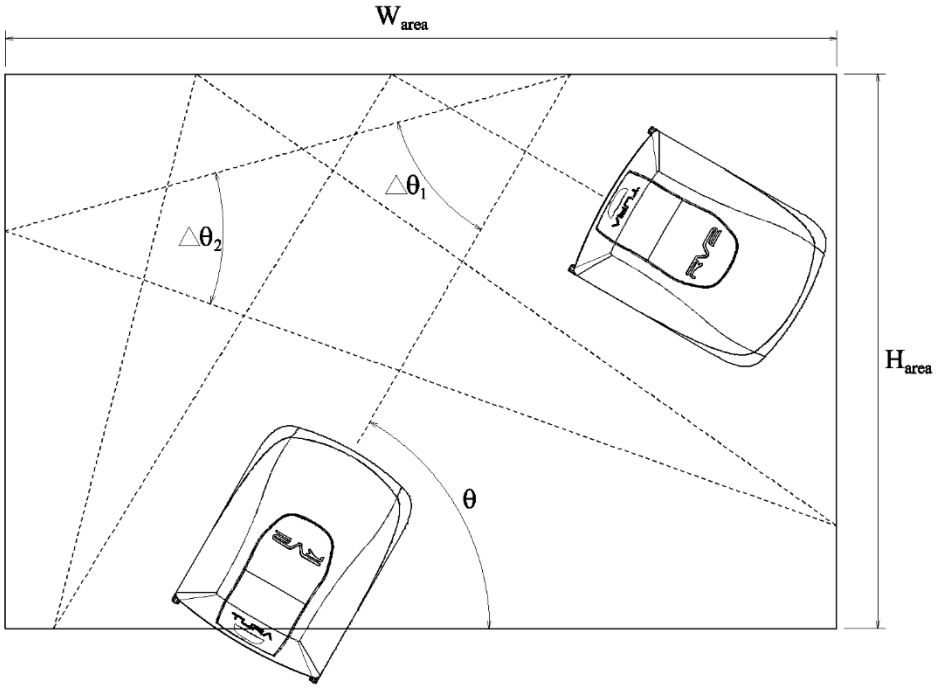


Figure 4.47: Random Pattern Schematic

4.2.3.4 Conditional Coverage Path Planning

CCPP is a deterministic path planning technique, which utilizes prior border (perimeter) and trajectory information for cellular decomposition to find an appropriate trajectory for the next movement.

CCPP works with an iterative fashion. Only straight line trajectories are used in CCPP. Those straight lines crosses the overall working area from border to border. Each trajectory is named as a "pass" and each pass decomposes overall area into small segments which are called cells. In each pass (or iteration), new polygonal cells arise. Cells are not united or deleted in CCPP. As the number of passes increases, number of cells are numerously increases.

In CCPP, robot's path starts from a border in any orientation. The first stage is target determination, in which coordinates of desired target point are calculated. Then heading is oriented through this point and the straight line trajectory is begun to be followed. When current trajectory intersects with previous passes or borders, those

intersections are calculated for further cellular decompositions. The pass ends at the border as mentioned. The main purpose of the algorithm is moving continuously to unmowed regions. At the end of each pass, unmowed cells are identified and new trajectories are calculated for the next pass. Trajectory creations and decompositions continues until there exists no unmowed areas left. Therefore CCPP is considered as a complete CPP technique.

CCPP main workflow steps are defined in Figure 4.48. Further paragraphs contains detailed information about each element.

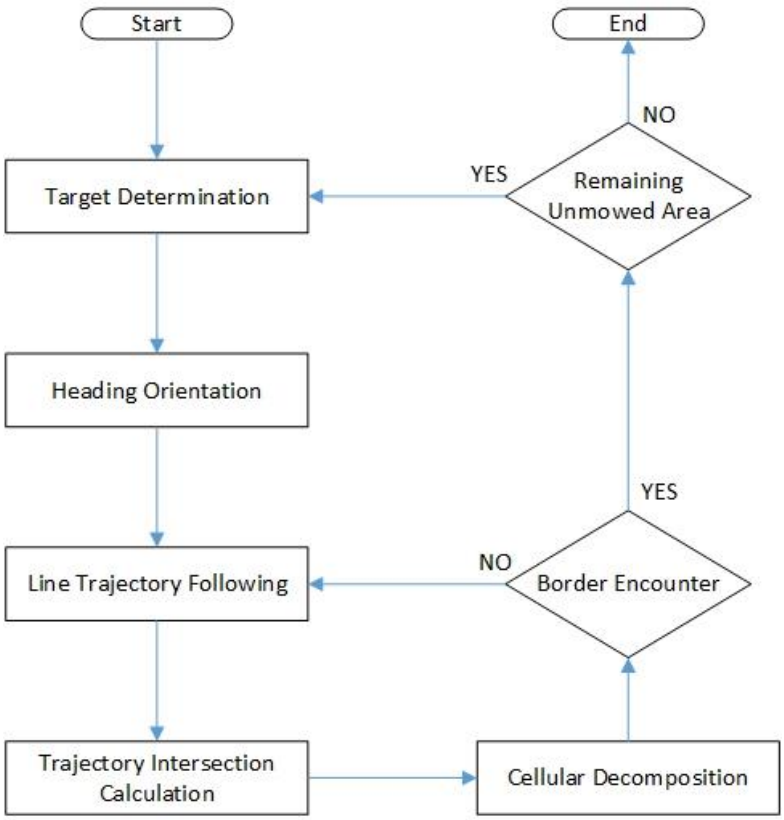


Figure 4.48: CCPP Flowchart

Target Determination

The first step of CCPP is target determination. This determination process is same whether it is an initial calculation or not. Consider a rectangular polygonal working environment, given in Figure 4.49. Robot is located to an arbitrary location on an arbitrary border (Robot must start from a border). Borders may be any kind of

connected geometric loop but they must consist of straight line segments. Regarding that nearly all of the domestic garden geometries are rectilinear polygons, curvature geometries are left out of scope. In this first step, no cellular decompositions made, therefore working area is recognized as a single cell.

Area magnitudes and their geometric centers (GC) are calculated first. For polygonal areas, area magnitude is calculated by;

$$A = \frac{1}{2}\sum_{i=0}^{N-1}(x_i y_{i+1} - x_{i+1} y_i) \tag{4.26}$$

where N denotes the number of vertices and x_i and y_i denotes vertex coordinates. Cartesian coordinates of GC can be calculated from;

$$GC_x = \frac{1}{6A}\sum_{i=0}^{N-1}(x_i + x_{i+1})(x_i y_{i+1} - x_{i+1} y_i) \tag{4.27}$$

$$GC_y = \frac{1}{6A}\sum_{i=0}^{N-1}(y_i + y_{i+1})(x_i y_{i+1} - x_{i+1} y_i) \tag{4.28}$$

For target determination, a "conditional" decision must be made in order to choose the next area to be travelled. The name CCPP is derived from this conditional decision. This condition is controlled by a cost function. The cost function has three main arguments. These are;

- Area magnitudes
- Trajectory length magnitudes
- Number of overpasses

At the beginning of each pass, cost function compares the "cost of travelling" for cells and selects an unmowed cell with minimum cost to pass next. Cost of each unmowed cell is calculated by following relation.

$$Cost_i = \frac{L_{t_i}}{\sqrt{A_i}} + n_{o_i} \frac{\sqrt{A_i}}{L_{t_i}} \tag{4.29}$$

L_{t_i} denotes the length of the trajectory, starting from the robot's current location that passes from GC of unmowed cell and ends on the border. A_i represents area magnitude of unmowed area and n_{o_i} represents the number of overpasses within the trajectory.

The cost function has two mathematical parts. The first term - $\frac{L_{t_i}}{\sqrt{A_i}}$ is the cost of distance. Since it is inefficient for robot to cross entire yard to reach a bigger unmowed cell, it is aimed to select a close unmowed cell by this mathematical term.

The second term - $n_{o_i} \frac{\sqrt{A_i}}{L_{t_i}}$ is the cost of possible overpasses. It is simply aimed to avoid from overpassed regions. Overpass denotes the trajectory-trajectory or trajectory-border intersections in this context. It is aimed to create leading trajectories with least possible overpasses for the next movement. Although overpassing is not a desired action, it is inevitable after some point in overall operation. Therefore it is necessary to distinguish earlier and late overpass avoiding actions.

In earlier passes, the number of overpasses in a pass are either zero or a few and $\frac{\sqrt{A_i}}{L_{t_i}}$ contribution is comparably high. Therefore overpass cost is high. As the number of passes increases, potential overpasses significantly increases. Therefore it is intended to reduce the effect of overpass cost as the number of passes increases. This intend is introduced with the inverse of distance cost. As unmowed cell magnitudes (A_i) getting smaller between passes, effect of overpass diminishes and at some point, so it is not important anymore whether to overpass over previous trajectories or not, since the job completion is soon.

L_{t_i} and A_i calculations are straightforward. For overpass counting, following model is developed. Model is schematically represented in Figure 4.49.

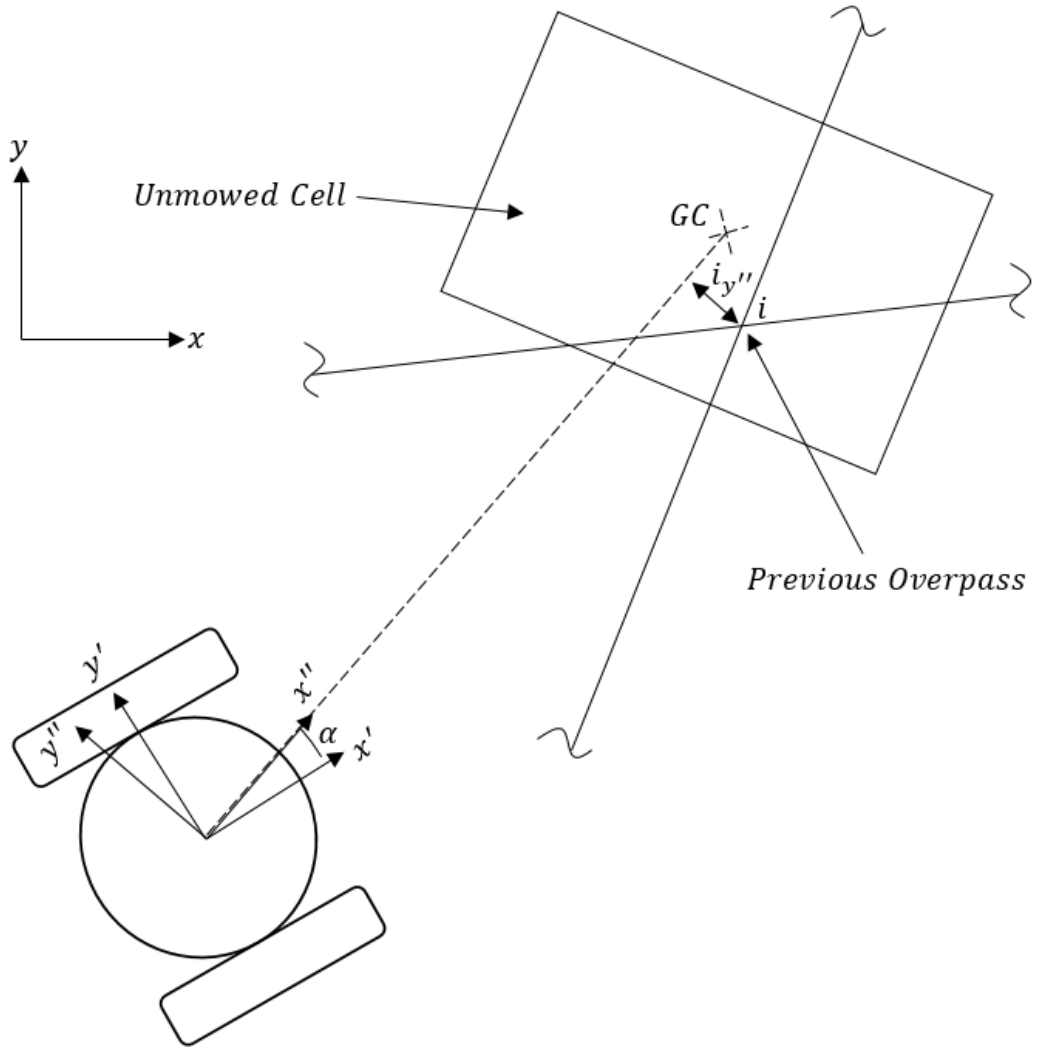


Figure 4.49: Overpass Count Modelling

The main aspect in overpass counting is to find out the number of line trajectory intersections within a stroked pass. With the systematic presented in Section 4.2.2, heading orientation angle α is calculated

$$\alpha = atan2(GC_{y'}, GC_{x'}) \tag{4.30}$$

where $GC_{x'}$ and $GC_{y'}$ are the coordinates of the geometric center of target cell with respect to the $x' - y'$ coordinate system.

$$GC_{x'} = GC_x - B_x$$
$$GC_{y'} = GC_y - B_y \tag{4.31}$$

B_x and B_y are body coordinates in global reference frame. After heading orientation angle α is calculated, body reference frame is rotated along α to obtain $x'' - y''$ coordinate frame. Lastly, the coordinates of an overpass point i on $x'' - y''$ coordinate system can be calculated as;

$$\begin{bmatrix} i_{x''} \\ i_{y''} \end{bmatrix} = \begin{bmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{bmatrix} \begin{bmatrix} i_{x'} \\ i_{y'} \end{bmatrix} \tag{4.32}$$

where,

$$i_{x'} = i_x - B_x$$
$$i_{y'} = i_y - B_y \tag{4.33}$$

Finally, when $i_{y''}$ calculated, it is easy to determine whether this overpass point lies on the stroked pass or not. This is determined simply by;

$$\left| i_{y''} \right| < \frac{W_b}{2} \tag{4.34}$$

condition. If so, this point i is counted as a potential overpass point. W_b denotes the mowing blade width (i.e. stroke). Using L_{t_i}, A_i and n_{o_i} calculations, the overall cost for cell i is computed using Equation 4.29. Cost of each unmowed cell is calculated with this algorithm, then GC of unmowed cell with minimum cost is determined as the target point.

Heading Orientation

Knowing the target point coordinates after the previous step, heading of the robot is adjusted to the selected unmowed cell.

Line Trajectory Following

Straight line trajectory obtained using current position, GC of selected unmowed cell and border. After that, robot starts to follow this new trajectory.

Trajectory Intersection Calculation

Intersections of current trajectory with the previous trajectories and the border are calculated. This information is used in decomposition process.

Cellular Decomposition

When a trajectory intersection calculated, that previous trajectory is split into two lines in geometrical manner. Portion of current trajectory, split lines and connected trajectories are united to form two new cells. Shortly, cellular decompositions are performed by splitting related cells into two. This splitting operation is done by splitting line definitions of both related previous trajectories and the related line segment of the border by using the ongoing line trajectory.

Border Encounter

At the end of each trajectory intersection, the intersection point is checked whether it is on the border or not. If it is not, line trajectory continues with the same heading. If so, the pass is ended and target determination process begins to search for the new target cell to travel.

Work Completion

When a border encounter occurs, the algorithm compares the current largest cell area with the mowing area of the ALM. If unmowed area magnitude is greater, target determination process is executed. If not, it reveals that a complete coverage is achieved.

CCPP algorithm is illustrated and explained step by step below.

Step 1: Working environment is identified by the line segments of the border. In this first step, no cellular decompositions made, therefore working area is known as a single cell. The coordinates of the GC of this polygonal cell and the area magnitudes are calculated. Cost is calculated but since there are no other unmowed cells to compare, GC of single cell is selected as the target passing point. Figure 4.50 represents area border, initial position of ALM and selected cell (with their identities).

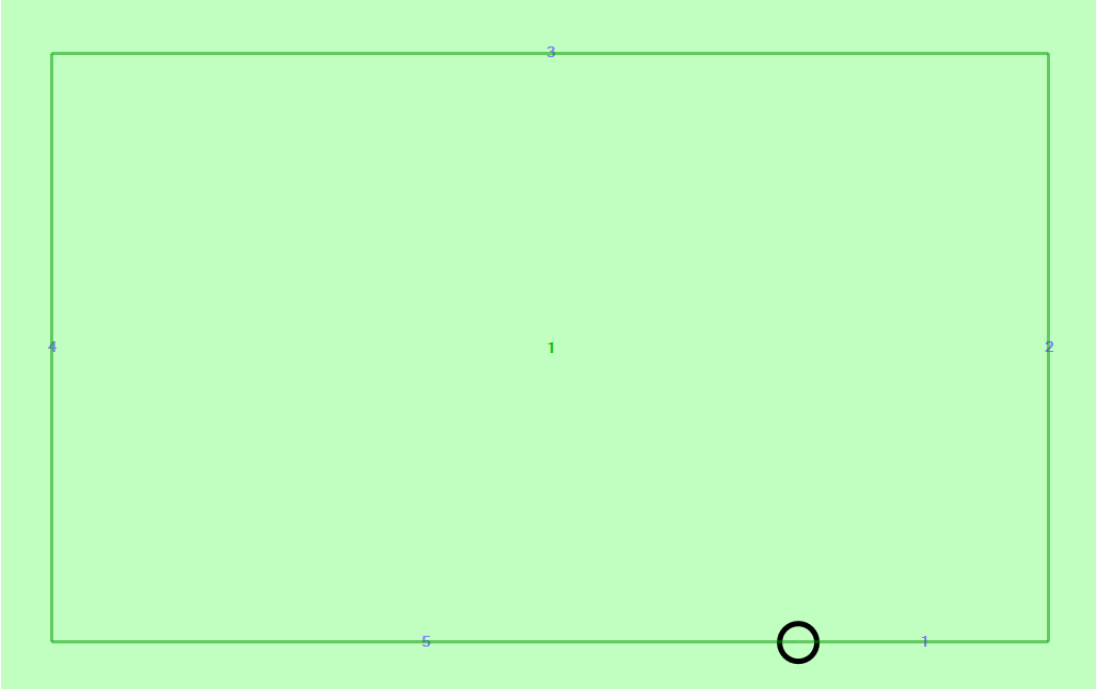


Figure 4.50: CCPP Technique Step 1

Step 2: Heading is adjusted to GC and first straight line trajectory of ALM is generated starting from the initial coordinates, passing through the geometric center of unmowed area and end on a border. By following this trajectory, first pass of ALM is performed and a bandwidth area is assumed to be mowed. This bandwidth is mowing blade diameter of the ALM.

The end of pass is determined after robot reaches to the border. When robot reaches to a line segment of the border, this segment is divided into multiple line segments. After intersection calculations, connectivity of each line is computed, loop search algorithms searches for the newly created loops. At the end of first pass, the working area is decomposed into two new cells, which are not necessarily to be identical. Figure 4.51

visualizes the above descriptions. Newly generated cells, area magnitudes and geometric center coordinates are all computed.

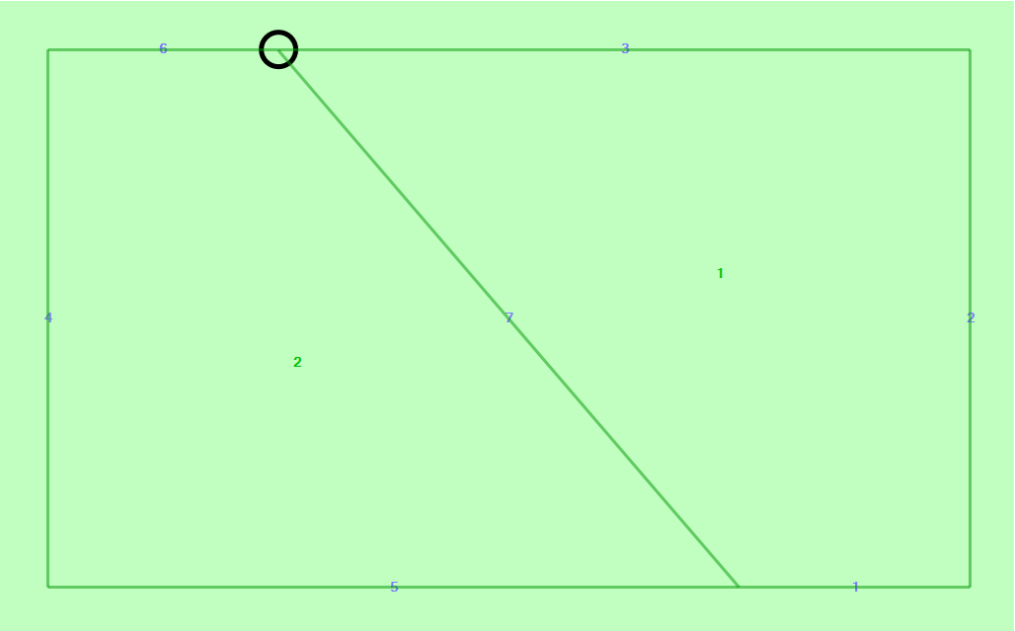


Figure 4.51: CCPP Technique Step 2

As seen, Cell 1 in Figure 4.51 is decomposed into two cells and their identities have renumbered. The mowed trajectory is represented in Figure 4.52.

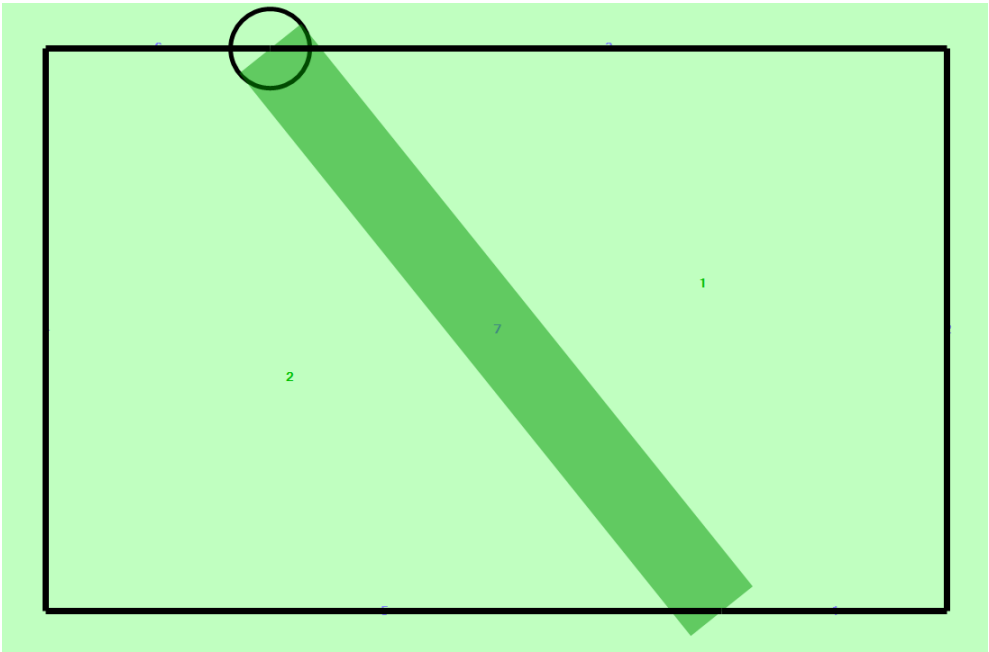


Figure 4.52: CCPP Technique Step 2 (Mowed Trajectory)

Step 3: For a new trajectory generation, a "conditional" decision must be made in order to choose the next cell to be travelled, therefore cost is calculated for both cells according to Equation 4.29. Even if their area magnitudes are equal, cost of travelling through Cell 2 is less than the cost of travelling through Cell 1, since the distance to the border passing through the GC of Cell 2 is closer (Figure 4.52).

As a result, heading is oriented towards GC of Cell 2 and second pass is performed. Figure 4.53 represents the conditional decision making process.

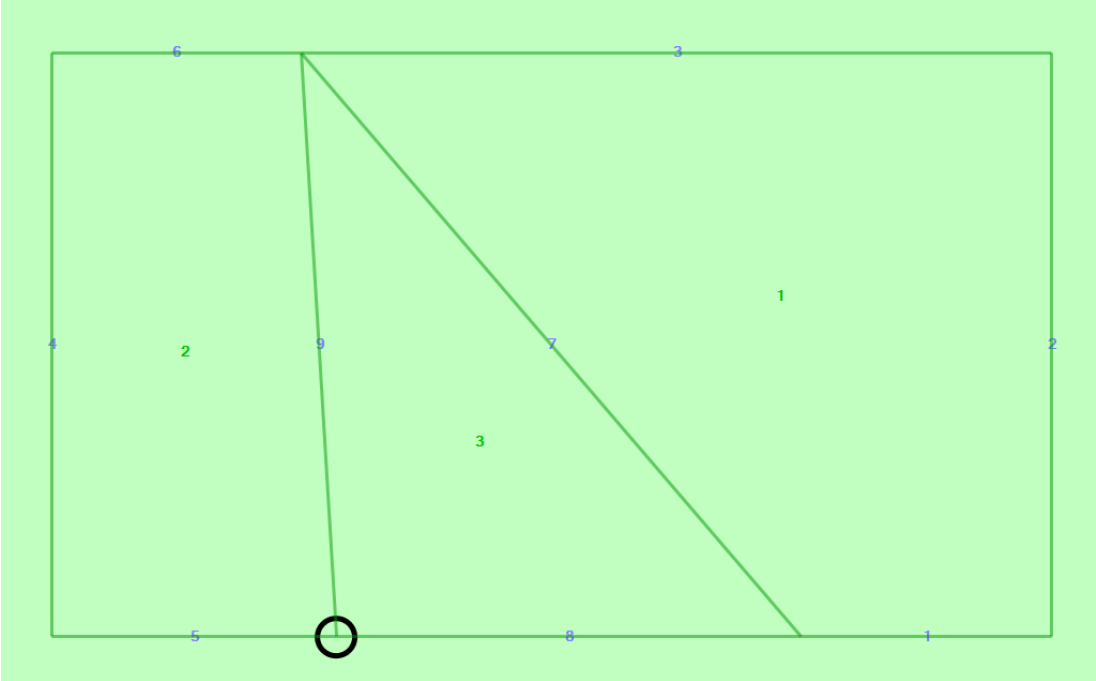


Figure 4.53: CCPP Technique Step 3

Now, Cell 2 is divided into two and two new cells are generated with renumbered identities.

Step 4: Regarding Figure 4.53, although Cell 1 and Cell 3 are larger in size, cost of travelling to Cell 2 is less, therefore third pass is performed over GC of Cell 2. Figure 4.54 illustrates the third pass.

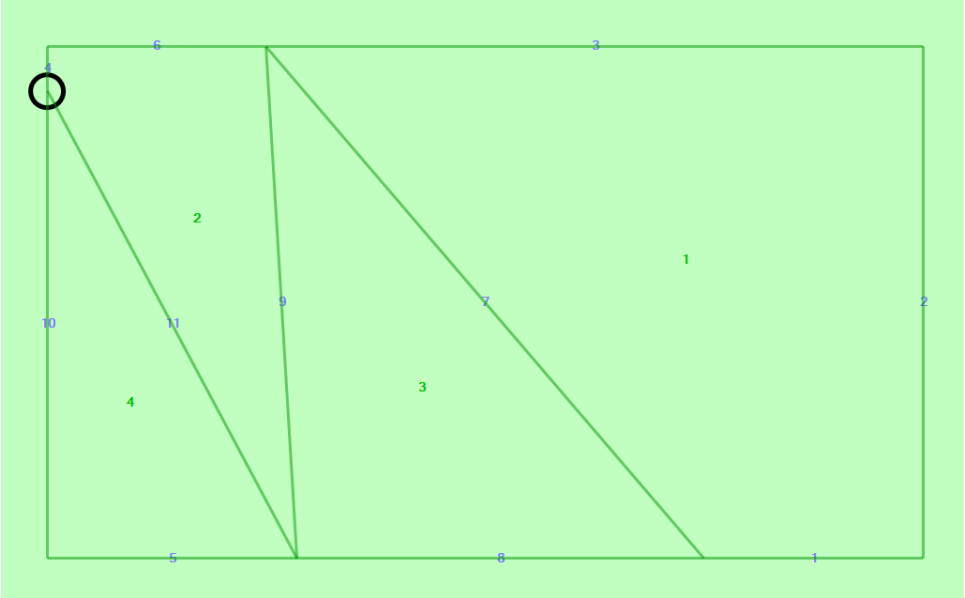


Figure 4.54: CCPP Technique Step 4

Step 5: Area magnitudes of Cell 2 and Cell 4 and their GC distances are relatively smaller than Cell 3 and 1. The number of overpasses with previous trajectories are as follows; 2 for Cell 1, 1 for Cell 2, 1 for Cell 3 and 0 for Cell 4 (Figure 4.54).

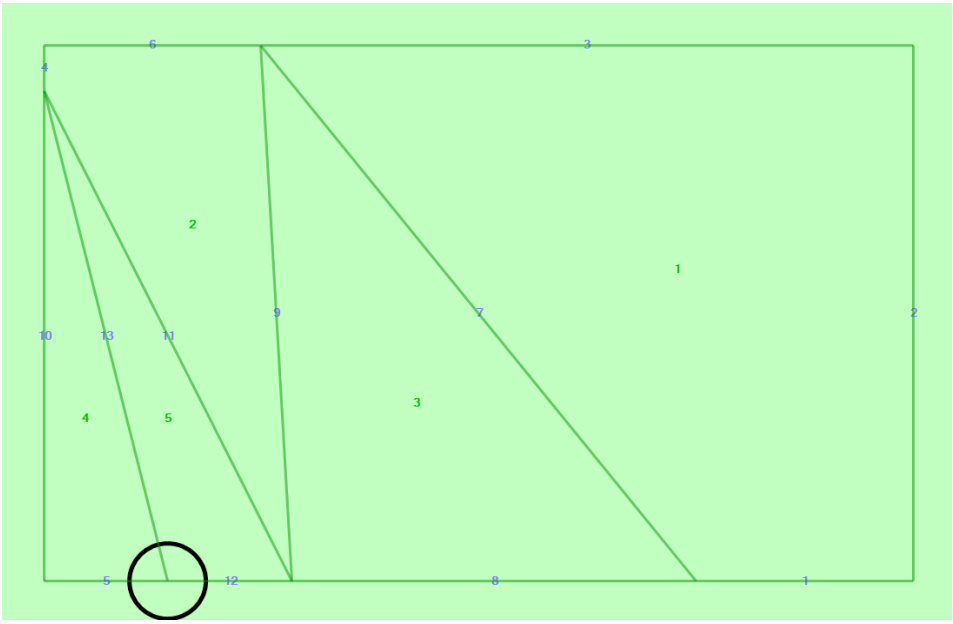


Figure 4.55: CCPP Technique Step 5

Similar to previous step, cost of Cell 4 is calculated as minimum (Figure 4.54) and the fourth pass is made from GC of Cell 4 (Figure 4.55).

Further Steps: As seen in the figures above, CCPP works in an iterative fashion. At the end of each pass, cost for each unmowed cell is computed and a new trajectory is generated. When new decompositions are made, unmowed cell definitions are also updated. When the area magnitude of each single unmowed cell is less than the cutting area of ALM, the operation is assumed to be completed. Figure 4.56 reveals some further steps of operation

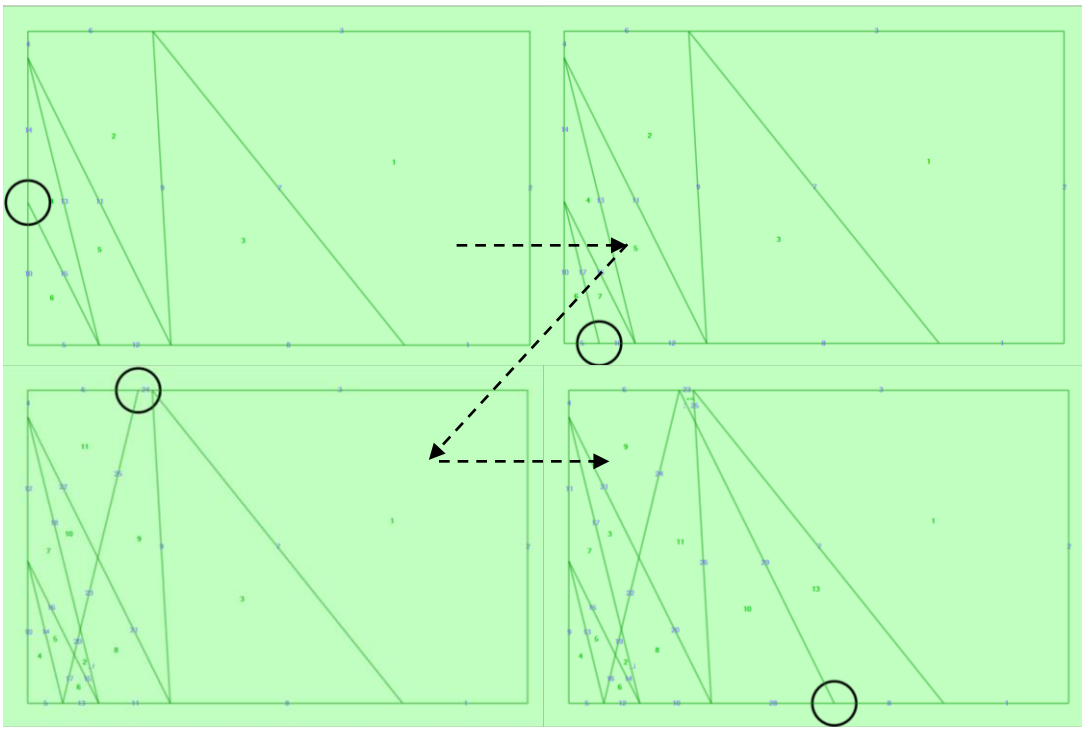


Figure 4.56: CCPP Technique Further Steps

Figure 4.57 represents the overall path for completed operation and Figure 4.59 reveals the mowed area at the end of the operation (with a 99% coverage).

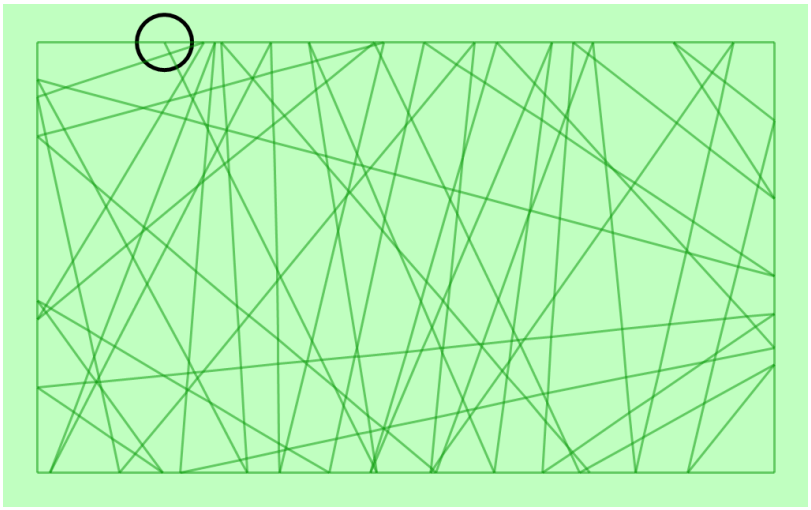


Figure 4.57: CCPP Technique Completed Coverage

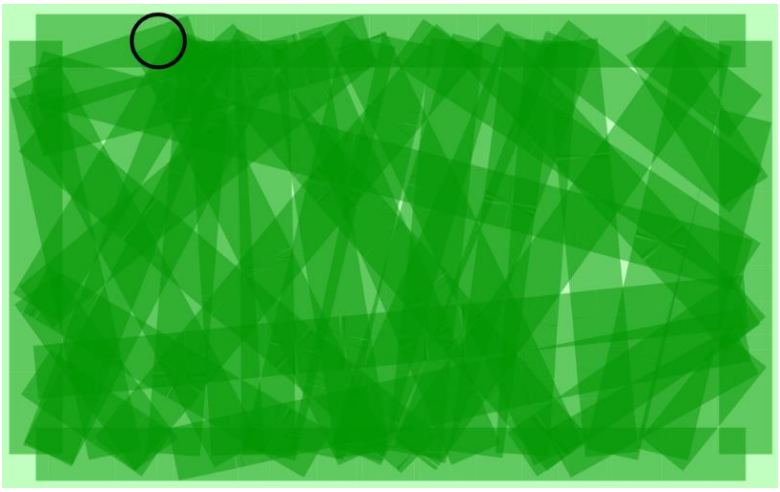


Figure 4.58: CCPP Technique Mowed Area at Completion

CCPP inherently demands more computational power as the working areas getting larger and larger. However CCPP is an iterative operation. It computes cellular decompositions only at border encounters. Therefore those computations do not consume robot's computational power while travelling. Regarding that most of the ALMs working environment is domestic gardens, this technique cannot be directly considered as NP hard. As a summary, CCPP combines the unbounded acts of randomized algorithms and well organized behaviors of deterministic CPP techniques, for a relatively low-cost computational requirement with low-cost hardware.

4.2.4 Error Modelling

One of the main aspects of this work is to compare coverage performances of conventional CPP methods with CCPP technique. Moreover, ideal and actual behaviors of each individual technique are also desired to be investigated.

Actual behavior of CPP methods are tried to be modelled by using an error model. In this context, it is assumed that systematic errors are compensated therefore only non-systematic errors are intended to be modelled.

Before introducing a mathematical error model, physical behavior of ALM is observed. After eliminating systematic errors (a procedure like UMBmark [17] is followed), it is seen that the ALM has a tendency to deviate towards the alignment of the castor wheels, occurred after previous rotation. In other words, if castor wheels are pointing N-W direction after a rotation, the ALM starts its new line trajectory with a CCW heading error. Symmetric behavior is also observed. In this observations maximum heading deviation (from ideal trajectory) is measured as 2 degrees for a straight line trajectory within the physical test environment.

After all, it is concluded that mathematical modelling of non-systematic errors with determinism is nearly impossible. Although numeric calculation errors, sensory data errors, etc. can systematically be modelled, their influence to total deviation is found negligible when compared with influence of environmental conditions, which are inherently random due to terrain or lawn conditions.

In this work, error models are introduced by deviating heading angle of line trajectories with a random angle between ±2 degrees, which is biased regarding previous rotations. After a CCW rotation, the probability of CCW heading deviation is defined as 95% and CW is 5%. For CW rotations, probabilities are reversed. By introducing such kind of biasing, actual behavior of the robot is intended to be reflected. This error model is applied on every trajectory or every pass of each CPP methods and used only in simulation models.

Figure 4.59 represents an overlaid graph of ideal and erroneous paths of ALM for a sample working environment with concave geometry (CCPP method is used.).

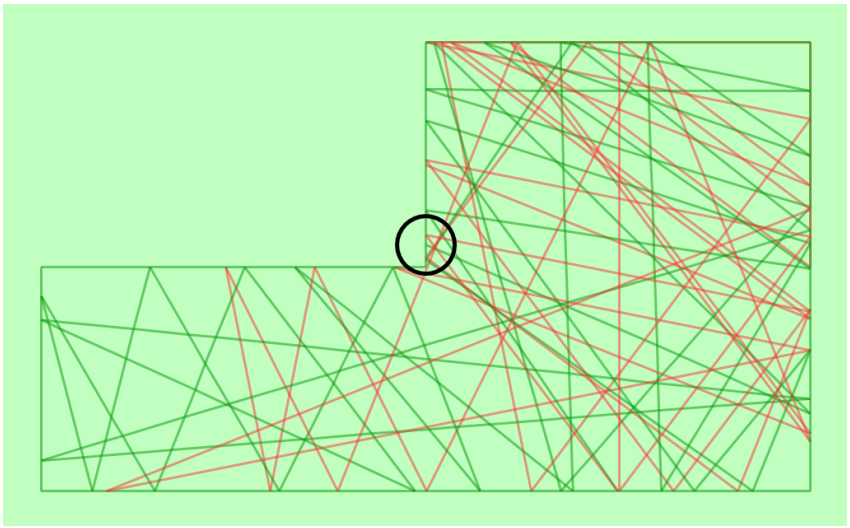


Figure 4.59: Sample Ideal (Green) and Erroneous (Red) Paths of ALM

The coverage of ideal and erroneous paths are represented in Figure 4.60. Erroneous (or actual) coverage is smaller than the ideal behavior as expected.

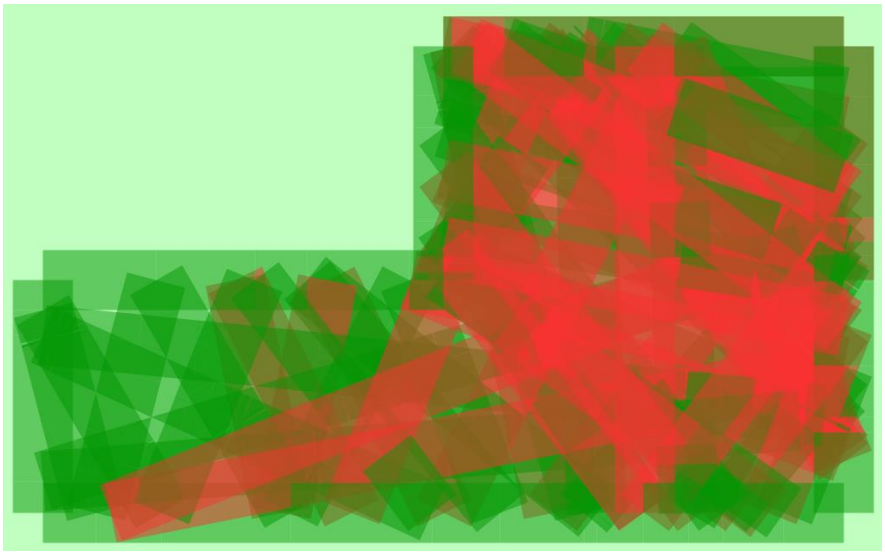


Figure 4.60: Sample Ideal (Green) and Erroneous (Red) Coverage of ALM

Evaluation and validation of developed error model is presented in Section 7.5.

4.2.5 Obstacle Avoidance

Since the ALM has IR sensors only on its front (not on sides), the obstacle avoidance algorithm has been developed in a deterministic manner (i.e. not heuristic). Flowchart of the obstacle avoidance algorithm is presented in Figure 4.61.

The ALM only utilizes straight line and zero-radius rotation trajectories. Since there is no need to avoid obstacles while turning, the algorithm is only active in straight line trajectories for static or dynamic obstacles. Details of developed obstacle algorithm is presented below.

When any of the IR sensors senses an obstacle (critical distance is 15 cm) in a straight line trajectory, obstacle avoidance algorithm starts to flow as a subroutine. First, it determines and stores its initial position and trajectory to follow after avoiding obstacles. In this context, initial position denotes the location of the ALM in the Cartesian plane just before the obstacle encounter. Then, it terminates the running low-level command - which is line command - and simply stops. After that, it checks each five of IR proximity sensors in order to locate the boundary of the obstacle (relative to itself) and to determine avoidance direction.

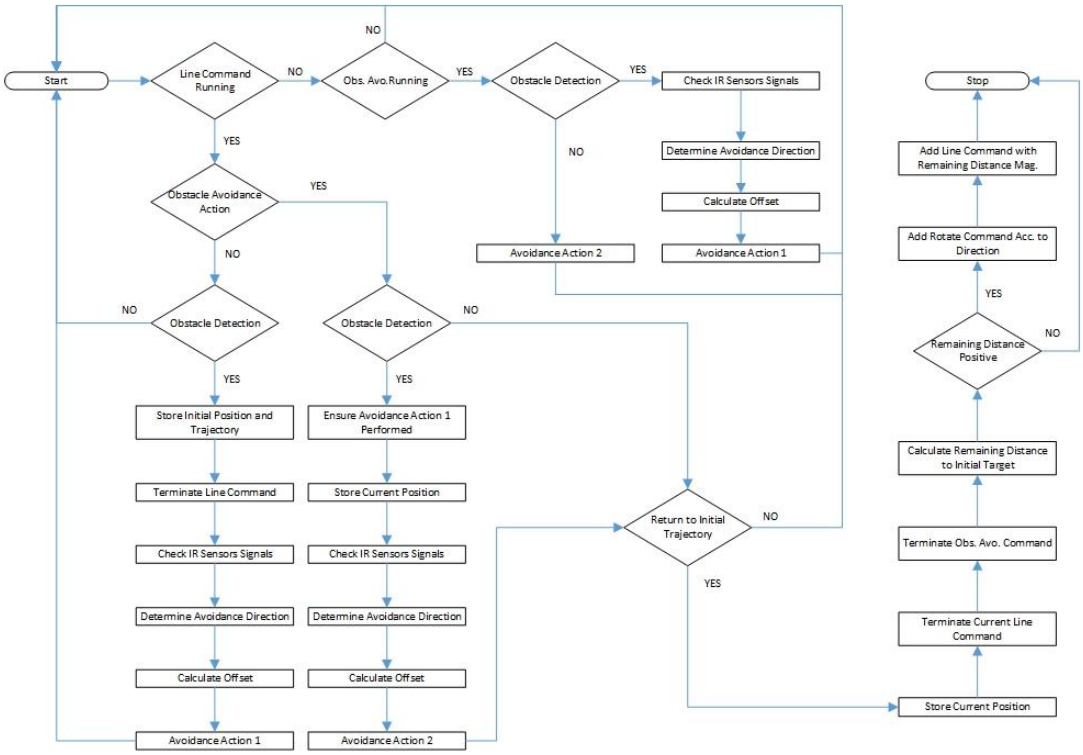


Figure 4.61: Obstacle Avoidance Algorithm Workflow

Avoidance direction determines the sign of the rotation (CW or CCW) to avoid from perceived obstacle. If the robot relatively locates an obstacle on its N-E direction, it will rotate in CCW direction. Vice versa is also true.

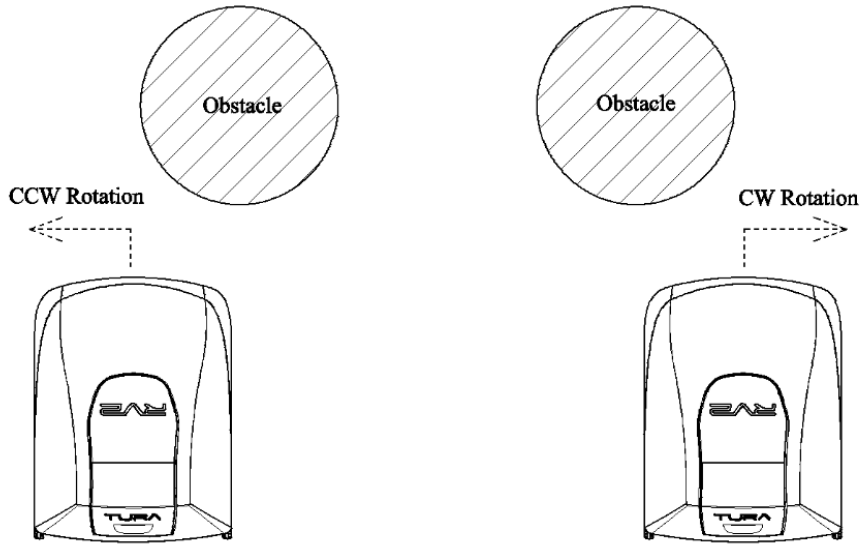


Figure 4.62: Avoidance Direction Determination Schematic

When sign of the rotation determined, the algorithm calculates the offset distance which indicates magnitude of the straight line trajectory to be performed. This offset distance is calculated according to obstacle encountering IR sensors. If only outer IR sensor senses and obstacle, offset will be relatively small, if more IR sensor encounters with an obstacle, this distance will be larger.

After avoidance offset is calculated, an action called "Avoidance Action 1" is performed. This action contains a straight line motion with a 90 degree rotation (opposite sign of first rotation) in advance.

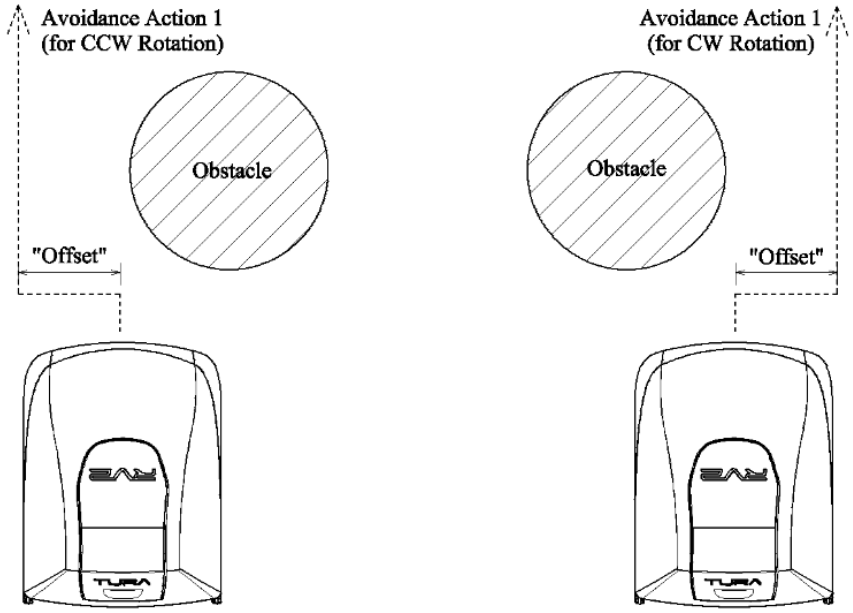


Figure 4.63: Avoidance Offset and Action 1 Schematic

If the obstacle continues to block ALM's way (or it may be dynamic), this avoidance behavior is recursively repeated until it clears its way. Illustration is presented in Figure 4.64.

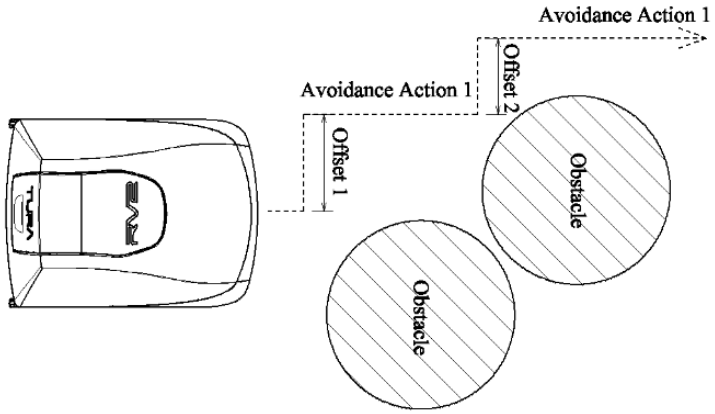


Figure 4.64: Recursive Avoidance Behavior Schematic

When there exists no obstacles on its way, ALM returns to its initial trajectory with an action called "Avoidance Action 2". This action also consists of a rotation and straight line.

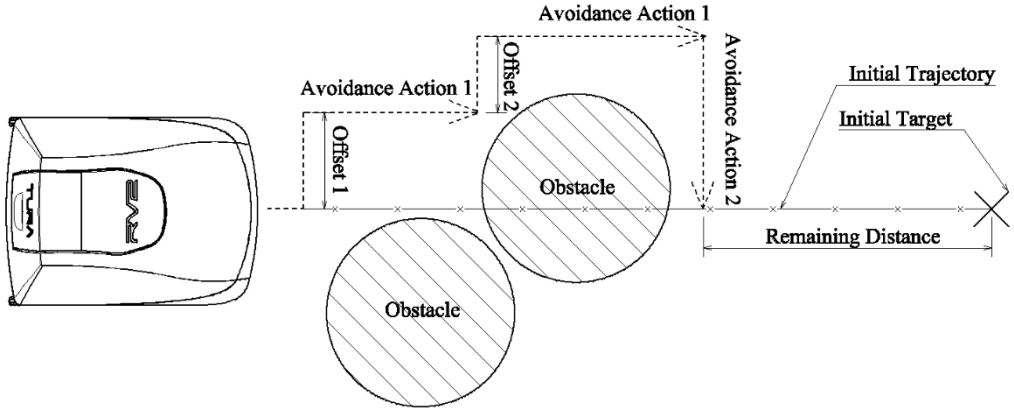


Figure 4.65: Avoidance Action 2 Schematic

Avoidance Action 2 is presented in Figure 4.65. In this action, ALM is forced to draw a straight line, until it coincides with the initial trajectory. After this intersection, intersection coordinates and the remaining distance to initial target point are calculated. ALM performs a final straight line motion to achieve target and obstacle avoidance subroutine ends.

With this uniquely developed obstacle avoidance technique, ALM can avoid multiple obstacles by skidding out or passing in between (when distance between obstacles are relatively large).

4.2.6 Sensor Fusion

As mentioned earlier, localization is the key component for autonomous mobile robot navigation performance. Sensory data like encoder, IMU, GPS, camera, LIDAR, etc. can be fused in order to decrease position estimation error for WMR's [36].

In this work, IMU data is used as a correction mechanism for encoder readings; since encoder data inevitably suffers from wheel slippage. For this purpose, an IMU - encoder data fusion performed with the aid of **K**alman **F**ilter (KF).

KF aims to estimate true states for linear systems by computing a weighted average between predicted and measured states. Weight is distributed according to uncertainty of values. In other words, priority is given to a state which has least uncertainty.

The KF consists of two stages - prediction and update. In the prediction step, the KF produces estimates of the current state variables, along with their uncertainties. In the update stage, measurement data is used to enhance prediction estimations that leads to an updated estimation or simply output.

KF algorithm is recursive. It only utilizes present measurements and previous state information, therefore no additional past information are needed. Workflow of KF is presented in Figure 4.66.

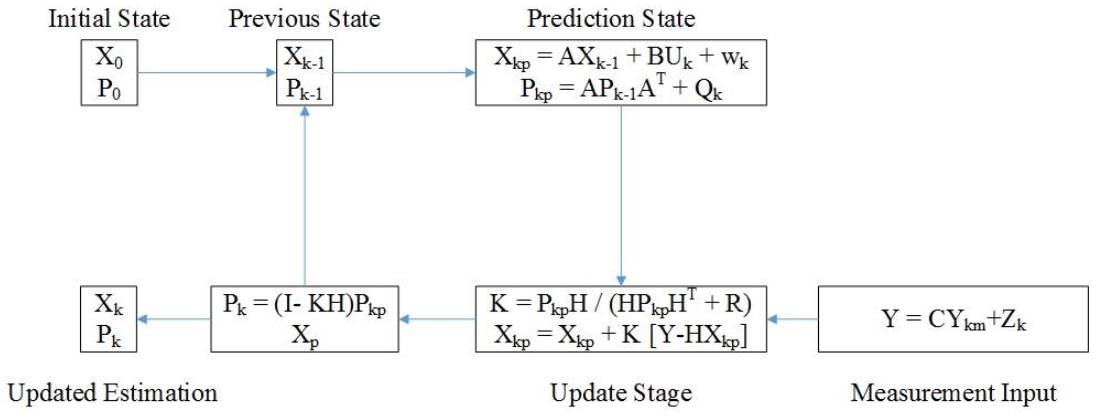


Figure 4.66: Kalman Filter Algorithm Workflow

In above representation, X represents state matrices, P is process covariance matrix, K is Kalman gain, R is sensor noise covariance matrix, U is control variable matrix, w is predicted state noise matrix, Q is process noise covariance matrix, Y is measurement of the state and Z is measurement noise (uncertainty).

State matrix for ALM is chosen as;

$$X = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \tag{4.35}$$

Relation between previous and current state of ALM are represented with Newton's equations of motion as;

$$\begin{aligned} x &= x_0 + \dot{x}dt + \frac{1}{2}\ddot{x}dt^2 \\ \dot{x} &= \dot{x} + \ddot{x}dt \end{aligned} \tag{4.36}$$

$$\begin{aligned} y &= y_0 + \dot{y}dt + \frac{1}{2}\ddot{y}dt^2 \\ \dot{y} &= \dot{y} + \ddot{y}dt \end{aligned} \tag{4.37}$$

$$\begin{aligned} \theta &= \theta_0 + \dot{\theta}dt + \frac{1}{2}\ddot{\theta}dt^2 \\ \dot{\theta} &= \dot{\theta} + \ddot{\theta}dt \end{aligned} \tag{4.38}$$

For prediction stage, Equation 4.39 is used.

$$X_{kp} = AX_{k-1} + BU_k + w_k \tag{4.39}$$

Using Equations 4.36 - 4.38, A, B and U matrices are expressed as follows.

$$U = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \ddot{x} \\ \ddot{y} \\ \ddot{\theta} \end{bmatrix} \tag{4.40}$$

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.41}$$

$$B = \begin{bmatrix} dt & 0 & 0 & \frac{dt^2}{2} & 0 & 0 \\ 0 & dt & 0 & 0 & \frac{dt^2}{2} & 0 \\ 0 & 0 & dt & 0 & 0 & \frac{dt^2}{2} \end{bmatrix} \tag{4.42}$$

All control variables in U are obtained from IMU data, whereas the measurement inputs are obtained from wheel encoder. Thus, calculated position from encoder measurements are improved with the IMU data. Measurement input equation is given in Equation 4.43.

$$Y_k = CY_{km} + Z_k \tag{4.43}$$

In Equation 4.43, current measurement matrix is represented as Y_{km}. It consists of x, y, θ states that are calculated from encoders as mentioned before. For simplicity, noise measurement matrix, Z_k selected as 0. According to Y_{km}, C matrix are obtained as identity.

$$C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.44}$$

In KF, state matrix always estimates the next state with some calculations. During this estimation process, errors could occur because of system uncertainties or noise. Process covariance matrix in Equation 4.35 shows this error in the estimation.

$$P_{kp} = AP_{k-1}A^T + Q_k \tag{4.45}$$

P_{k-1} in Equation 4.45 is;

$$P_{k-1} = \begin{bmatrix} dP_x^2 & dP_x dP_y & dP_x dP_\theta \\ dP_x dP_y & dP_y^2 & dP_y dP_\theta \\ dP_x dP_\theta & dP_y dP_\theta & dP_\theta^2 \end{bmatrix} \tag{4.46}$$

where dP_x, dP_y, dP_θ represents the change in error covariance. This partial derivatives are assumed to be zero (i.e. no change in x error due to y error). In this case P_{k-1} matrix can be simplified as;

$$P_{k-1} = \begin{bmatrix} dP_x^2 & 0 & 0 \\ 0 & dP_y^2 & 0 \\ 0 & 0 & dP_\theta^2 \end{bmatrix} \tag{4.47}$$

As it is mandatory to represent an initial error in KF (otherwise division by zeros created), $dP_x = 0.05$ m, $dP_y = 0.05$ m, $dP_\theta = 5^\circ$ and $Q_k = 0$ are chosen initially. By using the process covariance matrix and errors in the measurement, Kalman gain can be computed as in Equation 4.48.

$$K = \frac{P_{kp}P_{k-1}}{HP_{kp}H^T + R} \tag{4.48}$$

Kalman gain states the weighted factor based on comparing the error in the estimation to the measurement error. Measurement noise matrix R, constituted with the observation errors $\Delta x, \Delta y$ and $\Delta\theta$;

$$R = \begin{bmatrix} \Delta x^2 & 0 & 0 \\ 0 & \Delta y^2 & 0 \\ 0 & 0 & \Delta \theta^2 \end{bmatrix} \tag{4.49}$$

where Δx, Δy and $\Delta \theta$ are also initially set to 0.05 m, 0.05 m and 5° respectively.

According to Equation 4.48, if measurement noise R matrix tends to go to zero, then, Kalman gain K goes to one. In other words, when the error in measurements decreases, the measurement from encoders become more trustable and has more weight. On the contrary, the predicted value become more trustable for greater R.

Lastly, H matrix is used for just adjusting matrix dimensions.

$$P_K = (I - KH)P_{kp} \tag{4.50}$$

$$X_k = X_{kp} + K\left[Y - HX_{kp}\right] \tag{4.51}$$

The last step of KF is calculating the current state X_k, with respect to the Kalman gain, measurement inputs and predicted values in Equation 4.40 and updating process covariance matrix according to Equation 4.51 for the next cycle.

As a short summary, a linear Kalman Filter is implemented to increase pose accuracy of ALM, where IMU data is used for the prediction step and encoder data is taken into account for the measurement.

4.3 Controller Design

A hierarchical controller architecture is developed for ALM. The lowest level controller is wheel speed controller, in which the drive motor speeds are controlled to be run at desired speed values. Besides wheel speed controller, there are also high level position and heading controllers, cascaded with motion planner.

Wheel Speed Controller

Wheel speed controller uses the difference between the desired and the actual speeds of a wheel as the speed error. It is actually an I-controller allowing the robot to accelerate and decelerate quite smoothly. Proportional gain could also be added to this controller, however the main issue here is not trying to reach the desired value immediately or keeping the actual speed at desired value accurately, but just accelerating or decelerating the robot to the desired velocity smoothly. Since the system does not have acceleration and deceleration inputs (functions) particularly, in other words since the speed input has sudden changes, it is not convenient to use the derivative gain for that controller.

Controller block diagram of ALM is illustrated in Figure 4.67.

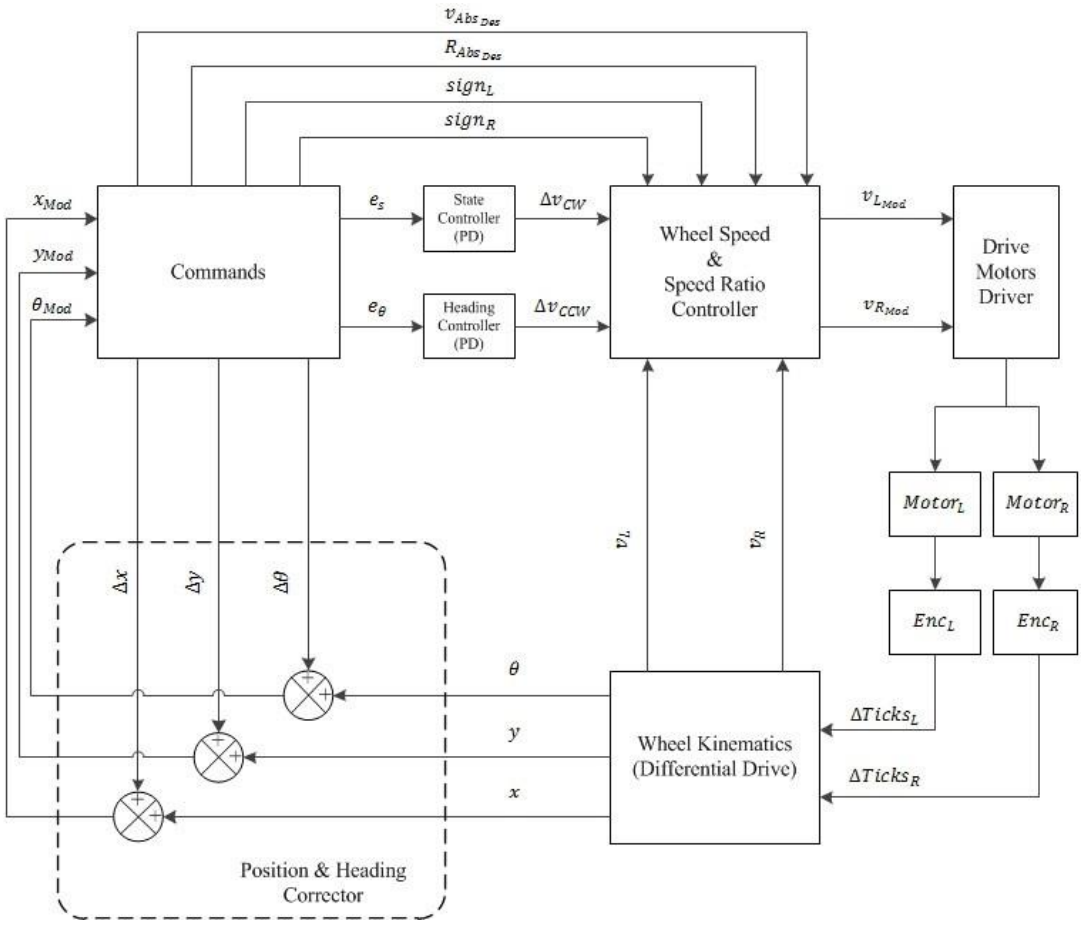


Figure 4.67: Main Block Diagram of the ALM

Position and Heading Angle Controllers

Position and heading angle controllers use the position error and the heading angle error values in order to calculate the increase or decrease in wheel speed. Here, both of them are PD-Controllers. Integral gain is not used here, because the need for the existing integral value obsoletes after the error has zeroed. Position error is zeroed when the robot is reached to the desired trajectory, and heading error is zeroed when the robot heading is reached to the desired heading angle.

As can be seen from Figure 4.68, a positive $\Delta\vartheta$ value indicates a CW rotation in order to steer for the desired trajectory, and a negative $\Delta\vartheta$ value implies a CCW rotation. What the position and heading angle controllers do is to calculate this $\Delta\vartheta$ values on their own way. Afterwards, $\Delta\vartheta$ value came from the heading angle controller is subtracted from the $\Delta\vartheta$ value came from the position controller. This operation gives the resultant $\Delta\vartheta$ so that it can be added to the absolute desired wheel speed in order to steer for a small amount. This modified absolute speed will then be used in the wheel speed controller to be compared with the actual wheel speed for speed error calculations.

In addition to speeding up one wheel, the wheel speeds ratio should be taken into account. Overall system architecture is based on speeding one wheel up while keeping the speed of the other absolutely at the desired speed. Since all the physical dimensions are definite, it can easily be calculated how much the modified ratio should be in order to keep one wheel's speed as is, while speeding the other wheel up. So, what is done here is calculating a $\Delta\vartheta$ value using the controller's output ($\Delta\vartheta$), and adding this $\Delta\vartheta$ value to the desired absolute ratio came from the currently running command. Depending on the system architecture, all those calculations are done by using absolute values, because the directions of the wheels are defining by commands itself.

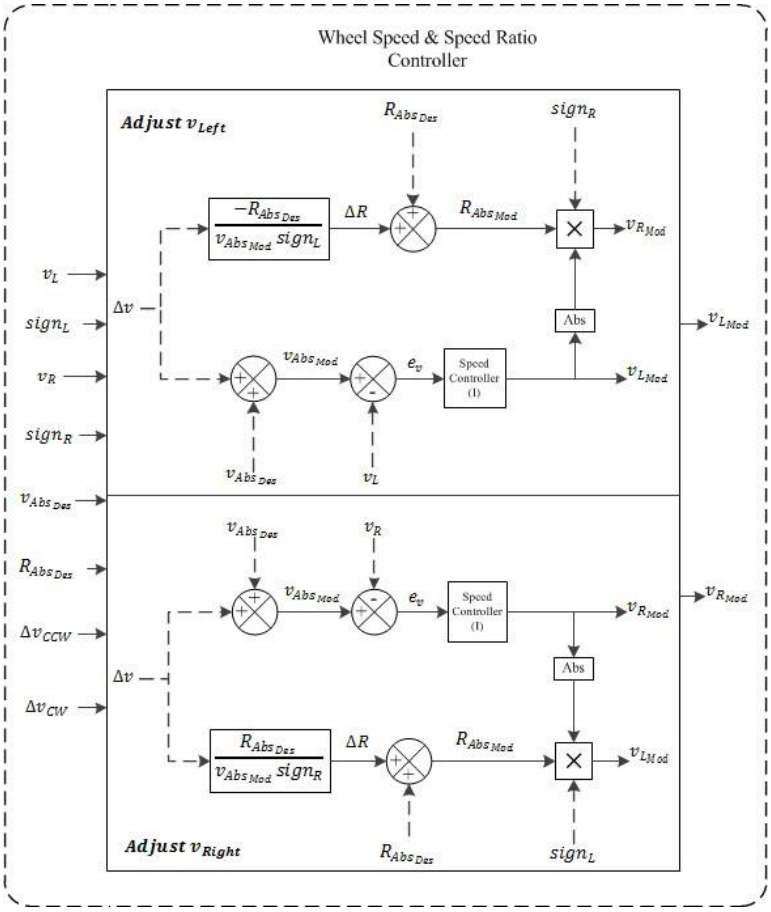


Figure 4.68: Wheel Speed and Speed Ratio Controller

4.4 Software Design

Working environments are inevitably different in terms of shape and size. Additionally, they all may have static and dynamic obstacles which makes the complete environmental identification much harder.

For an area coverage, a coverage path must be planned as mentioned in earlier sections. Each path consists of trajectories and inside, they may have sub-trajectories. Important details of software architecture, operation modes and software commands (for trajectory and path generation) used in ALM, have been presented in this section.

Since it is not in the scope, software architecture details of the developed simulator is not presented in this work.

4.4.1 Operation Modes

ALM is designed as versatile as possible for every kind of usage. There are four different operation modes for the ALM - RF, RC, Bluetooth and Autonomous. Having those capabilities, it is intended to make the end-user to select the way of mowing.

4.4.1.1 RF Mode

RF mode provides a connection between the ALM and computers with an external hardware. This external hardware can be connected to a PC by a USB port. An application on that devices sends a character to RF module which is on the robot by using this external hardware. Also, RF module is connected to the main controller, therefore robot acts according to the character that main controller receive over this RF protocol. Moreover, power switching and speed adjustment of mower motor can be done by using this application.

4.4.1.2 RC Mode

RC mode establishes a connection between the ALM and an RC remote control device. RC remote control device transmits signals to the RC receiver which is placed on the robot. RC receiver module is connected to the main controller, so it operates with the same fashion as RF has. Power switching and mower motor speed adjustments can also be performed over a remote control.

4.4.1.3 Bluetooth Mode

For operating ALM over mobile devices such as mobile phones, tablets and computers, a Bluetooth receiver has also been placed on ALM. With a readily available open source software, ALM can be driven with the aid of corresponding Bluetooth communication protocol.

4.4.1.4 Autonomous Mode

For trajectory generation, two primitive software objects: line and rotate commands are used. These are low-level trajectory generation commands. With the aid of those commands, any lawn area can theoretically be covered.

Although these commands are considered to be enough, dealing with this type of low-level commands is definitely not reasonable for area coverage. For an area coverage, a coverage path planning method is essential. Therefore, defining some high-level commands that utilizes low-level commands is surely logical. Therefore, four main high-level CPP commands - CCPP, Rectangular Inward Spiral, Parallel Swath and Random CPP algorithms are introduced.

Inside those high-level path planning algorithms, some heuristics like obstacle avoidance also exists. Obstacle Avoidance command runs as an interrupt routine for the path planner. If there is no obstacle encounter during the motion, path planner just works as a trajectory controller.

To perform all those low-level and high-level commands in a sequence, there is a one-dimensional array which is called Commands-Array. This array lists all the commands to be performed by the robot. When autonomous mode is selected, if there is no command in the Commands-Array, a random command is automatically added to this array. If the array has one or more commands in it, the first ordered command is initiated firstly. When first command is completed, this command is removed from the Commands-Array. Also, all remaining elements in the Commands-Array are reordered. Hence, there is a first-come, first-served buffering operation.

When all low-level and high-level commands are considered, these commands may reserve more than one element of Commands-Array. Because all low-level and high-level commands have an identification number and parameter. For example, line command reserve two elements of Commands-Array. One of them is required for the command-ID and the other one is required for the parameter of length. Similar to line command, rotate command demands two element size from the Commands- Array.

One of them is required for command-ID and other one is required for the parameter of rotation angle.

For high-level commands, situation is a quite different. Likewise to low-level commands, high-level commands reserve one element from Commands-Array for the command-ID at the beginning. Additionally, they reserve one more elements at the end of the high-level command. Elements between them are reserved for the low-level commands which are added by high-level commands.

All calculations such as derivation, integration, etc. has been performed in a time interval that is determined by timer interrupt. For all other necessary information such as status of IR sensors, tick counts from encoders, battery status and current values, slave microcontrollers were used. Thus, main controller gathers related information from these slaves in each time interval for updating their current values.

All low-level commands check the status of reaching target point in each time interval. So, when a low-level command was finished, it states that robot reaches the target point coordinates with the given tolerance. Because of that reason, initialization calculations of next low-level command is done by using the target point coordinates and heading angle of previous command. Trajectory and CPP generation commands are explained below.

4.4.2 Trajectory Generation Commands

4.4.2.1 Line Command

Command Syntax: lin (l)

l: desired length of the line to be tracked

When the line command is running first time, wheel speeds ratio (Section 4.3), target point coordinates and desired heading angle for the straight line are calculated with respect to the current position and heading angle. Furthermore, the position error and the heading angle error values are calculated in each time interval.

4.4.2.2 Rotate Command

Command Syntax: rot (Δθ)

Δθ: desired rotation angle for the heading of the robot

With a similar manner, wheel speeds ratio, target point coordinates and desired heading angle are calculated depending on the current position and heading angle values. The position error and the heading angle error values are also calculated in each time interval.

4.4.3 CPP Generation Commands

4.4.3.1 CCPP Command

Command Syntax: ccpp(n)

n: the identification number of predefined border definition

This command initiates the uniquely developed CPP algorithm, which consists of low-level trajectories.

When CCPP command starts to run first time, necessary low-level commands which are calculated according to conditional selection are added to Commands-Array. Moreover, initialization information of CCPP command is removed from Commands-Array by this function. When the low-level commands are generated by CCPP command, final information of this command is also removed from the Commands-Array.

Mathematical details of CCPP algorithm is presented in Section 4.2.3.4.

4.4.3.2 Random CPP Command

Command Syntax: rPa()

Random CPP command is another high-level command which utilizes low-level line and rotate commands in a suitable sequence.

When random CPP command starts to run first time, necessary low-level commands which are calculated depending on the current position and heading angle values are added to Commands-Array with initialization information for reaching the desired trajectory. Furthermore, initialization information of random CPP command is removed from Commands-Array by this function. When the low-level commands which are generated by random CPP command have been done, information of this command is also removed from the Commands-Array.

Mathematical details of random CPP algorithm is presented in Section 4.2.3.3.

4.4.3.3 Parallel Swath CPP Command

Command Syntax: pSw(h, w)

h: desired height of the area to be mowed
w: desired width of the area to be mowed

Parallel Swath CPP again utilizes low-level commands and geometric information of working area for path generation.

As the blade width of the ALM is constant, Parallel Swath CPP command offsets line trajectories by this magnitude and fulfills the working area by this low-level commands. Those commands are added to and deleted from the Commands-Array similarly with other CPP algorithms.

Mathematical details of random CPP algorithm is presented in Section 4.2.3.1.

4.4.3.4 Rectangular Inward Spiral CPP Command

Command Syntax: iSp (h, w)

h: desired height of the area to be mowed
w: desired width of the area to be mowed

Rectangular Inward Spiral CPP command works same with Parallel Swath CPP algorithm, but the trajectory orientations and magnitudes are calculated along spiral pattern.

Mathematical details of random CPP algorithm is presented in Section 4.2.3.2.

4.4.4 Obstacle Avoidance Command

Command Syntax: oAv()

Obstacle avoidance command is a high level command. Similarly, this high level command uses low level commands in suitable sequence.

When Obstacle Avoidance command starts to run first time, necessary low-level commands are added according to the IR sensors status. Those commands are created, and will automatically be added as urgent to the Commands-Array in order. Adding low-level commands urgently provides a priority when it is compared to other high-level commands. In other words, low-level commands which is added by obstacle avoidance command are always run before other low-level commands which are added by other path planners. Thus, there is no necessity to add initialization information to the Commands-Array. After all low-level commands which are generated by obstacle avoidance command have been performed, finalization information of this command is removed from the Commands-Array by this function.

CHAPTER 5

SIMULATION

Besides all development work, extensive simulations are also performed in order to;

- Observe ideal and erroneous paths of the ALM for different CPP methods
- Compare ideal and erroneous coverage performances of CCPP technique
- Visualize behavior of CCPP technique for different working environments
- Compare conventional CPP and CCPP technique performances

For this purposes, a unique, specialized simulation software with GUI is built up.

In this section, simulation environment, simulation scenarios and simulation results are presented in detail.

5.1 Simulation Environment

Two main parameters are taken into account while making the decision about which environment should the simulator be developed in. First, considering the actual controller hardware, readily available libraries or objects of any programming language should not be used for the mathematical models and control algorithms side. All the codes on this side should be applicable to the actual controller hardware. Second, the programming language should have the ability to develop graphical drawing capabilities for the graphical user interface side. Considering these criteria, Visual Basic (VB) seems to be suitable for developing this simulator with aforementioned purposes.

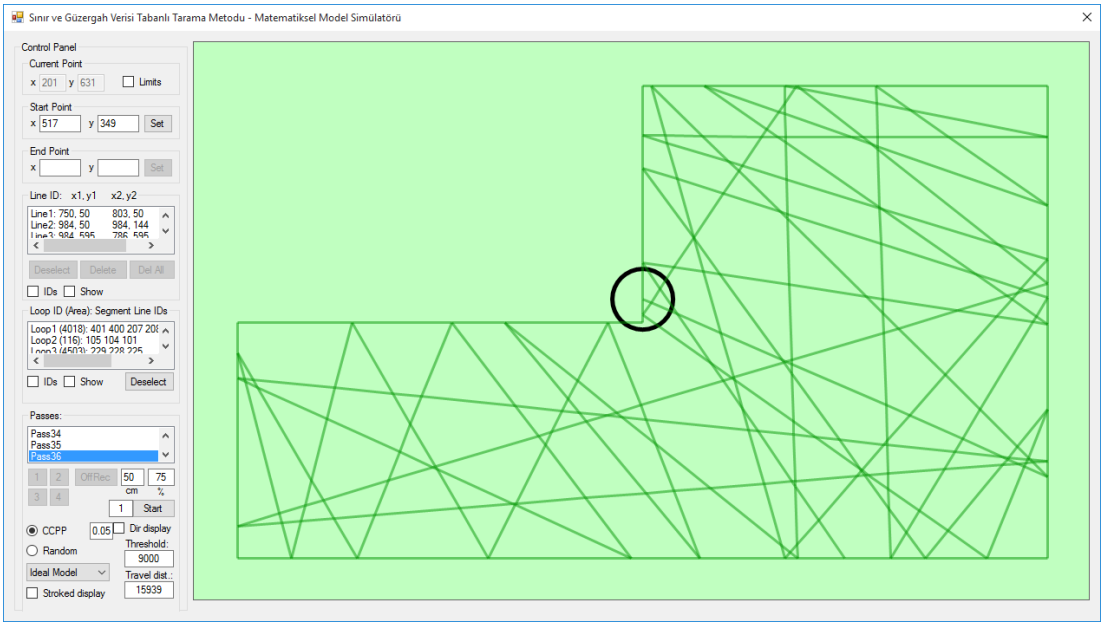


Figure 5.1: Simulator GUI

Visual Basic is a third-generation event-driven programming language and associated development environment (IDE) from Microsoft. One can put together an application using the components provided with Visual Basic itself. The language not only allows programmers to create simple graphical user interface (GUI) applications, but can also develop complex applications as well. Programming in VB is a combination of visually arranging components or controls on a form, specifying attributes and actions of those components, and writing additional lines of code for more functionality. Since default attributes and actions are defined for the components, a simple program can be created without the programmer having to write many lines of code.

The latest version of Visual Basic (VB 2015) is used as the main programming language to develop the simulator application within this thesis. Visual Basic 2015 IDE comprises a few windows; the Form window, the Solution Explorer window and the Properties window. It also consists of a toolbox which contains many useful controls that allows a programmer to develop Visual Basic 2015 applications.

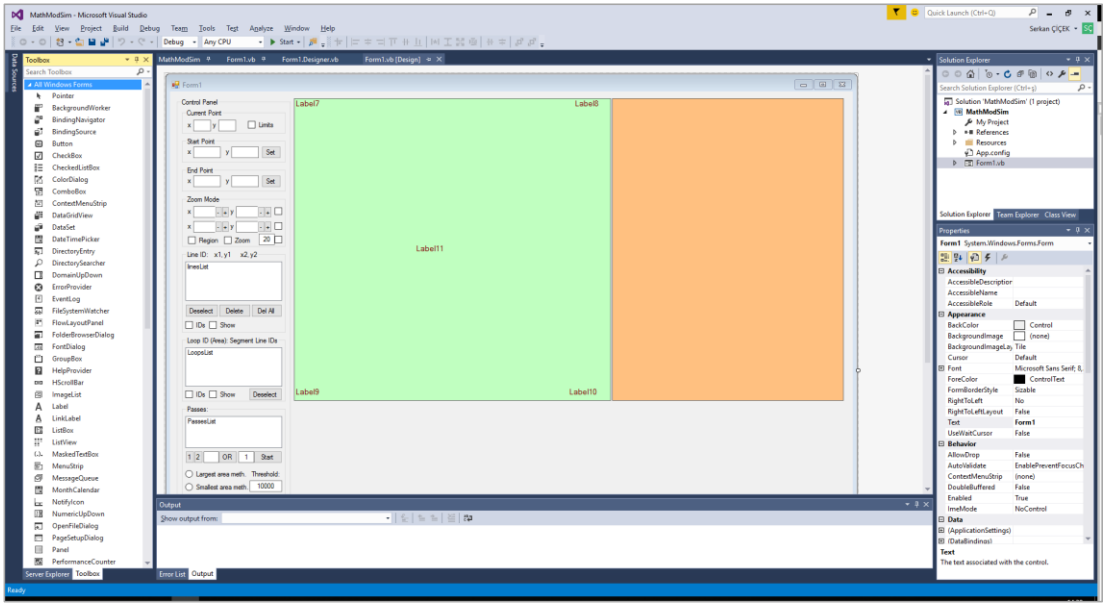


Figure 5.2: VB Design Interface

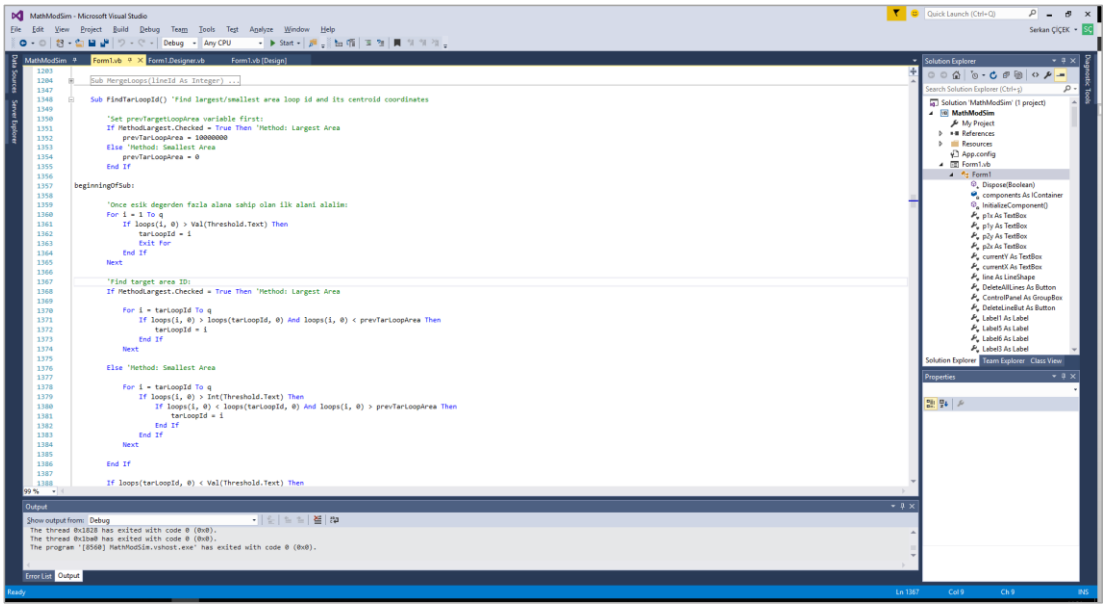


Figure 5.3: VB Code Window

5.2 Simulation Scenarios and Aspects

Simulations of CPP techniques are differentiated over environment geometry (whether it is concave or convex) and CPP techniques that are used (parallel swath, CCPP, etc.).

Also in both segments, ideal and erroneous behaviors of ALM are simulated. Simulation scenarios are summarized in Table 5.1.

Table 5.1: Simulation Scenarios

		Ideal Path Simulations	Erroneous Path Simulations
Convex Polygon	CCPP	+	+
	Random CPP	+	+
	Parallel Swath CPP	+	+
	Inward Spiral CPP	+	+
Concave Polygon	CCPP	+	+
	Random CPP	+	+
	Parallel Swath CPP	+	+
	Inward Spiral CPP	+	+

For simulations over convex polygon, a rectangular area of 4.0 m x 2.5 m (10 m^2) is decided to be used. For concave polygon simulations, one-fourth of this area is dismissed and the resulting L-shaped geometry is used (total of 7.5 m^2). This rectangular and L-shaped geometries are the most types of common domestic gardens. Schematics for polygon geometries are illustrated in Figure 5.4.

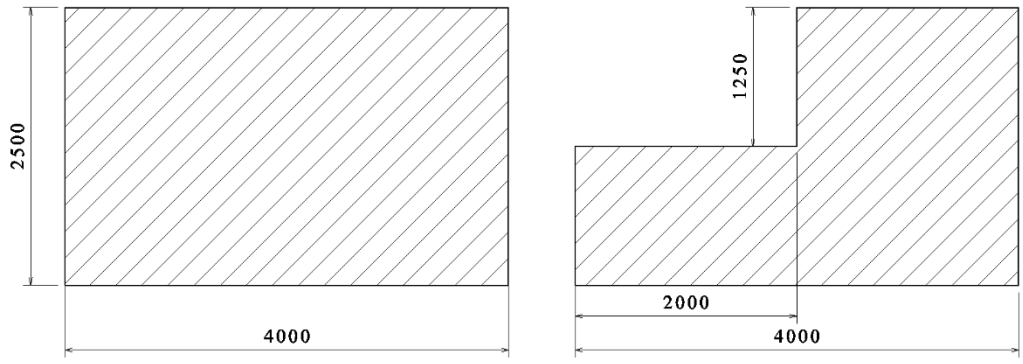


Figure 5.4: Convex and Concave Polygon Geometries Used in Simulations

For deterministic CCPP technique, three simulations are performed over each polygonal area with a different starting point. For random CPP method, ten simulations are performed over each polygon with two different starting points (5 simulations for each starting point).

For deterministic Parallel Swath and Inward Spiral CPP techniques, three simulations are found sufficient for each polygonal area regarding that only erroneous coverage is going to be differ between simulations.

The main aspect of simulations is to understand conditional behavior of CCPP technique and to compare this technique with conventional CPP methods. The comparison metrics are chosen as coverage percentage, travelled distance (analogous with completion time) and energy consumption.

The simulation scenarios are also used in outdoor tests in order to make meaningful comparisons.

5.3 Simulation Results

This section presents simulation results in detail. In each section, results of a single simulation is presented as a sample.

5.3.1 Convex Polygon Simulation Results

5.3.1.1 Convex Polygon CCPP Simulation Results

Figures 5.5 - 5.8 represent one sample from CCPP simulations with an initial starting point of (0 m, 0 m).

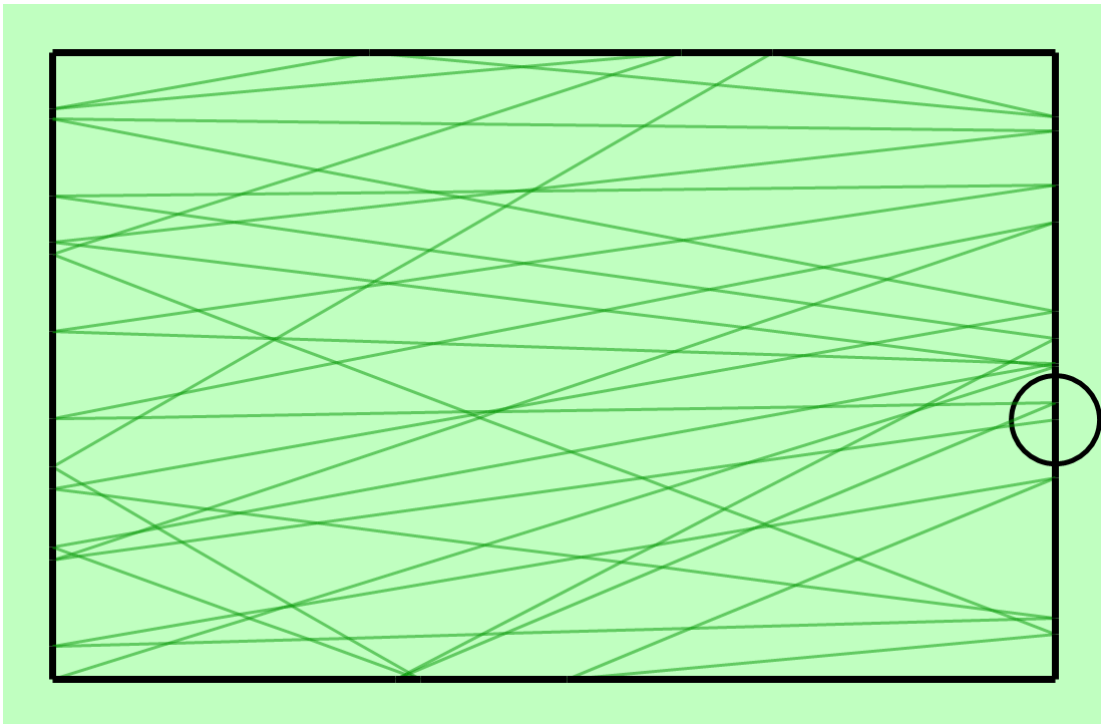


Figure 5.5: Convex Polygon CCPP Ideal Path

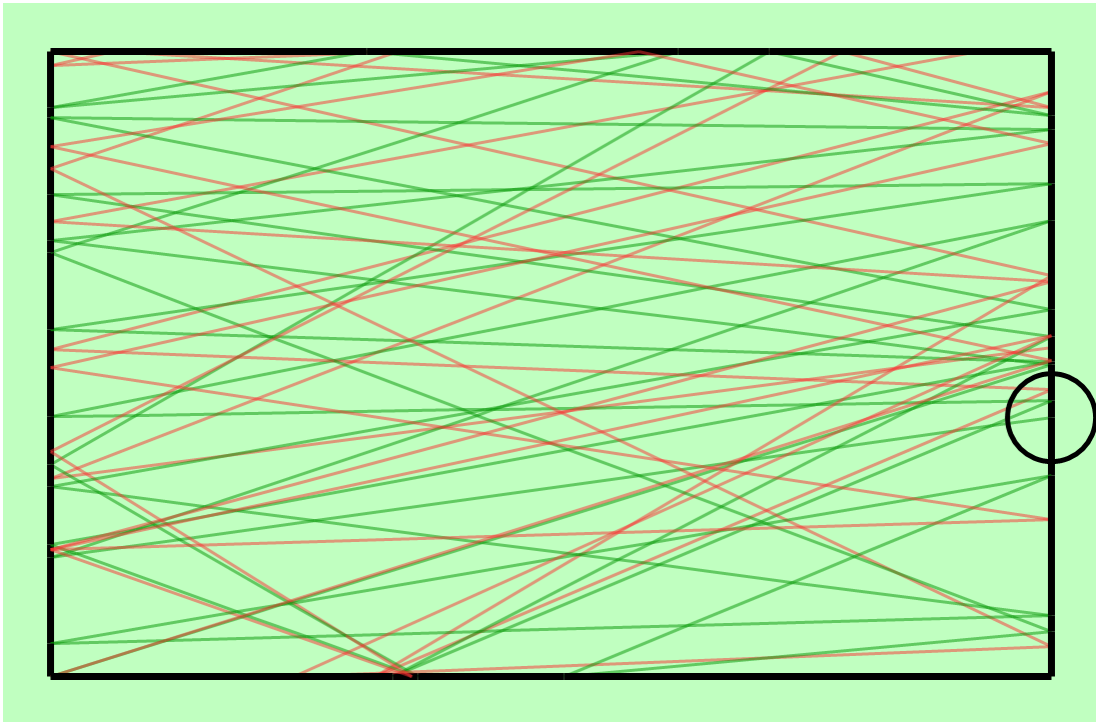


Figure 5.6: Convex Poly. CCPP Ideal (Green) and Err. (Red) Paths

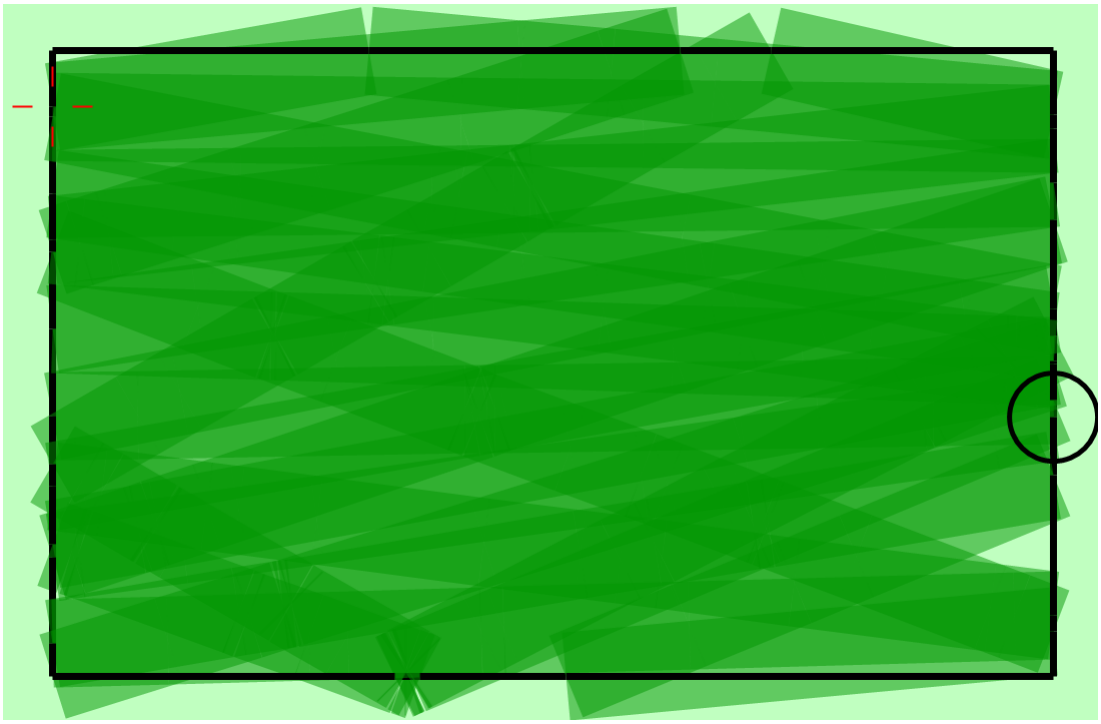


Figure 5.7: Convex Polygon CCPP Ideal Coverage

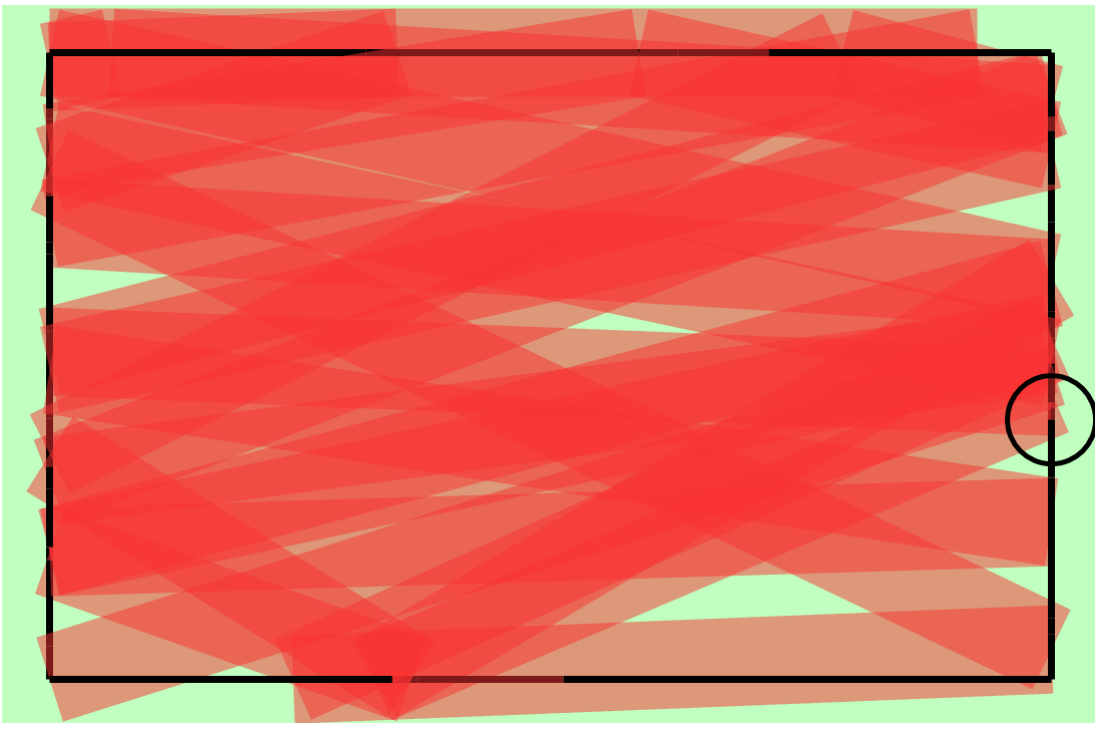


Figure 5.8: Convex Polygon CCPP Erroneous Coverage

5.3.1.2 Convex Polygon Random CPP Simulation Results

Figures 5.9 - 5.12 represent one sample of random CPP simulations with an initial starting point of (1.0 m, 0 m).

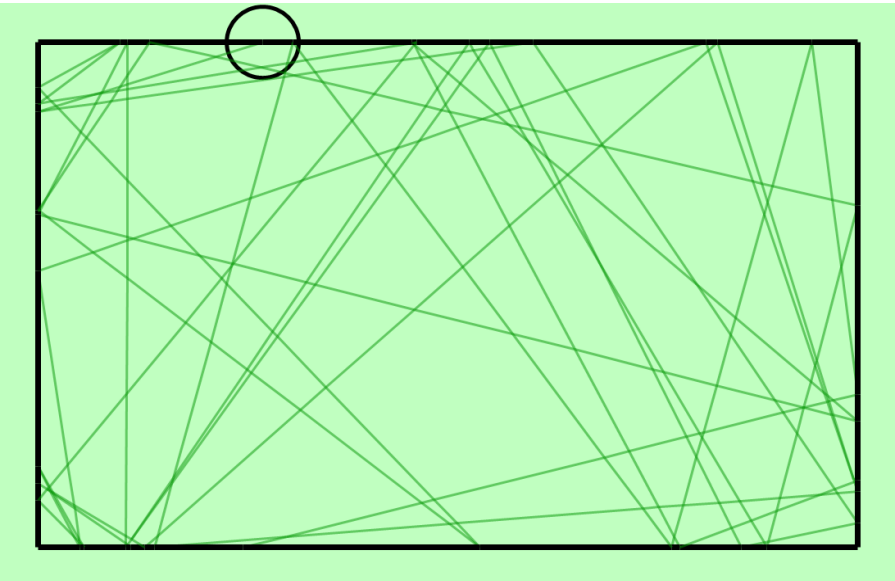


Figure 5.9: Convex Polygon Random CPP Ideal Path

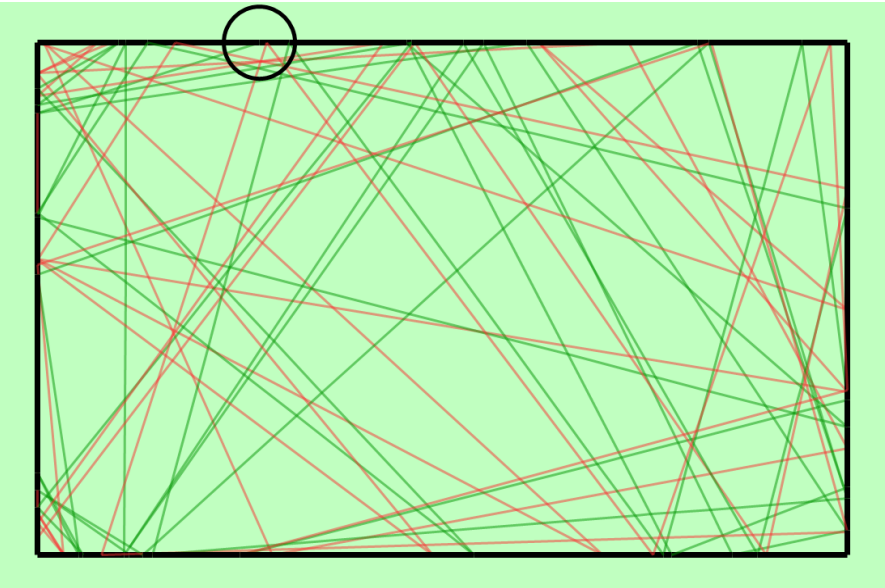


Figure 5.10: Convex Poly. Random CPP Ideal (Green) and Err. (Red) Paths

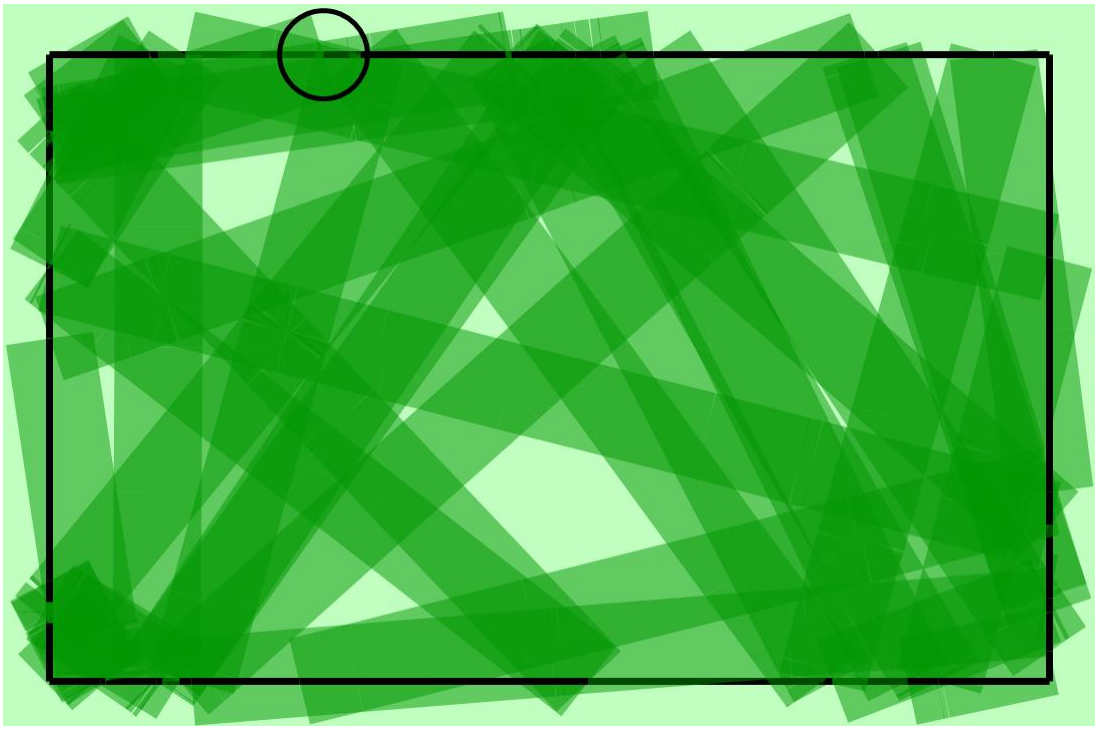


Figure 5.11: Convex Polygon Random CPP Ideal Coverage

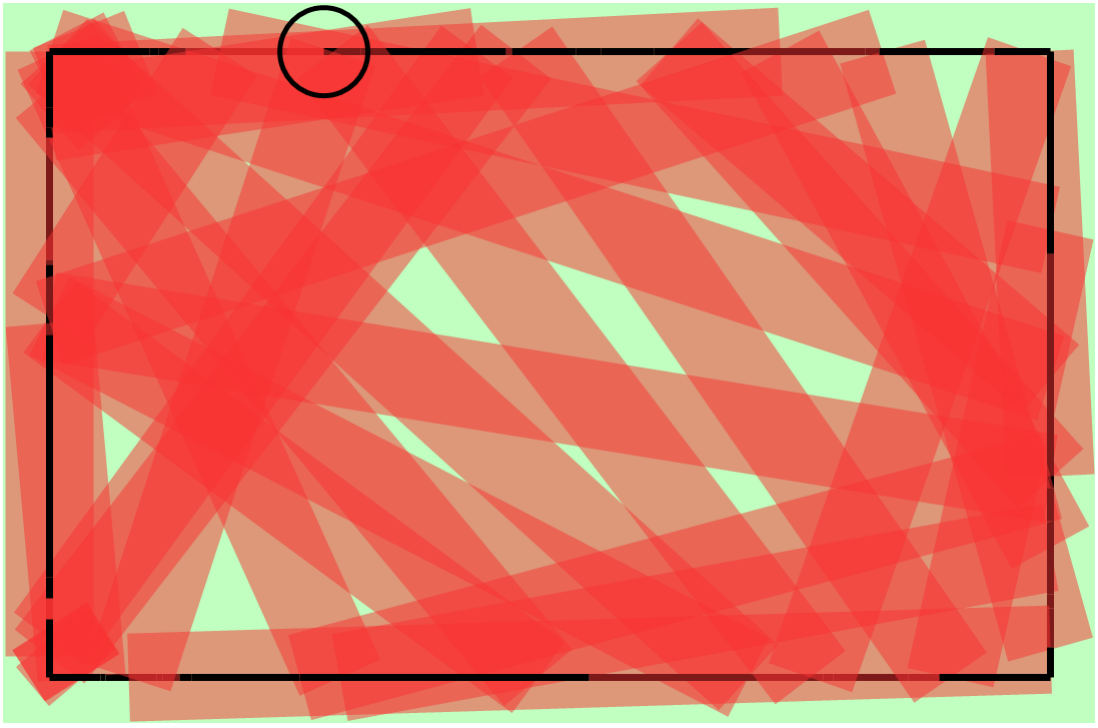


Figure 5.12: Convex Polygon Random CPP Erroneous Coverage

5.3.1.3 Convex Polygon Parallel Swath CPP Simulation Results

Figures 5.13 - 5.16 represent one sample of parallel swath CPP simulations with an initial starting point of (4.0 m, 0 m).

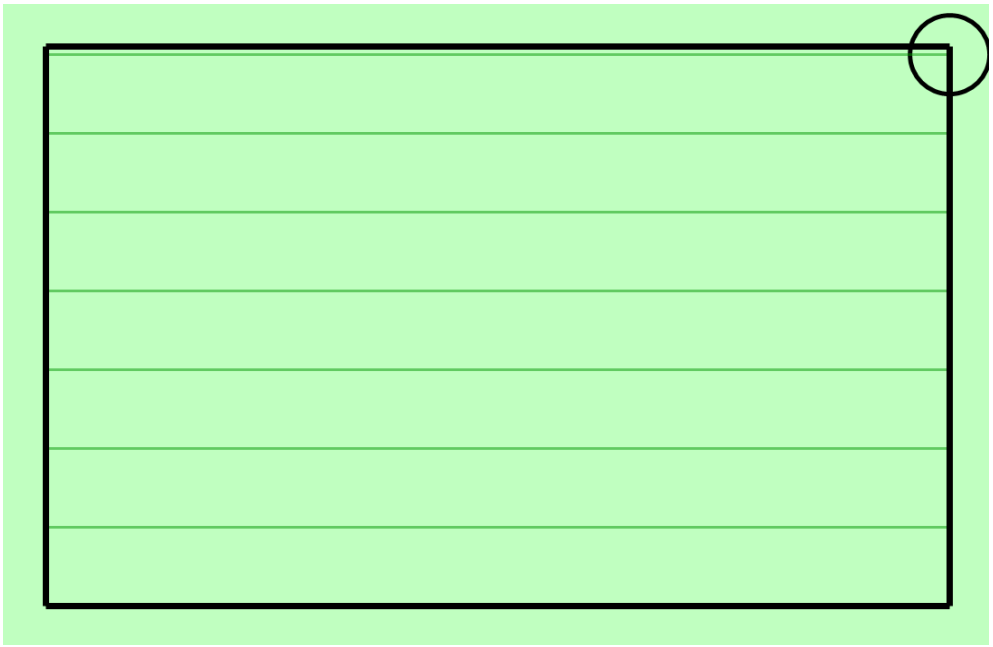


Figure 5.13: Convex Polygon Parallel Swath CPP Ideal Path

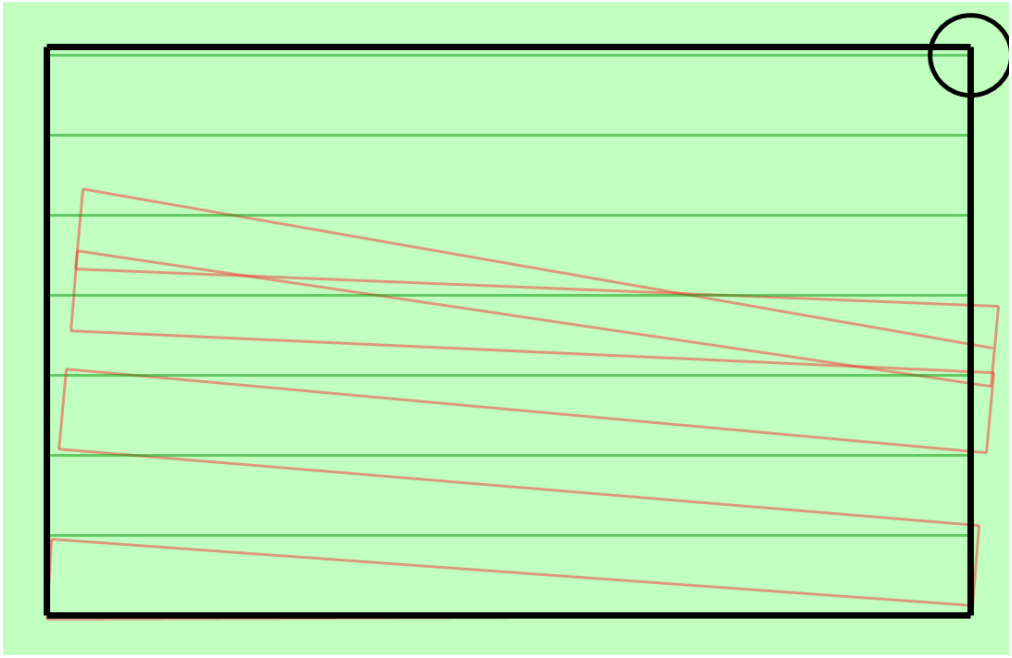


Figure 5.14: Convex Poly. Parallel Swath CPP Ideal (Green) and Err. (Red) Paths

Figure 5.15: Convex Polygon Parallel Swath CPP Ideal Coverage

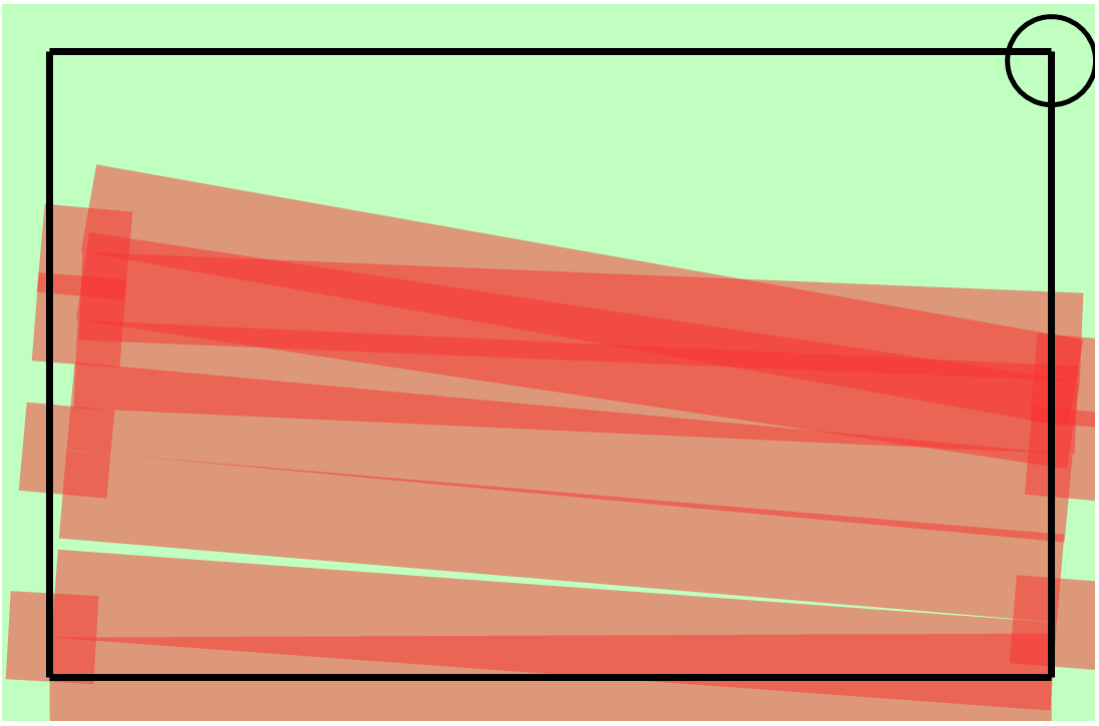


Figure 5.16: Convex Polygon Parallel Swath CPP Erroneous Coverage

5.3.1.4 Convex Polygon Inward Spiral CPP Simulation Results

Figures 5.17 - 5.20 represents one sample of inward spiral CPP simulations with an initial starting point of (4.0 m, 0 m).

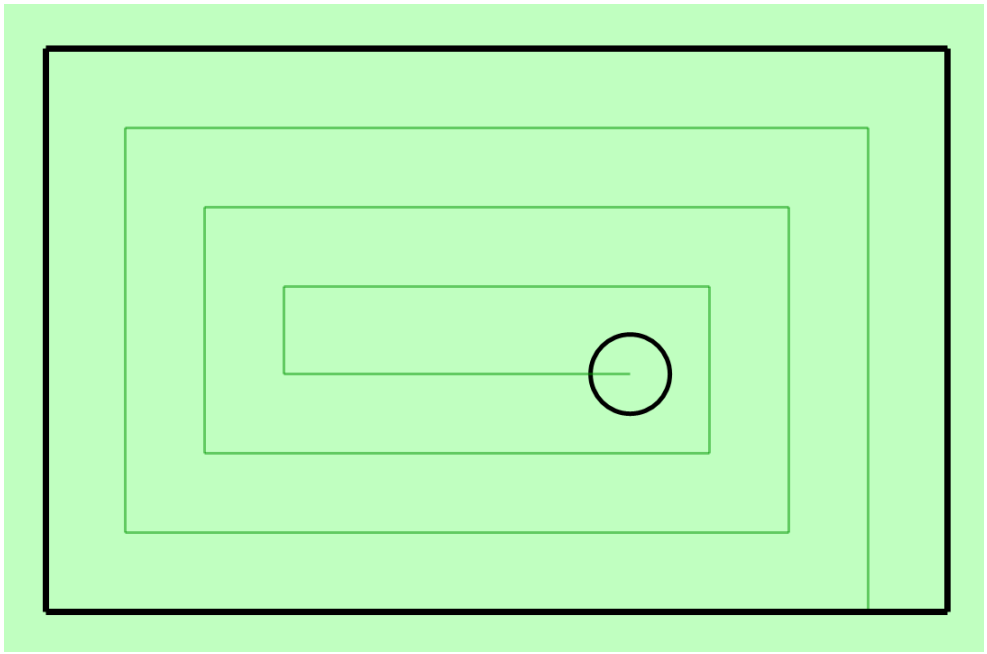


Figure 5.17: Convex Polygon Inward Spiral CPP Ideal Path

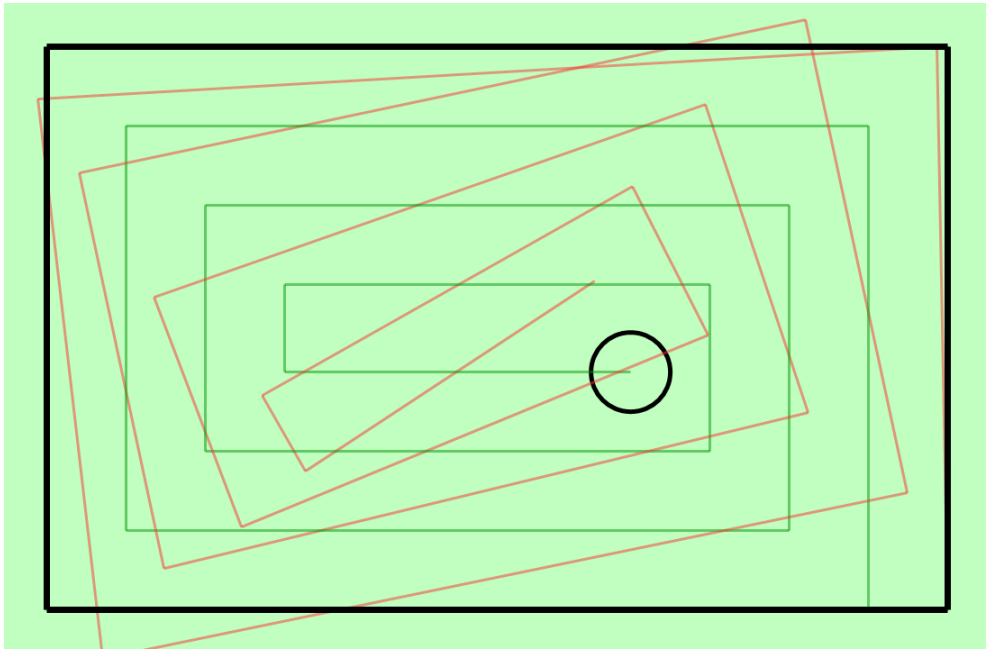


Figure 5.18: Convex Poly. Inward Spiral CPP Ideal (Green) and Err. (Red) Paths

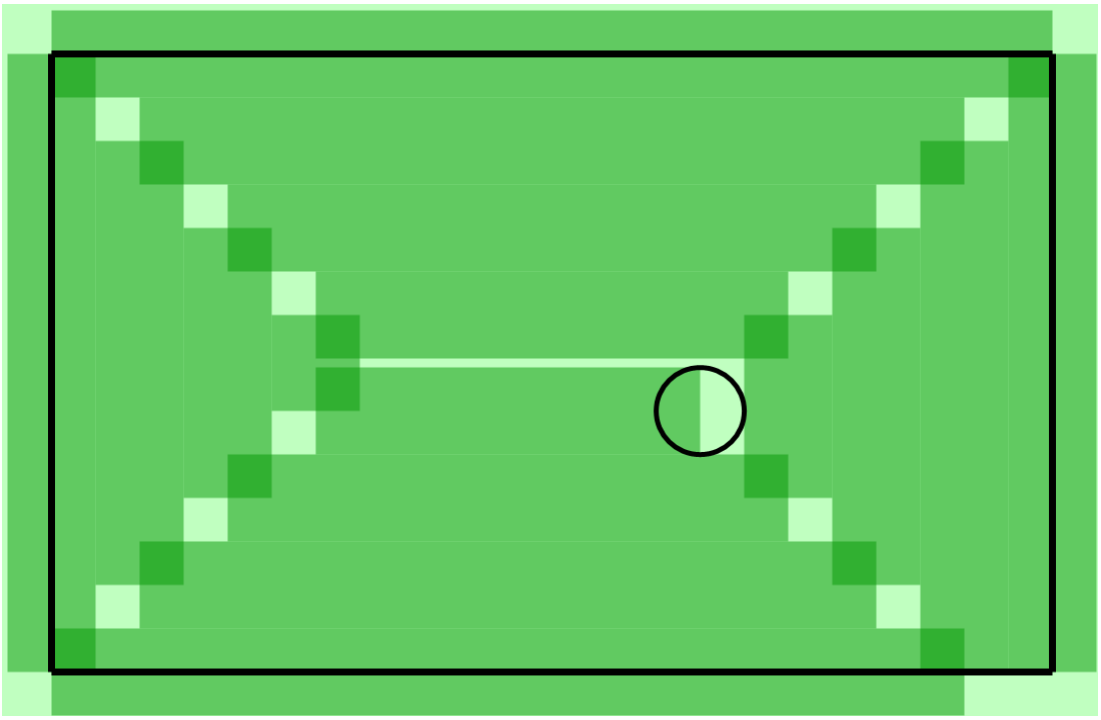


Figure 5.19: Convex Polygon Inward Spiral CPP Ideal Coverage

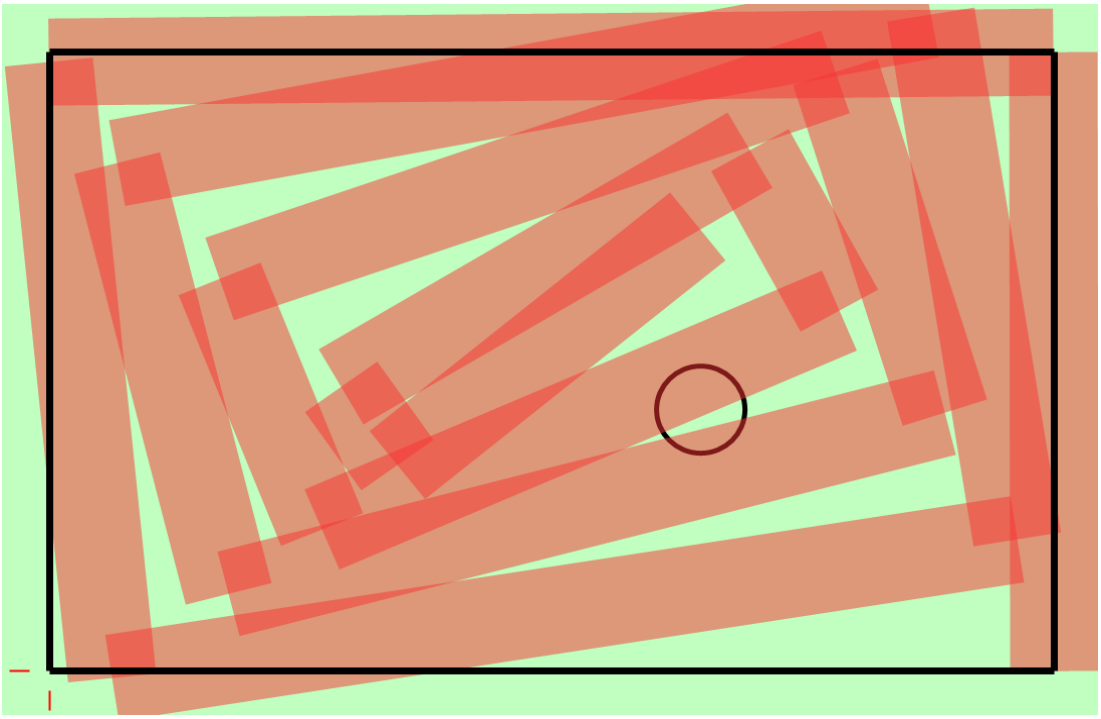


Figure 5.20: Convex Polygon Inward Spiral CPP Erroneous Coverage

5.3.2 Concave Polygon Simulation Results

5.3.2.1 Concave Polygon CCPP Simulation Results

Figures 5.21 - 5.24 represent one sample of CCPP simulations with an initial starting point of (0 m, 0 m).

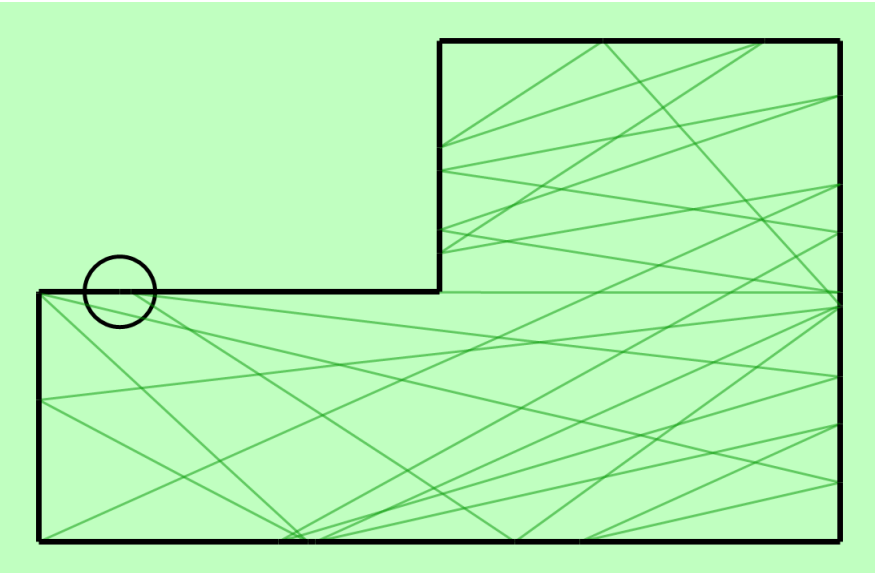


Figure 5.21: Concave Polygon CCPP Ideal Path

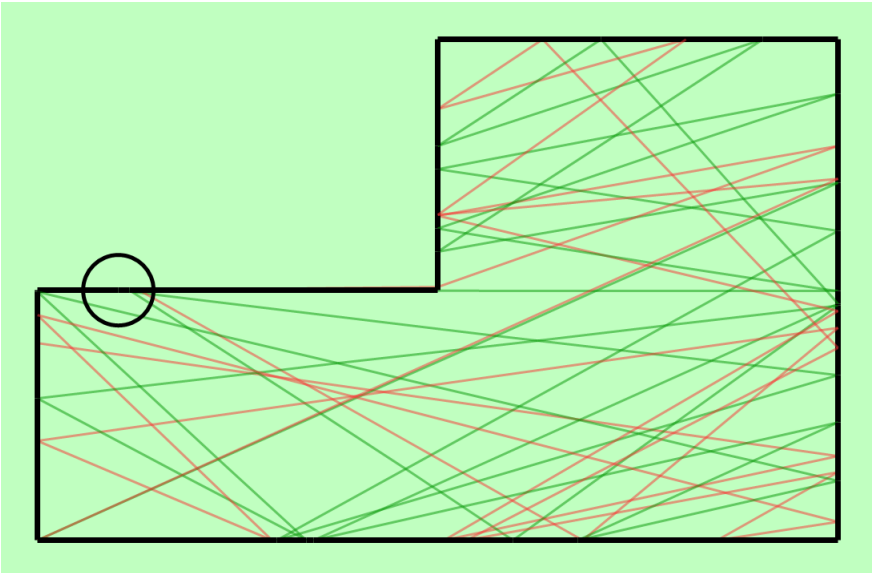


Figure 5.22: Concave Poly. CCPP Ideal (Green) and Err. (Red) Paths

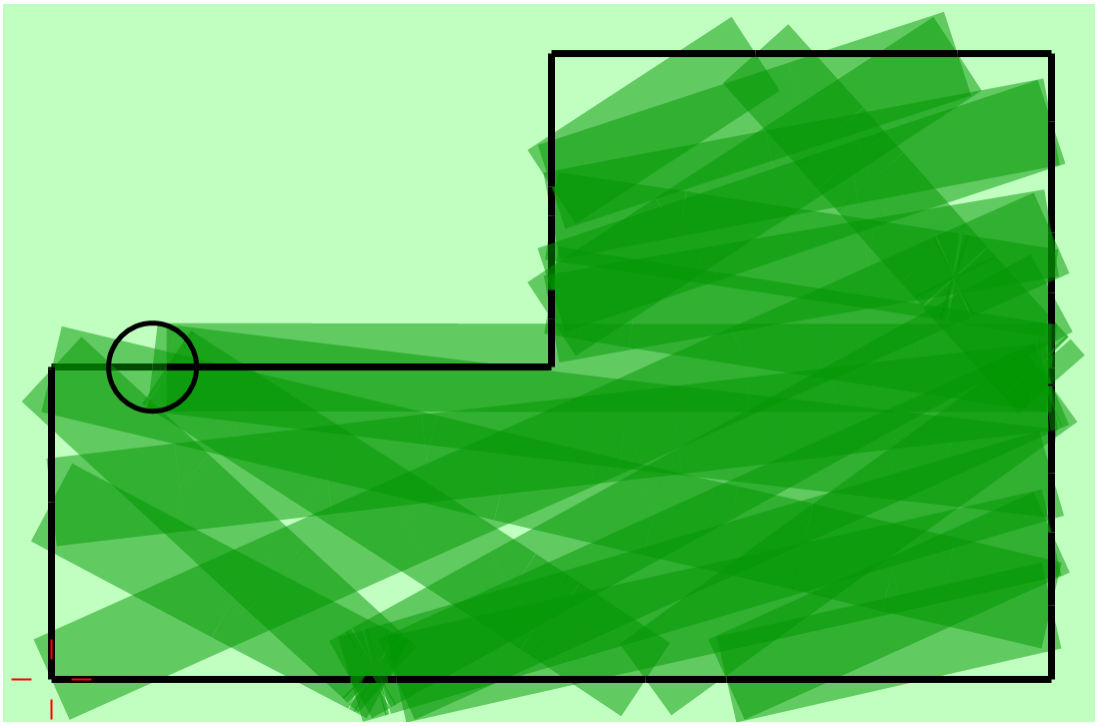


Figure 5.23: Concave Polygon CCPP Ideal Coverage

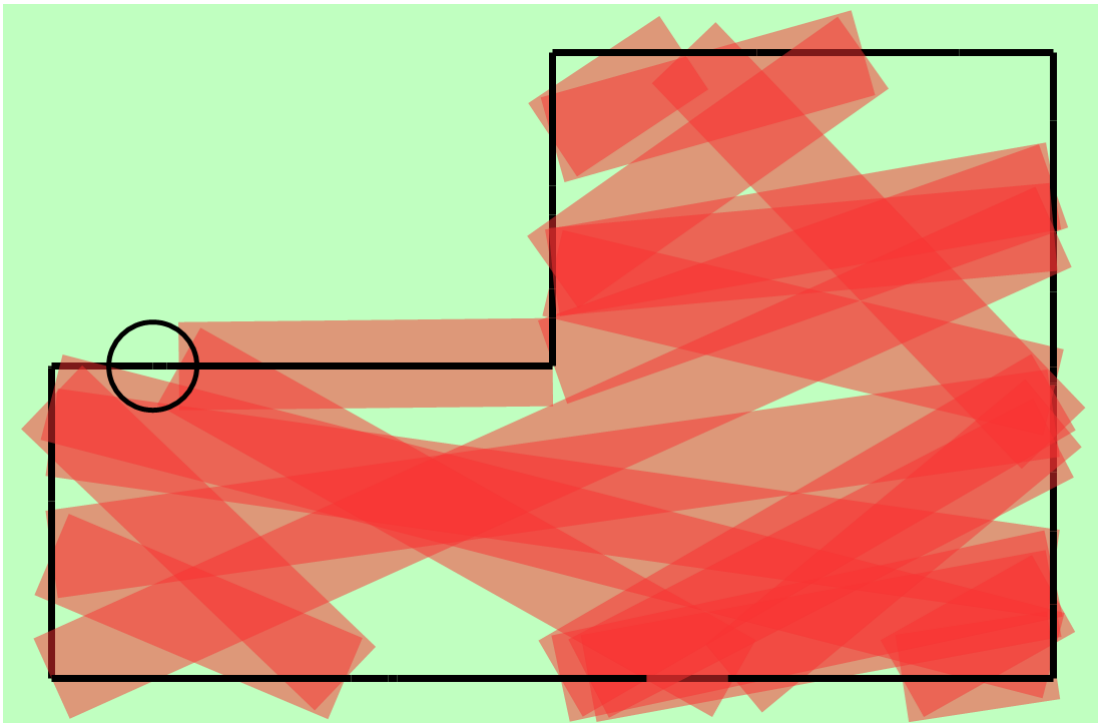


Figure 5.24: Concave Polygon CCPP Erroneous Coverage

5.3.2.2 Concave Polygon Random CPP Simulation Results

Figures 5.25 - 5.28 represent random CPP simulations with an initial starting point of (1.0 m, 0 m).

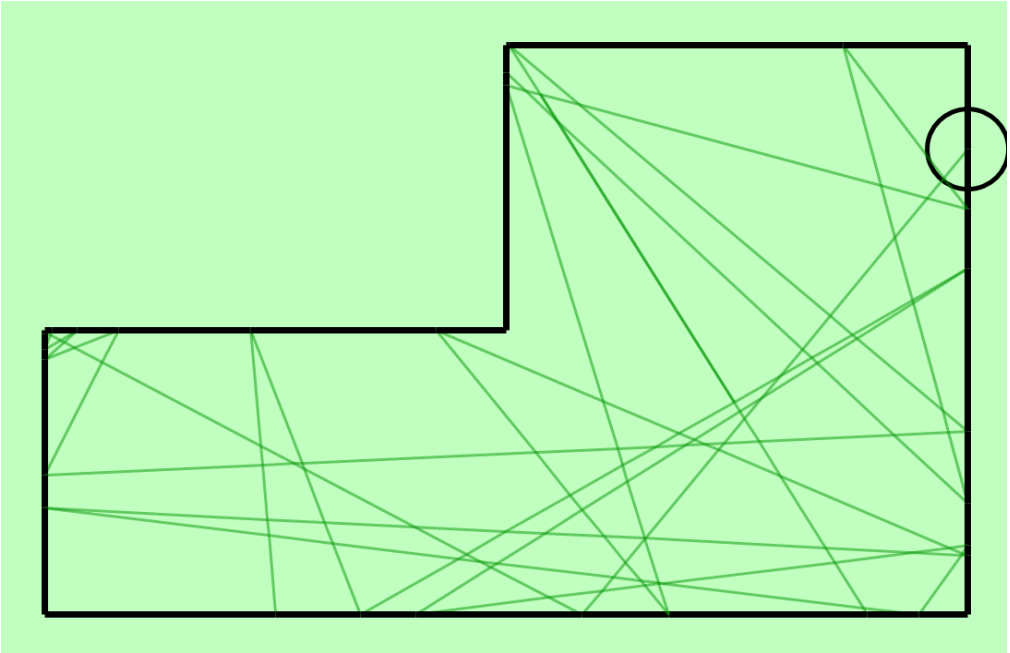


Figure 5.25: Concave Polygon Random CPP Ideal Path

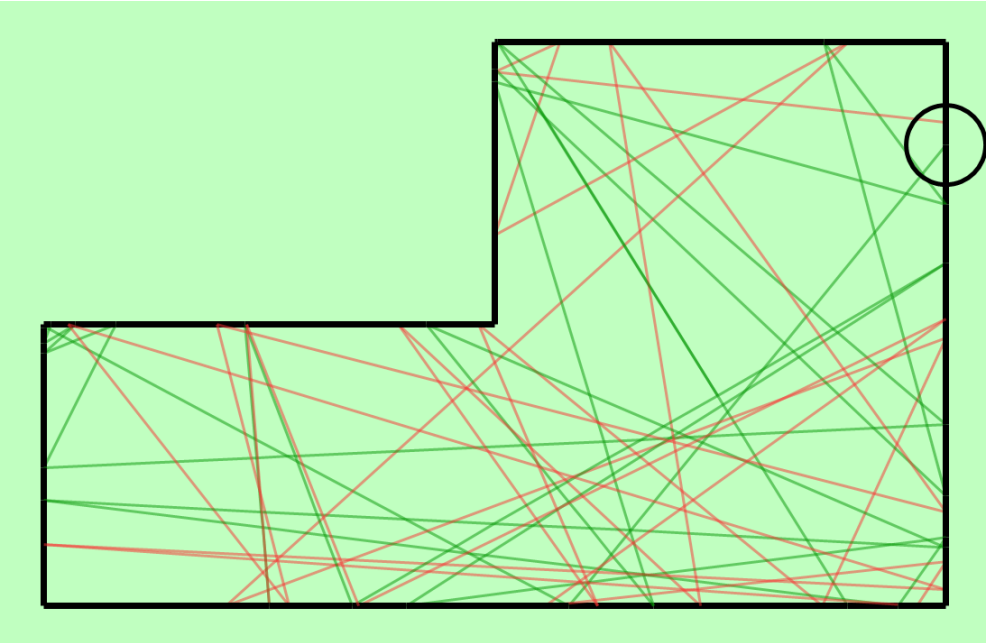


Figure 5.26: Concave Poly. Random CPP Ideal (Green) and Err. (Red) Paths

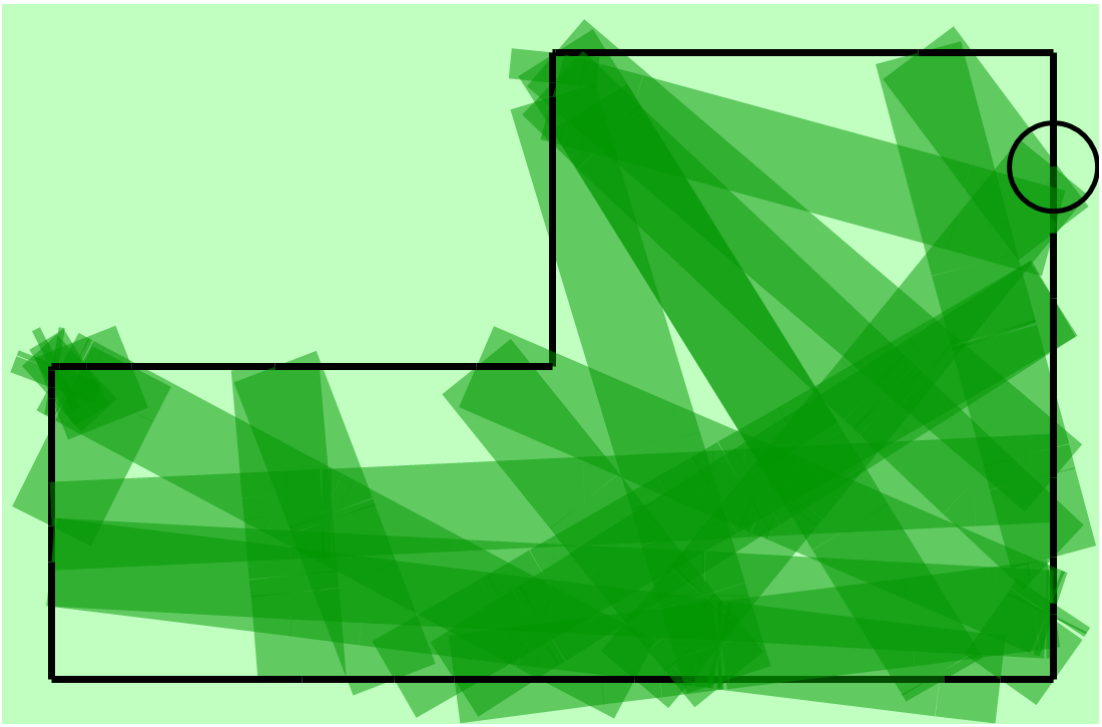


Figure 5.27: Concave Polygon Random CPP Ideal Coverage

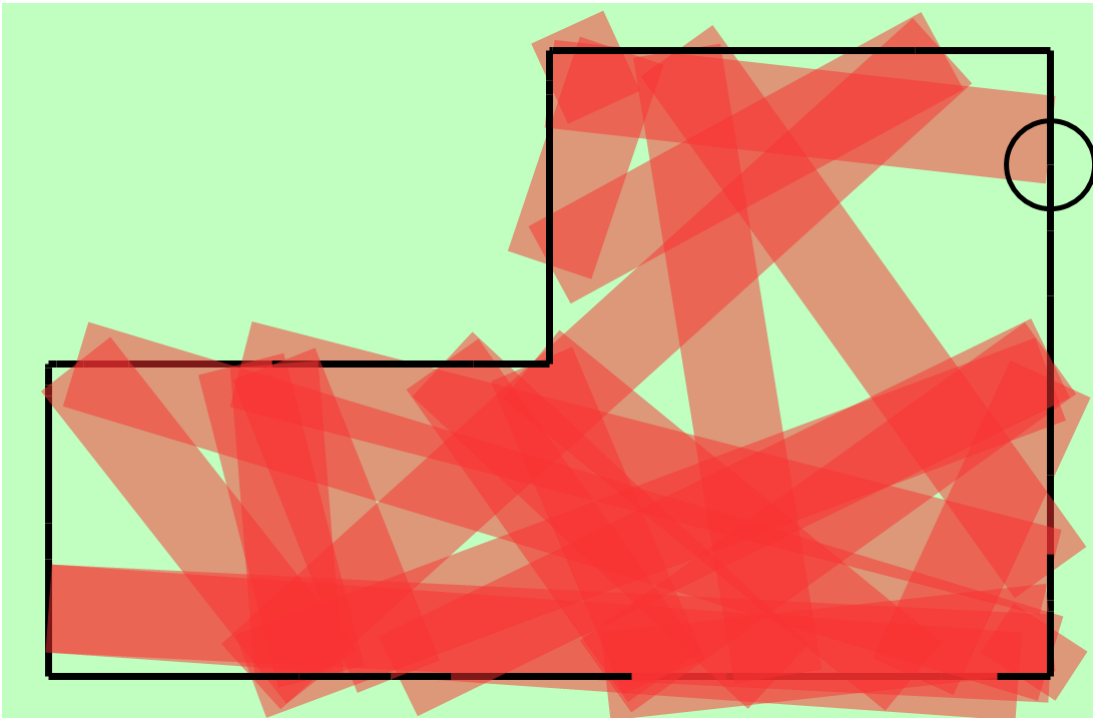


Figure 5.28: Concave Polygon Random CPP Erroneous Coverage

5.3.2.3 Concave Polygon Parallel Swath CPP Simulation Results

Figures 5.29 - 5.32 represent one sample of parallel swath CPP simulations with an initial starting point of (4.0 m, 0 m).

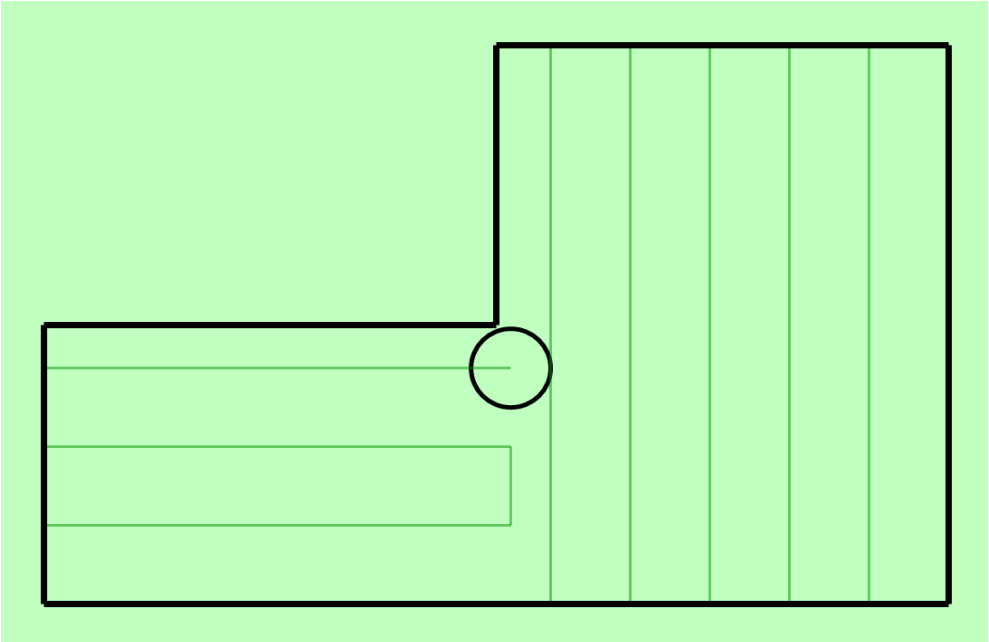


Figure 5.29: Concave Polygon Parallel Swath CPP Ideal Path

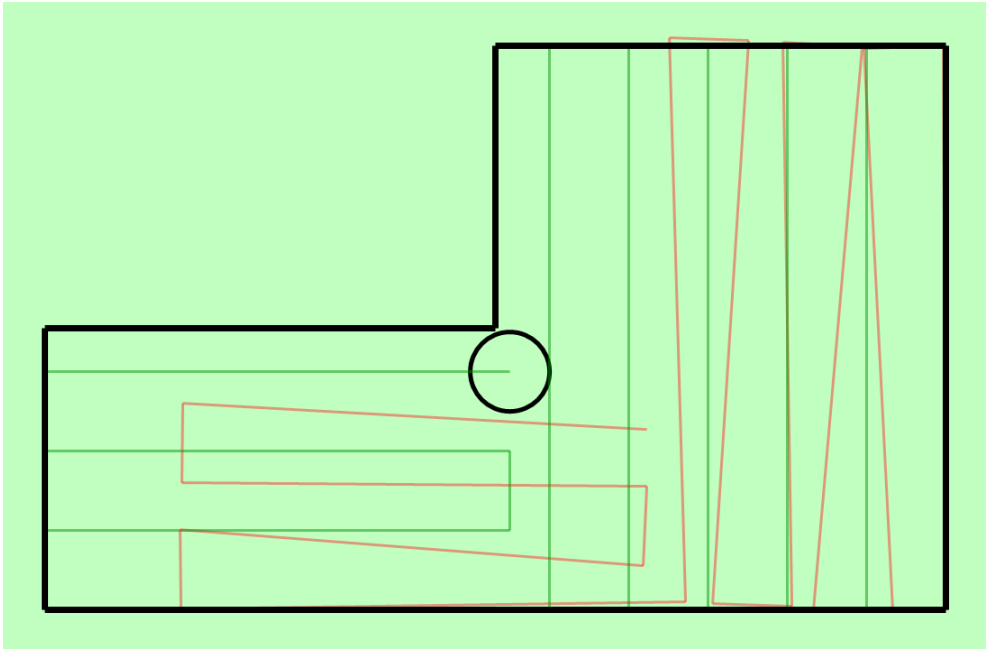


Figure 5.30: Concave Poly. Par. Swath CPP Ideal (Green) and Err. (Red) Paths

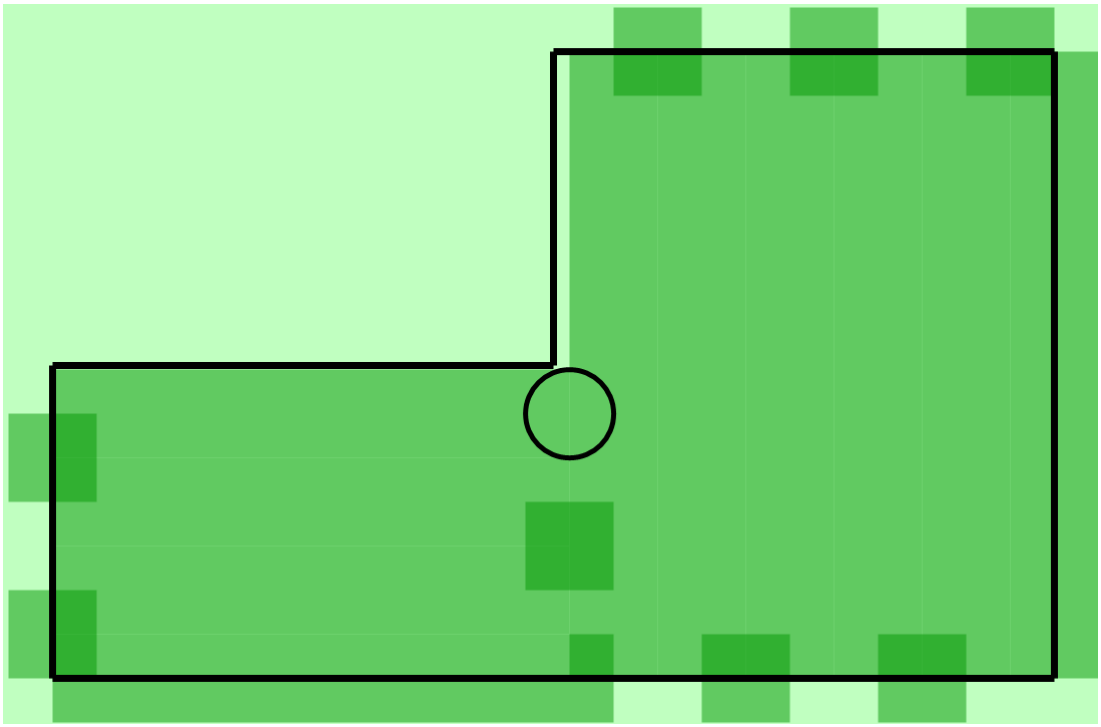


Figure 5.31: Concave Polygon Parallel Swath CPP Ideal Coverage

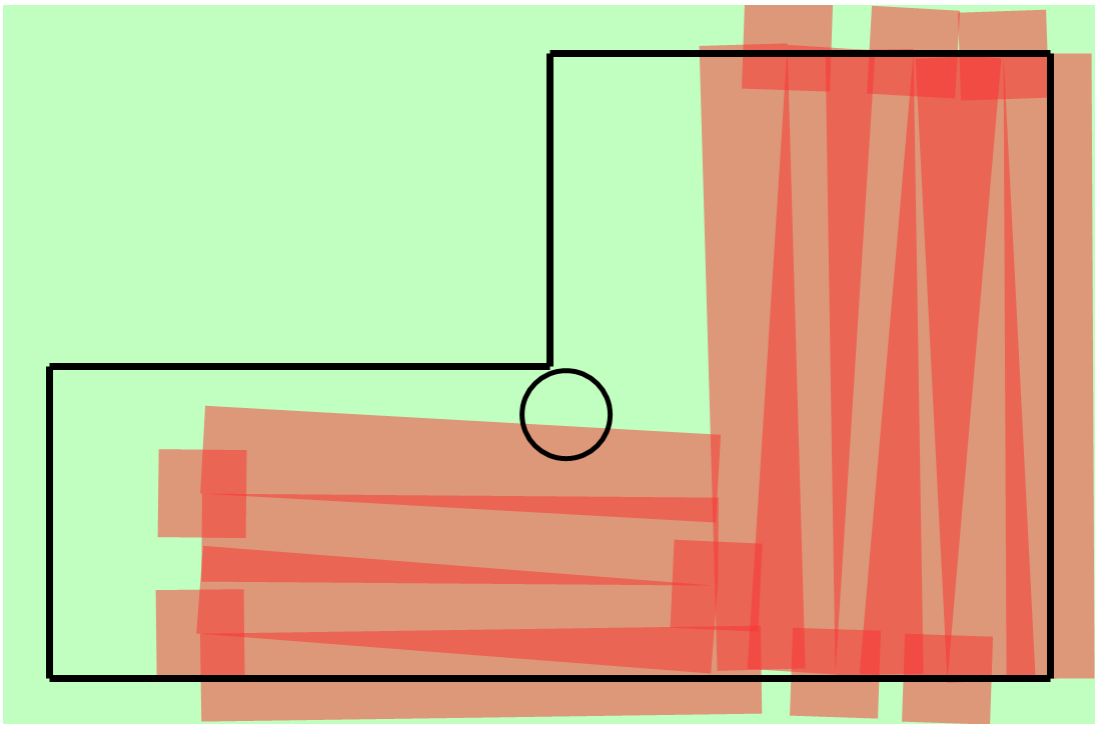


Figure 5.32: Concave Polygon Parallel Swath CPP Erroneous Coverage

5.3.2.4 Concave Polygon Inward Spiral CPP Simulation Results

Figures 5.33 - 5.36 represent one sample of inward spiral CPP simulations with an initial starting point of (4.0 m, 0 m).

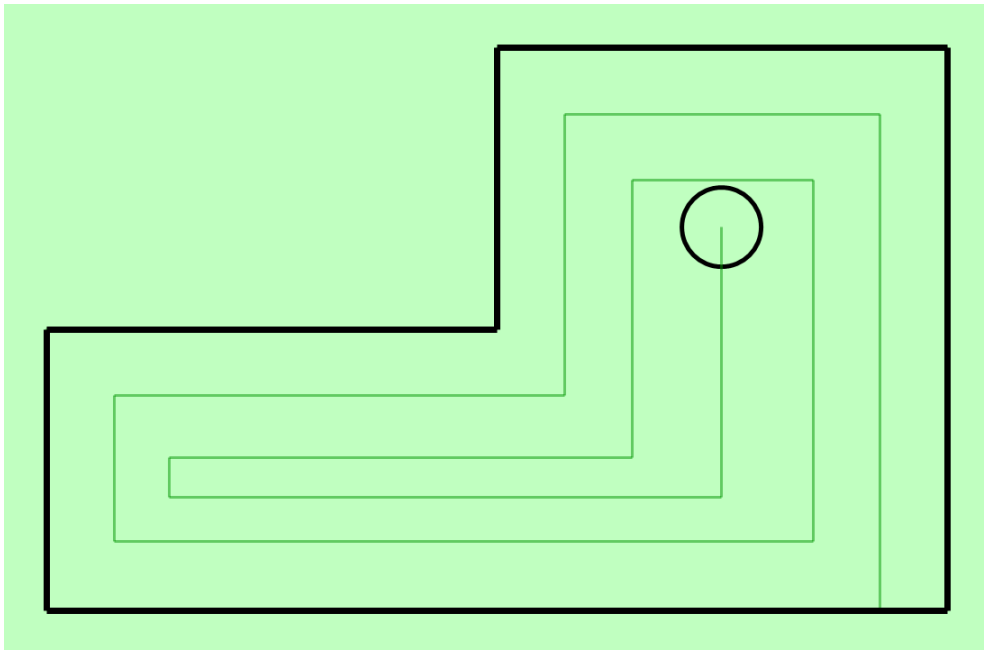


Figure 5.33: Concave Polygon Inward Spiral CPP Ideal Path

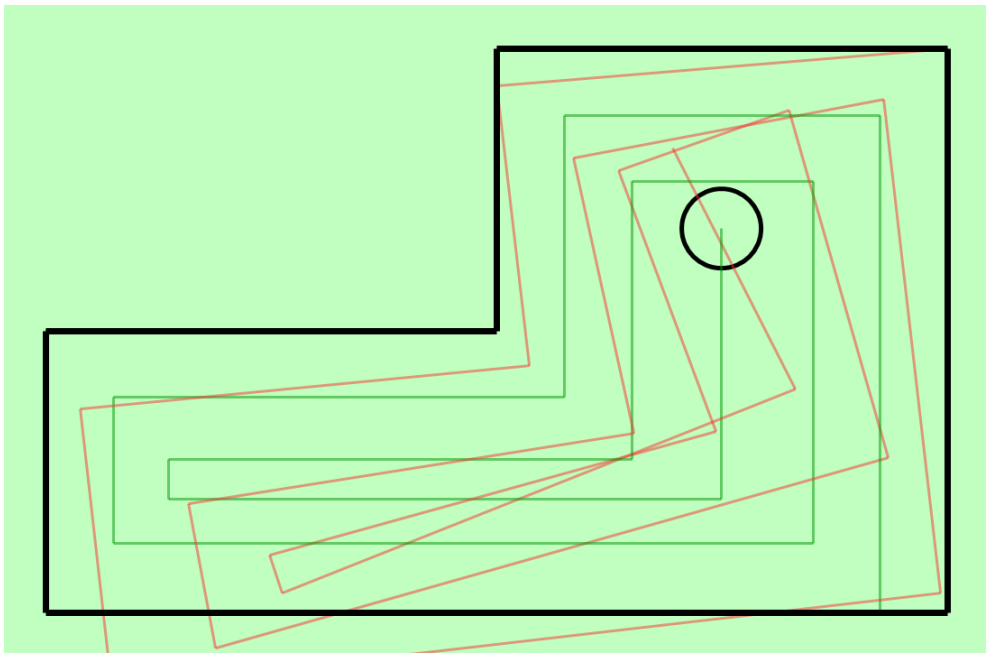


Figure 5.34: Concave Poly. Inward Spiral CPP Ideal (Green) and Err. (Red) Paths

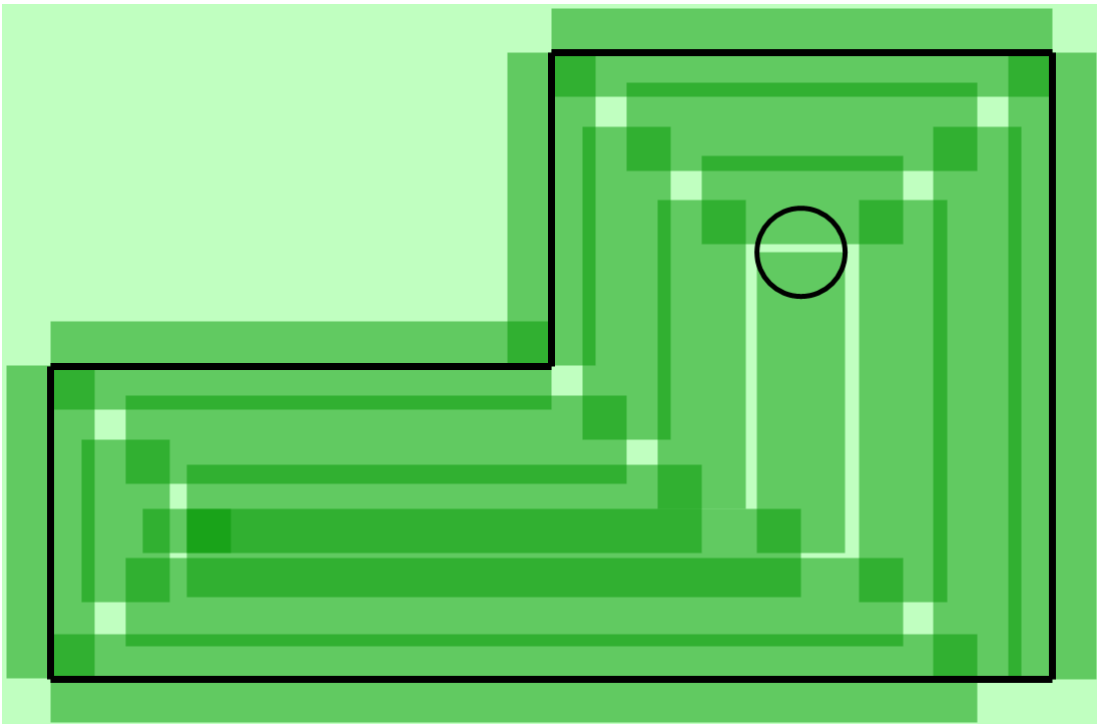


Figure 5.35: Concave Polygon Inward Spiral CPP Ideal Coverage

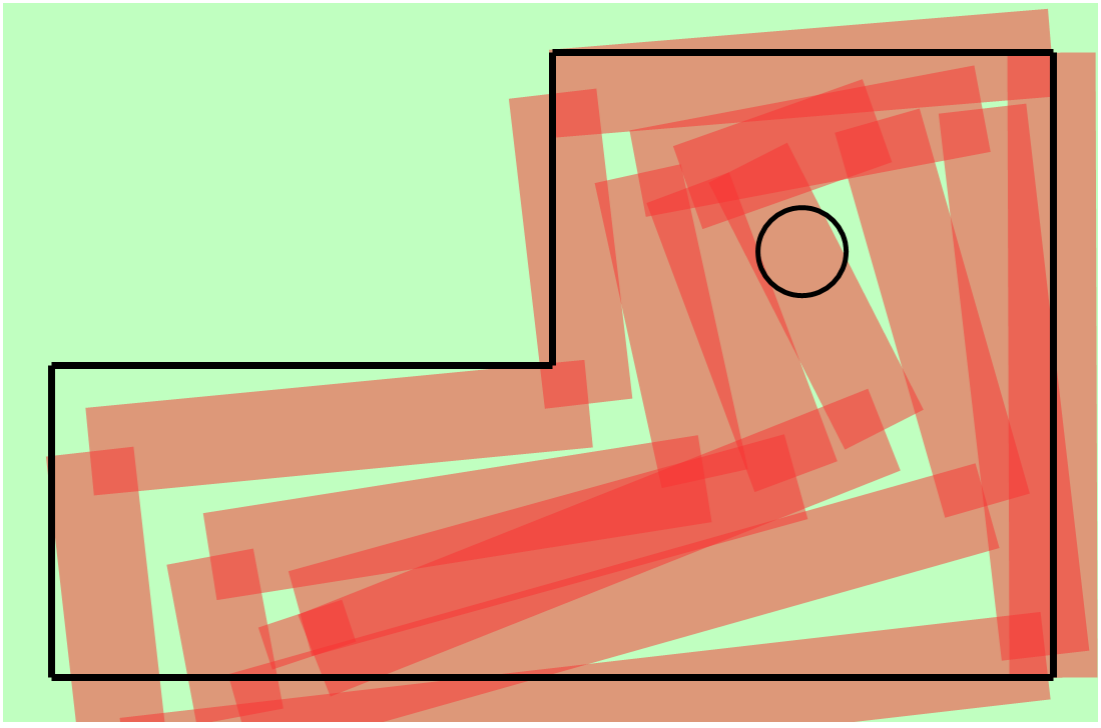


Figure 5.36: Concave Polygon Inward Spiral CPP Erroneous Coverage

5.3.3 Simulation Results Summary

This section summarizes simulation results for different CPP techniques. For result evaluation, coverage percentage, completion time and energy consumption metrics are used.

Completion time of operation might be considered analogous with travelled distance of ALM. However, the use of travelled distance or completion time metrics for energy consumption is ambiguous even if they seem similar. Travelled distance metric only measures line trajectories (rotations are not counted) and completion time metric can also be misleading about rotations.

Energy consumption of a mobile robot is directly related to the number of wheel rotations. Therefore number of encoder counts are thought to be used to evaluate energy consumption. For unit consistency, a metric called "Driven Distance" is created and used as the indicator of energy consumption, which is actually the total distance taken by two wheels. Explicitly, Driven Distance parameter is derived from the number of encoder ticks, since the parameter indicating the energy consumption should also include wasted wheel revolutions due to possible slippages.

In simulation environment, the Driven Distance parameter are calculated by just using the length parameter (l) of the line command, and the heading angle parameter ($\Delta\theta$) of the rotate command. Encoder tick calculating/counting is performed first for both simulation and the real test environments to evaluate Driven Distance. Number of encoder ticks for each trajectory is calculated by using Equations 5.1 - 5.2.

$$Tick_{Total,line} = l \times \frac{CPR_{encoder} \times i}{\pi \times R} \tag{5.1}$$

$$Tick_{Total,rotation} = \Delta\theta \times \frac{L \times CPR_{encoder} \times i}{2 \times \pi \times R} \tag{5.2}$$

After obtaining total encoder tick counts, Equation 5.3 is used in order to calculate the Driven Distance parameters.

$$Driven\ Distance = Tick_{Total} \times \frac{2 \times \pi \times R}{CPR_{encoder} \times i} \quad (5.3)$$

5.3.3.1 Simulation Results Summary for Convex Polygon

Table 5.2: Simulation Result Summary for CCPP for Convex Polygon

Convex Polygon CCPP Results				
Sim. #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Ideal	99.62	103.40	243.89
	Erroneous	92.76	102.79	242.38
2	Ideal	98.40	80.91	201.67
	Erroneous	91.75	89.90	219.19
3	Ideal	98.92	80.80	192.93
	Erroneous	90.74	82.67	195.62
Average	**Ideal**	**98.98**	**88.37**	**212.83**
	Erroneous	**91.75**	**91.78**	**219.06**

Table 5.3: Simulation Result Summary for Random CPP for Convex Polygon

Convex Polygon Random CPP Results				
Sim. #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Ideal	95.61	81.96	196.70
	Erroneous	91.90	81.30	191.86
2	Ideal	95.21	81.18	197.26
	Erroneous	88.23	70.29	168.69
3	Ideal	94.03	83.79	201.93
	Erroneous	91.05	89.03	211.00

4	Ideal	92.04	80.03	193.67
	Erroneous	82.37	85.15	198.39
5	Ideal	95.68	86.27	212.22
	Erroneous	90.78	73.59	172.20
6	Ideal	76.15	85.83	209.42
	Erroneous	71.59	78.63	186.35
7	Ideal	82.47	82.17	205.42
	Erroneous	73.65	80.59	186.16
8	Ideal	80.09	79.98	187.95
	Erroneous	75.63	72.13	173.11
9	Ideal	77.12	76.89	187.61
	Erroneous	72.45	76.01	183.56
10	Ideal	70.57	80.56	194.14
	Erroneous	71.31	78.42	186.63
Average	**Ideal**	**85.89**	**81.86**	**198.63**
	Erroneous	**80.89**	**78.51**	**185.79**

Table 5.4: Simulation Result Summary for Par. Swath CPP for Convex Polygon

Convex Polygon Parallel Swath CPP Results				
Sim #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Ideal	100.00	34.45	79.28
	Erroneous	69.71	34.45	90.28
2	Ideal	100.00	34.45	79.28
	Erroneous	47.69	34.45	90.34
3	Ideal	100.00	34.45	79.28
	Erroneous	58.59	34.45	82.35
Average	**Ideal**	**100.00**	**34.45**	**79.28**
	Erroneous	**58.66**	**34.45**	**87.65**

Table 5.5: Simulation Result Summary for Inw. Spiral CPP for Convex Polygon

Convex Polygon Inward Spiral CPP Results				
Sim. #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Ideal	100.00	34.69	86.20
	Erroneous	77.70	34.69	88.94
2	Ideal	100.00	34.69	86.20
	Erroneous	74.15	34.69	88.99
3	Ideal	100.00	34.69	86.20
	Erroneous	80.88	34.69	88.28
Average	**Ideal**	**100.00**	**34.69**	**86.20**
	Erroneous	**77.57**	**34.69**	**88.73**

5.3.3.2 Simulation Results Summary for Concave Polygon

Table 5.6: Simulation Result Summary for CCPP for Concave Polygon

Concave Polygon CCPP Results				
Sim. #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Ideal	96.79	57.95	143.63
	Erroneous	94.29	54.68	139.96
2	Ideal	97.10	50.01	127.57
	Erroneous	92.27	49.46	126.46
3	Ideal	98.49	62.27	155.54
	Erroneous	82.94	64.10	159.54
Average	**Ideal**	**97.46**	**56.74**	**142.24**
	Erroneous	**89.83**	**56.08**	**141.98**

Table 5.7: Simulation Result Summary for Random CPP for Concave Polygon

Concave Polygon Random CPP Results				
Sim. #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Ideal	89.90	52.34	141.54
	Erroneous	84.89	52.97	141.18
2	Ideal	92.01	53.22	142.27
	Erroneous	71.98	36.16	96.37
3	Ideal	88.08	52.00	144.90
	Erroneous	80.86	50.21	133.15
4	Ideal	89.71	51.71	141.85
	Erroneous	71.88	40.09	106.83
5	Ideal	96.33	50.29	137.88
	Erroneous	83.19	51.90	133.61
6	Ideal	94.66	53.31	142.99
	Erroneous	70.89	37.65	98.54
7	Ideal	81.25	51.68	138.04
	Erroneous	72.45	49.69	128.81
8	Ideal	78.63	52.67	147.76
	Erroneous	73.79	50.21	131.49
9	Ideal	89.12	49.87	142.69
	Erroneous	75.44	39.69	104.21
10	Ideal	73.67	52.11	149.10
	Erroneous	70.48	51.63	135.56
Average	**Ideal**	**87.34**	**51.92**	**142.90**
	Erroneous	**75.58**	**46.02**	**120.97**

Table 5.8: Simulation Result Summary for Par. Swath CPP for Concave Polygon

Concave Polygon Parallel Swath CPP Results				
Sim. #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Ideal	100.00	26.22	72.76
	Erroneous	75.41	26.22	75.95
2	Ideal	100.00	26.22	72.76
	Erroneous	74.81	26.22	78.56
3	Ideal	100.00	26.22	72.76
	Erroneous	64.77	26.22	79.11
Average	**Ideal**	**100.00**	**26.22**	**72.76**
	Erroneous	**71.66**	**26.22**	**77.87**

Table 5.9: Simulation Result Summary for Inw. Spiral CPP for Concave Polygon

Concave Polygon Inward Spiral CPP Results				
Sim. #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Ideal	100.00	32.76	89.39
	Erroneous	81.93	32.76	91.45
2	Ideal	100.00	32.76	89.39
	Erroneous	79.02	32.76	91.03
3	Ideal	100.00	32.76	89.39
	Erroneous	78.89	32.76	92.55
Average	**Ideal**	**100.00**	**32.76**	**89.39**
	Erroneous	**79.94**	**32.76**	**91.67**

5.3.3.3 Simulation Results Comparisons between CPP Methods

Table 5.2 - 5.9 presents individual simulation results. Result averages for convex and concave polygonal areas are presented in Table 5.10 and Table 5.11 consequently.

Table 5.10: Convex Polygon CPP Simulation Results Comparison

Convex Polygon CPP Simulation Results Average				
CPP Method	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
CCPP	Ideal	98.98	88.37	212.83
	Erroneous	91.75	91.78	219.06
Random CPP	Ideal	85.89	81.86	198.63
	Erroneous	80.89	78.51	185.79
Parallel Swath	Ideal	100.00	34.45	79.28
	Erroneous	58.66	34.45	87.65
Inward Spiral	Ideal	100.00	34.69	86.20
	Erroneous	77.57	34.69	88.73

Table 5.11: Concave Polygon CPP Simulation Results Comparison

Concave Polygon CPP Simulation Results Average				
CPP Method	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
CCPP	Ideal	97.46	56.53	142.24
	Erroneous	89.83	48.94	141.98
Random CPP	Ideal	87.34	51.92	142.90
	Erroneous	75.58	46.02	120.97
Parallel Swath	Ideal	100.00	26.22	72.76
	Erroneous	71.66	26.22	77.87
Inward Spiral	Ideal	100.00	32.76	89.39
	Erroneous	79.94	32.76	91.67

Simulation results showed that CCPP technique accomplished almost a complete coverage for both convex and concave polygons. CCPP actually is a complete CPP algorithm. However, for the sake of a fair comparison, path finalization threshold is kept above the cutting area of ALM in the simulation software.

Regarding erroneous behaviors, CCPP increased coverage performance of ALM for both convex and concave working areas. Those improvements are given in Table 5.12.

Table 5.12: Coverage Performance Increase by CCPP in Simulation Results

Compared Method	Polygon Type	Coverage Performance Increase by CCPP [%]
Random CPP	Convex	13.43
	Concave	18.85
Parallel Swath	Convex	56.41
	Concave	25.36
Inward Spiral	Convex	18.28
	Concave	12.37

Hence, simulations showed that CCPP covers 12 - 56 % more when compared with conventional CPP techniques.

More sophisticated comparisons made by introducing an overall performance comparison metric. As been mentioned earlier, this metric is a function of coverage percentage, travelled distance and energy consumption of ALM. The overall performance metric for result comparison is defined by Equation 5.3.

$$Performance = \%\, Coverage \times \frac{travelled\ distance}{driven\ distance} \tag{5.3}$$

where $\%\, Coverage$ represents the success of CPP in its main objective. More explicitly, it is the coverage percentage of the erroneous simulation output. The $\frac{travelled\ distance}{driven\ distance}$ term represents the useful work, which can be considered as the work

done per consumed energy. An example calculation for CCPP simulation in convex polygon is presented in Equation 5.4.

$$Performance = \frac{91.75}{100} \times \frac{91.78}{219.06} = 0.384 \quad (5.4)$$

Overall simulation performances for CPP techniques are presented in Table 5.13.

Table 5.13: Overall Simulation Performances for CPP Techniques

Compared Method	Polygon Type	Overall Performance
CCPP	Convex	0.384
	Concave	0.310
Random CPP	Convex	0.342
	Concave	0.288
Parallel Swath	Convex	0.231
	Concave	0.241
Inward Spiral	Convex	0.303
	Concave	0.286

This comparison also reveals that CCPP theoretically improves overall performance up to 66% for convex polygons and up to 28% for concave ones. When compared with random CPP, CCPP provides a 12.46% overall performance increase for specified complex geometry and 7.69% for concave one.

With the aid of simulations, CCPP technique proved its superiorities over conventional CPP methods in every aspect. More sophisticated comparisons performed with physical tests, are presented in Chapter 7.

CHAPTER 6

PRODUCTION AND INTEGRATION

6.1 Production

This section presents the production phases and final productions for the ALM.

6.1.1 Mechanical Manufacturing

The main body is manufactured from Delrin material by conventional CNC milling operation.

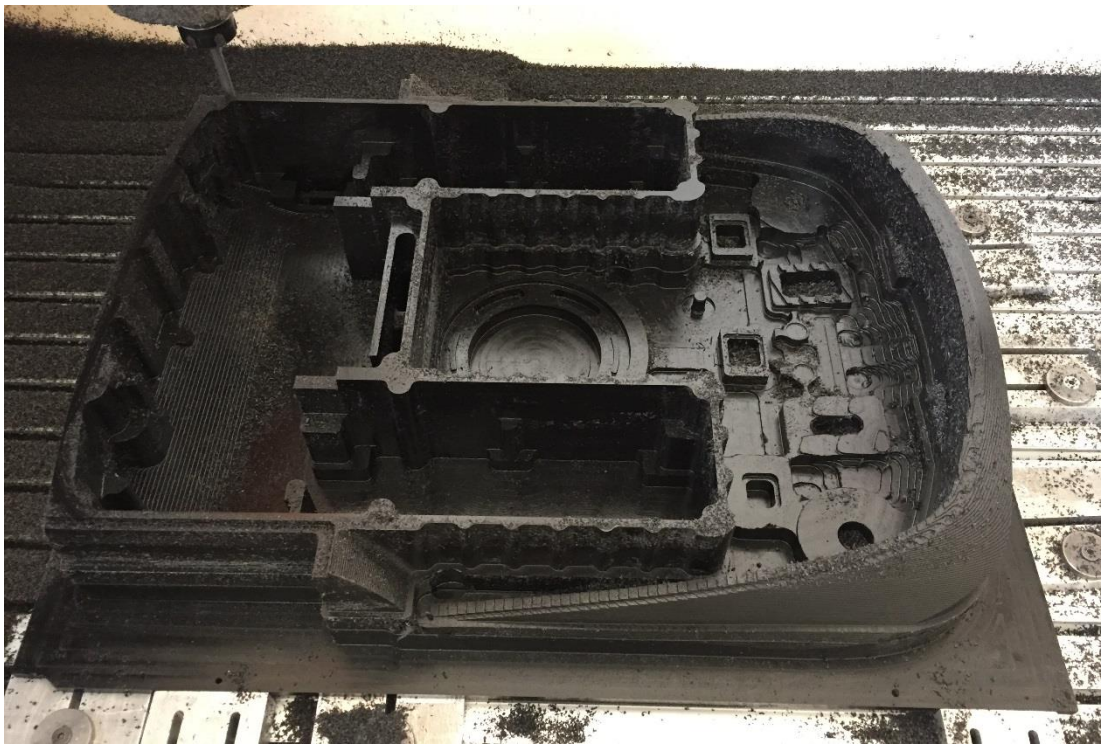


Figure 6.1: Main Body Milling Operation

The final main body is given in Figure 6.2.

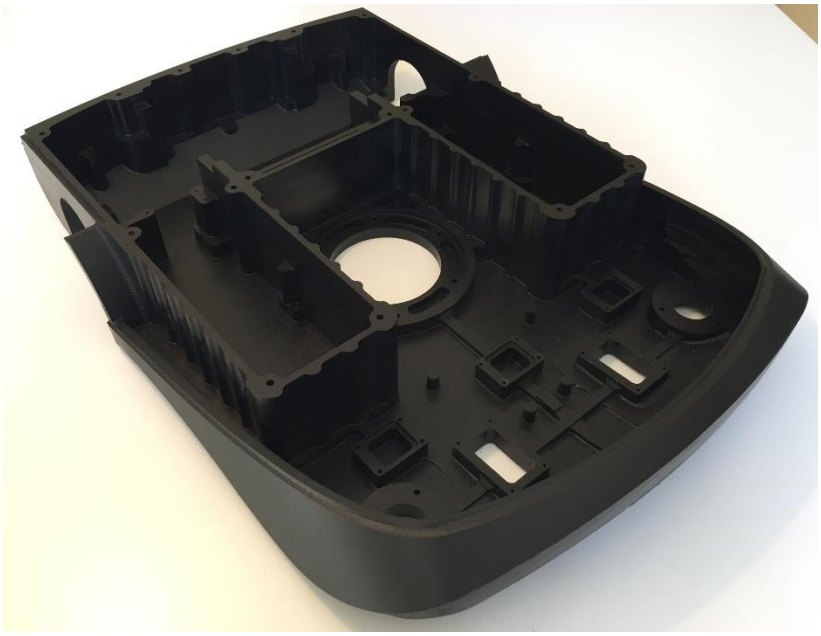


Figure 6.2: Finalized Main Body

The outer cover is determined to be manufactured by vacuum forming process. For this method, a male plywood die has been manufactured.

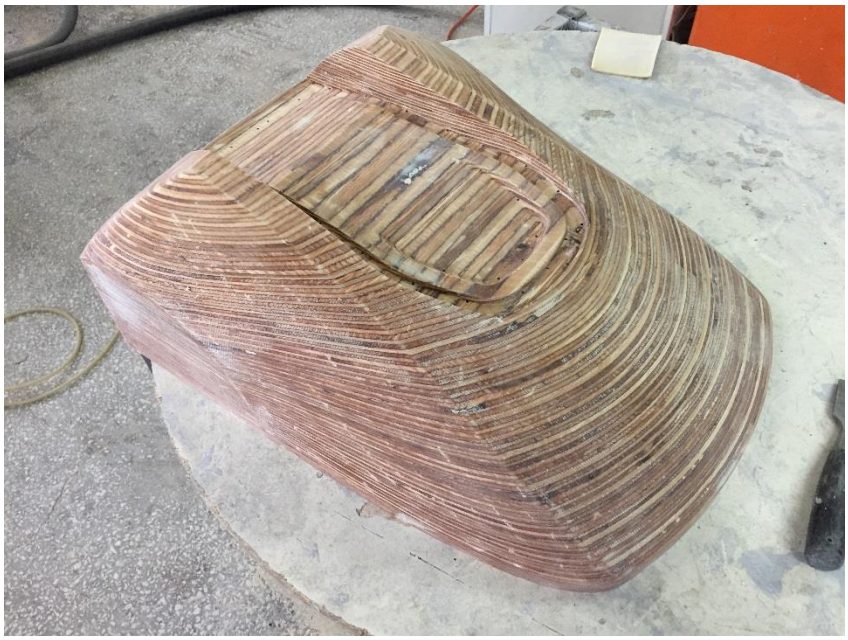
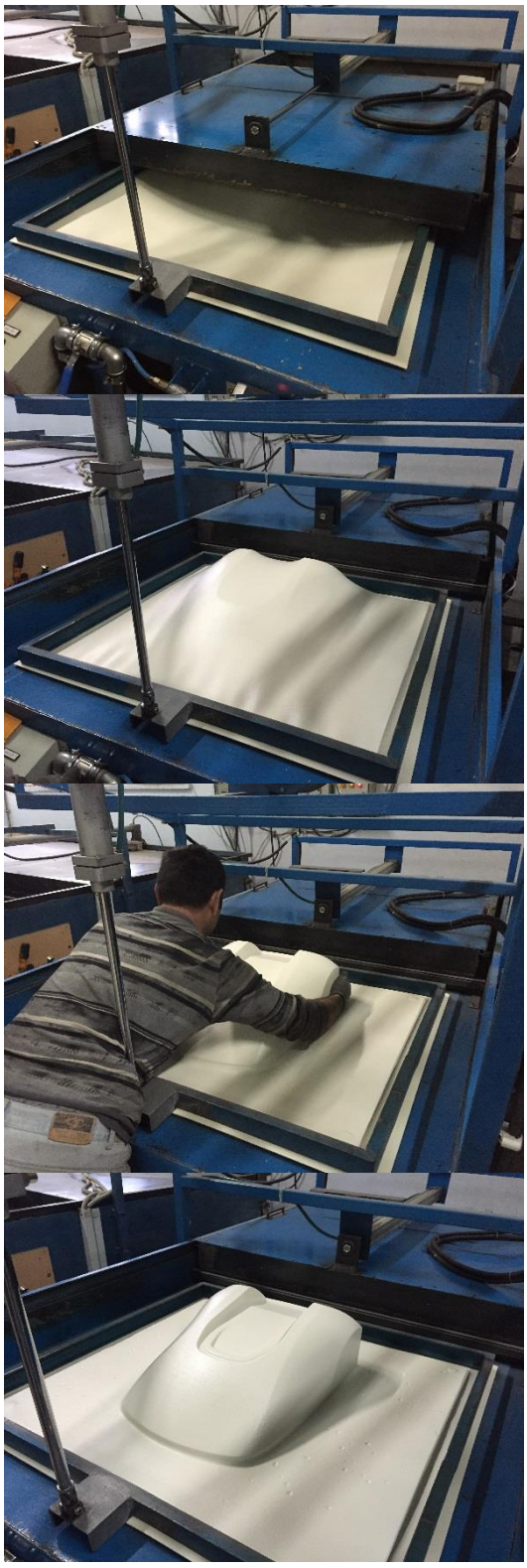


Figure 6.3: Plywood Die for Outer Cover

Stages for outer cover vacuum forming production is given in Figure 6.4.



Stage 1

Plastic sheets is heated up to a certain degree, then the male die lifted onto plastic sheet

Stage 2

Plastic takes the partial shape of the die

Stage 3

After lift operation completes, corners have been shaped manually

Stage 4

The air inside the die is vacuumed and final geometry is obtained. Afterwards, unnecessary regions have been cut out to form final cover.

Figure 6.4: Production Stages of Outer Cover

The constructive parts of ALM, drive system and mower system components are all manufactured from Delrin by turning and milling operations. Some components of ALM are presented in Figure 6.5.

Figure 6.5: Some Manufactured Components of ALM

In addition to conventional manufacturing processes, some unconventional types - 3D printing - is also used for production complex and/or functional components. Rear cap, user panel body, cutting height adjustment knob components are manufactured by 3D printing with PLA material. Printed components are presented in Figure 6.6.

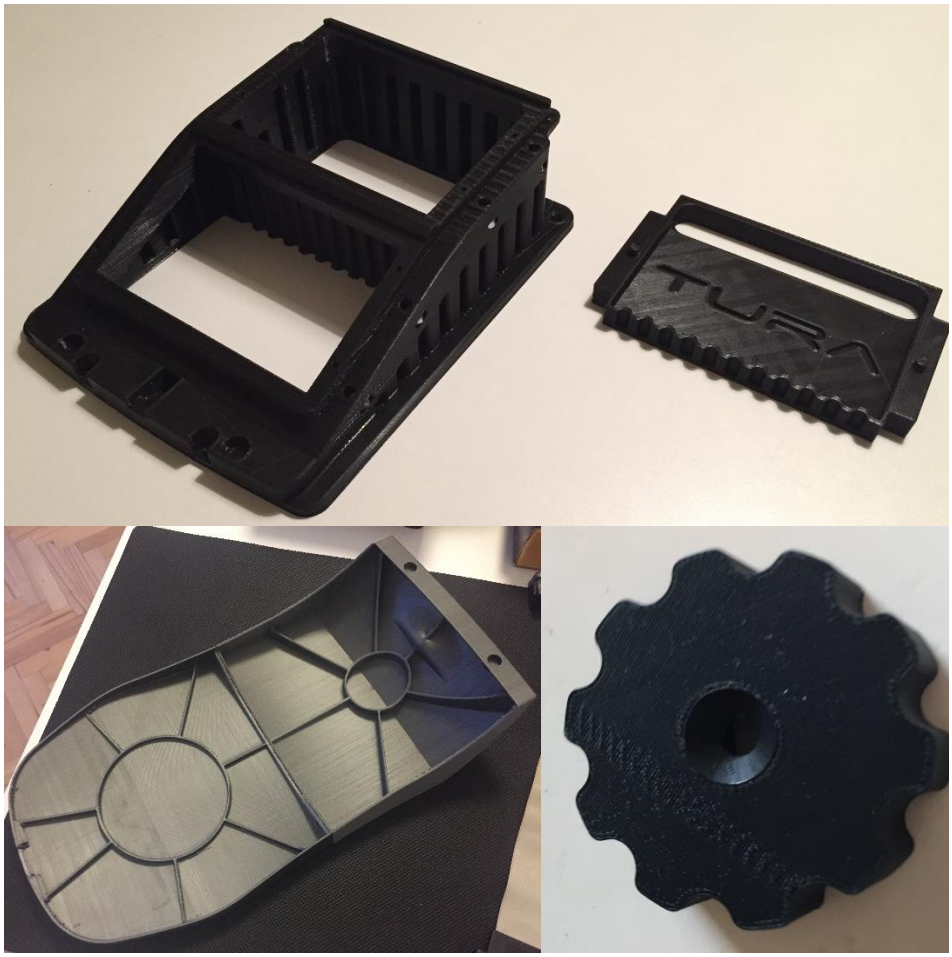


Figure 6.6: Some 3D Printed Components of ALM

6.1.2 Electronics Hardware Production

Besides mechanical manufacturing, some ALM electronics hardware are also produced in-house.

A battery stack is produced from individual LI-ION cells. Grouping, charge control and overcharge protection circuits with harnesses are produced. Figure 6.7 resembles finalized battery stack.

Figure 6.7: Final Battery Stack of ALM

Since it cannot be found as a ready-to-use product, inductive sensors are custom designed and produced. Besides, a perimeter wire to enclose ALM's working environment is also uniquely designed and manufactured. Those components are presented in Figure 6.8 - 6.9.

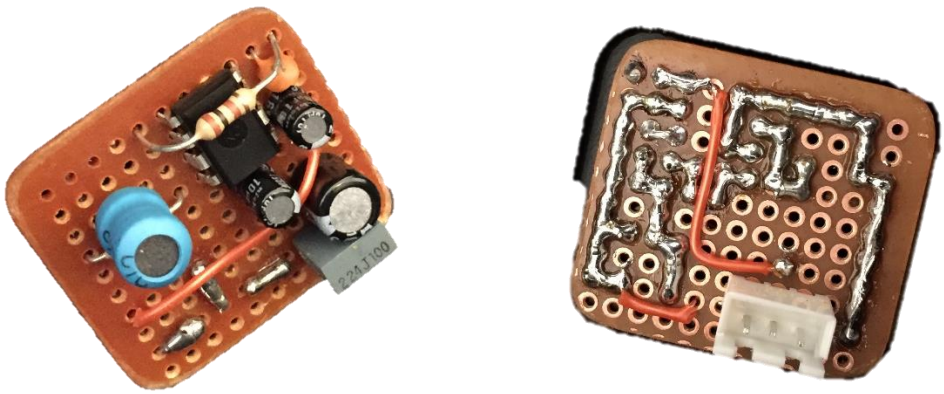


Figure 6.8: Finalized Inductive Sensors

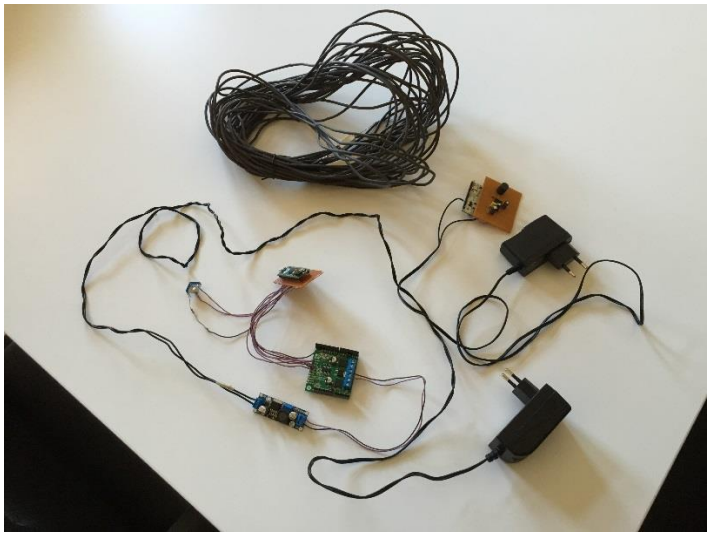


Figure 6.9: Finalized Perimeter Wire Equipment

6.2 Integration

In the system integration phase, major subcomponents of ALM is assembled first. For drive system, ball bearings, drive motor, belt-pulley mechanism and drive shaft are mounted on the drive system body and drive system assembly is finalized.

Figure 6.10: Drive System Integration

Ball bearings, drive motor, belt-pulley mechanism and drive shaft are mounted on the drive system body and drive system assembly is finalized.

For the mowing system, mower motor, motion plate, motion screw with cutting height adjustment mechanism components are assembled onto mowing system body. Figure 6.11 resembles the integration of mowing system.

Figure 6.11: Mowing System Integration

After completion of drive and mowing subsystems, user panel, IR sensors, charge pins, castor wheels and their suspension mechanisms are assembled.

Finalized integration of the ALM is revealed in Figures 6.12 - 6.14.

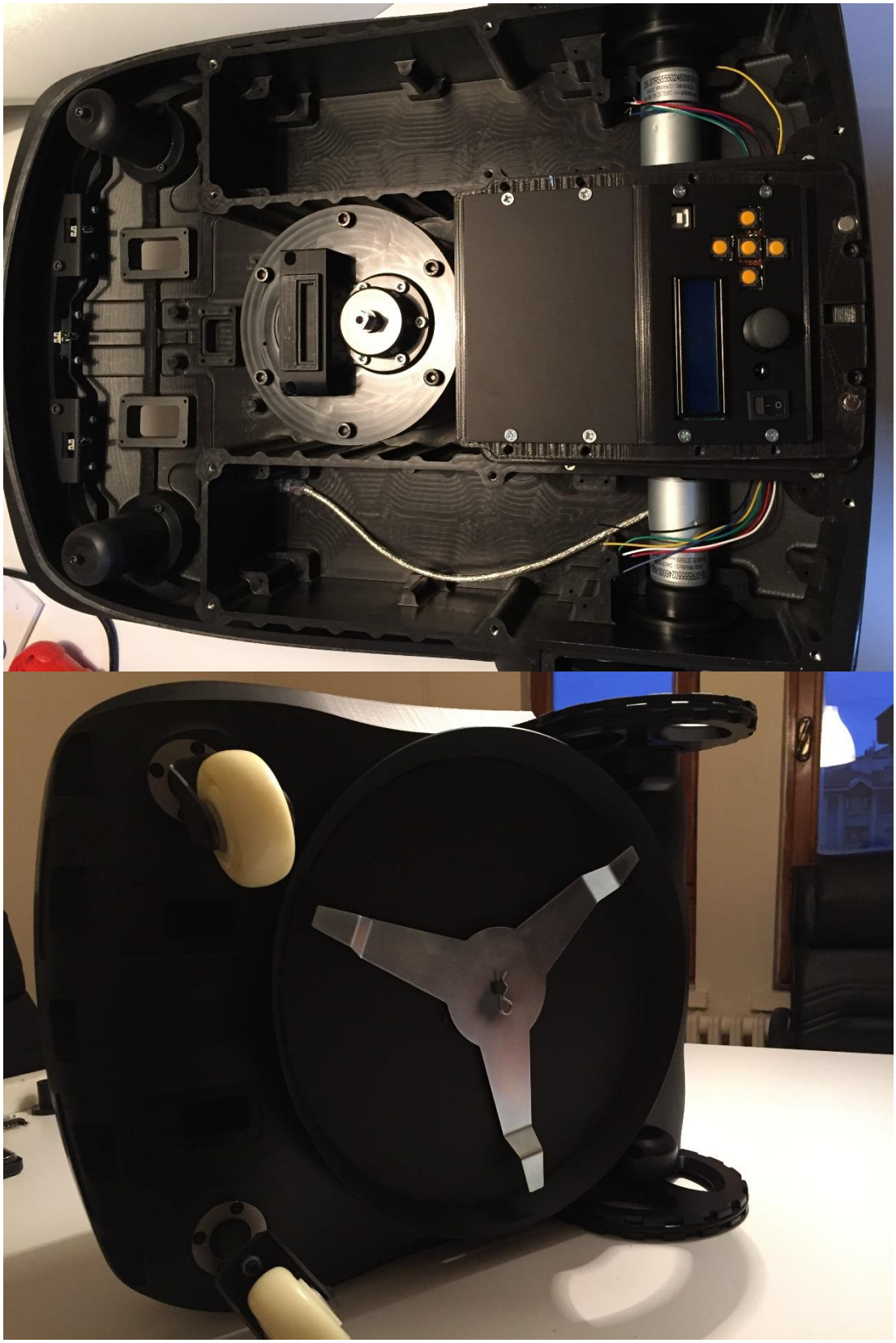


Figure 6.12: Finalized Main Body Integration

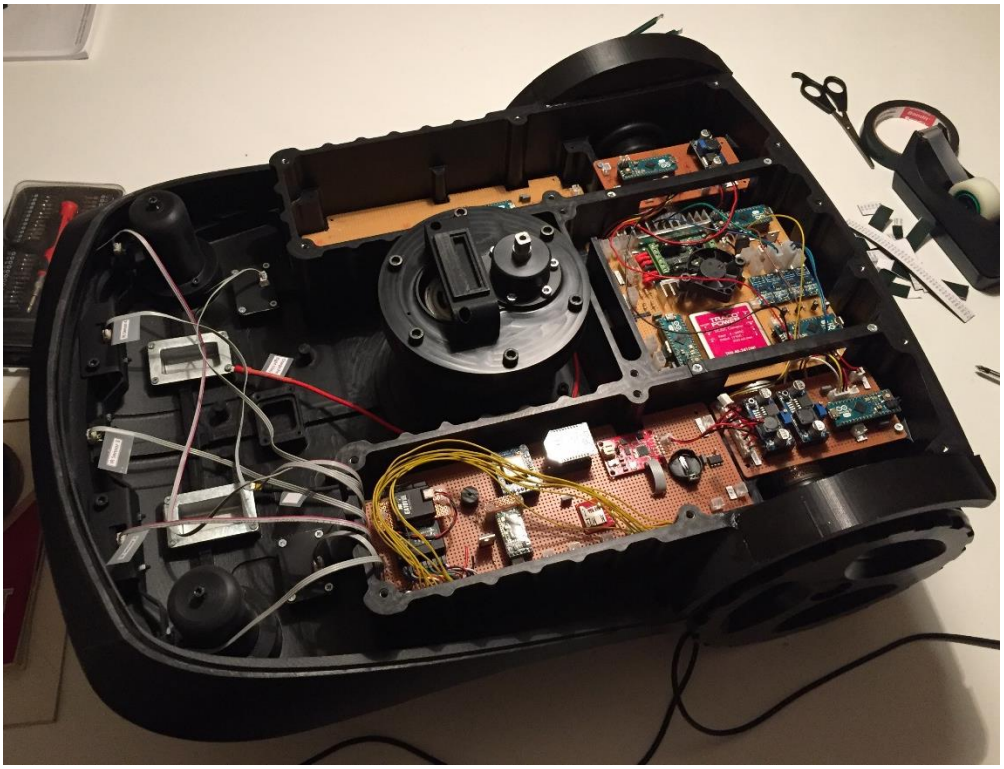


Figure 6.13: Finalized Mechanical-Electronics Integration

Figure 6.14: Finalized System Integration

CHAPTER 7

TESTING AND VALIDATION

This chapter presents the comprehensive physical testing of ALM. In this context, outdoor tests are performed to reveal actual performance of developed CCPP technique and its pros and cons over conventional CPP methods. Testing environment, test setup and test results are revealed in the following sections.

7.1 Test Environment and Setup

A generic test setup has been built in order to compare and contrast ALM's performance on different polygonal geometries with varying CPP techniques. A rectangular test field of 10 m^2 (4.0 m x 2.5 m) has been built for convex polygon testing and one-fourth of this area is dismissed for testing ALM over concave polygon.

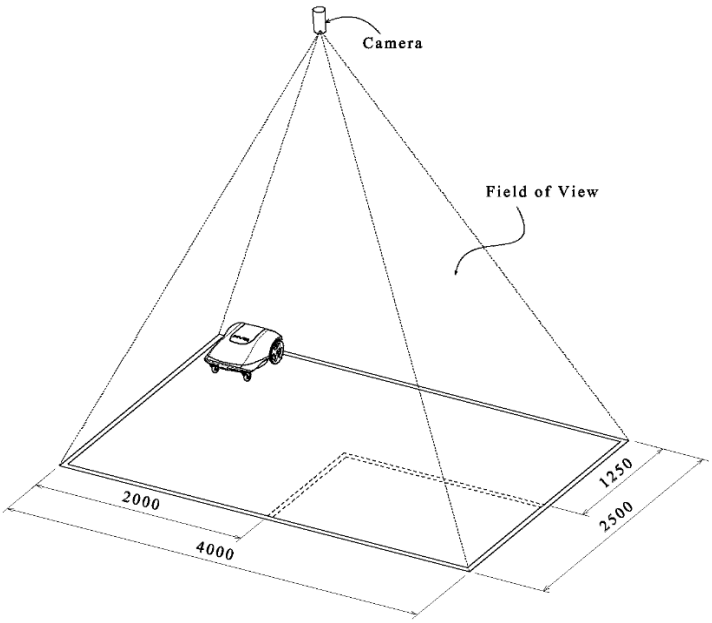


Figure 7.1: Test Setup Schematic

Figure 7.1 presents the test setup schematic. The outdoor tests are performed over lawn with uneven terrain conditions. A camera has been placed on the top of the centroid of the working area. Figure 7.2 shows a sample camera view of the test environment.

Figure 7.2: Sample Camera View of Outdoor Test

7.2 Test Scenarios and Aspects

Same scenarios with simulations are tested on outdoor environment. Three sets of tests for CCPP technique, ten sets for random CPP method and three sets for Parallel Swath and Inward Spiral CPP techniques are performed.

Test scenarios are summarized on Table 7.1.

Table 7.1: Test Scenarios

Convex Polygon	CCPP	+	+
	Random CPP	+	+
	Parallel Swath CPP	+	+
	Inward Spiral CPP	+	+

Concave Polygon	CCPP	+	+
	Random CPP	+	+
	Parallel Swath CPP	+	+
	Inward Spiral CPP	+	+

Outdoor tests aimed to reveal actual performance of ALM and developed CCPP technique. For this purpose, performance metrics like travelled distance, energy consumption and coverage percentage of different CPP methods are obtained in each test. Besides, deviation of actual path from ideal is also evaluated.

7.3 Data Acquisition and Post-Processing

Total of 38 tests are performed for outdoor environment. Within all of these tests, ALM motion is recorded by the camera. As optical distortion is inevitable for the camera and our test setup, post-processing is made for compensating the distortion with a commercial software.

A circular marker has been placed on the top of the ALM. With the aid of this marker, trajectories and overall paths in tests are obtained over recordings. When path is obtained, it is stroked by blade width in order to visualize coverage percentage of the robot.

7.4 Test Results

7.4.1 Convex Polygon Test Results

7.4.1.1 Convex Polygon CCPP Test Results

Figure 7.3 represents the results of a sample of CCPP path with an initial starting point of (3.0 m, 0 m).

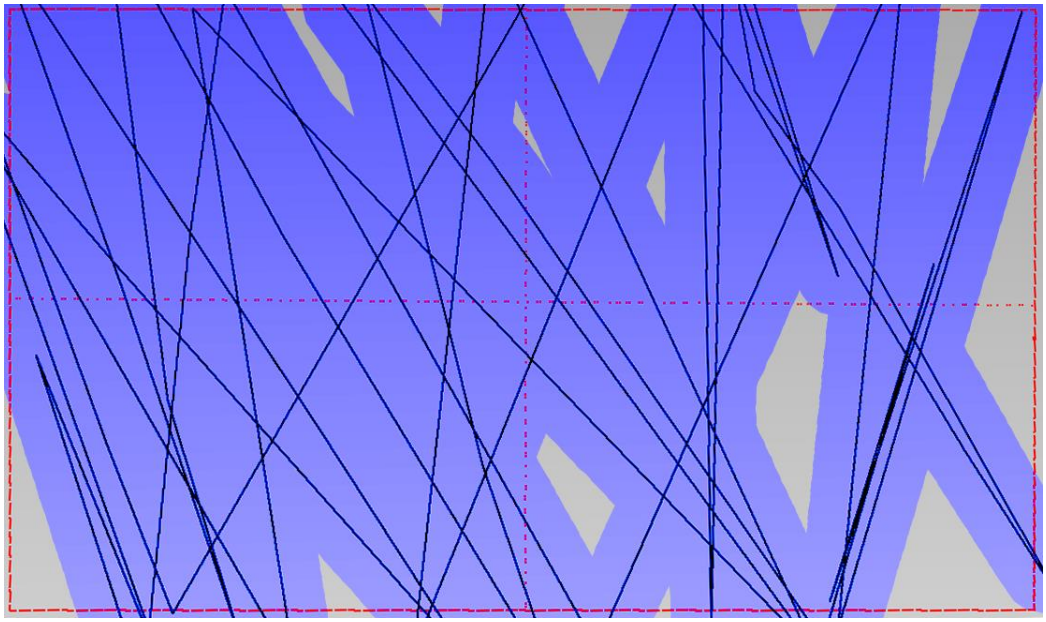


Figure 7.3: Convex Polygon CCPP Actual Path and Coverage

7.4.1.2 Convex Polygon Random CPP Test Results

Figure 7.4 represents the results of a sample of random CPP path with an initial starting point of (1.0 m, 0 m).

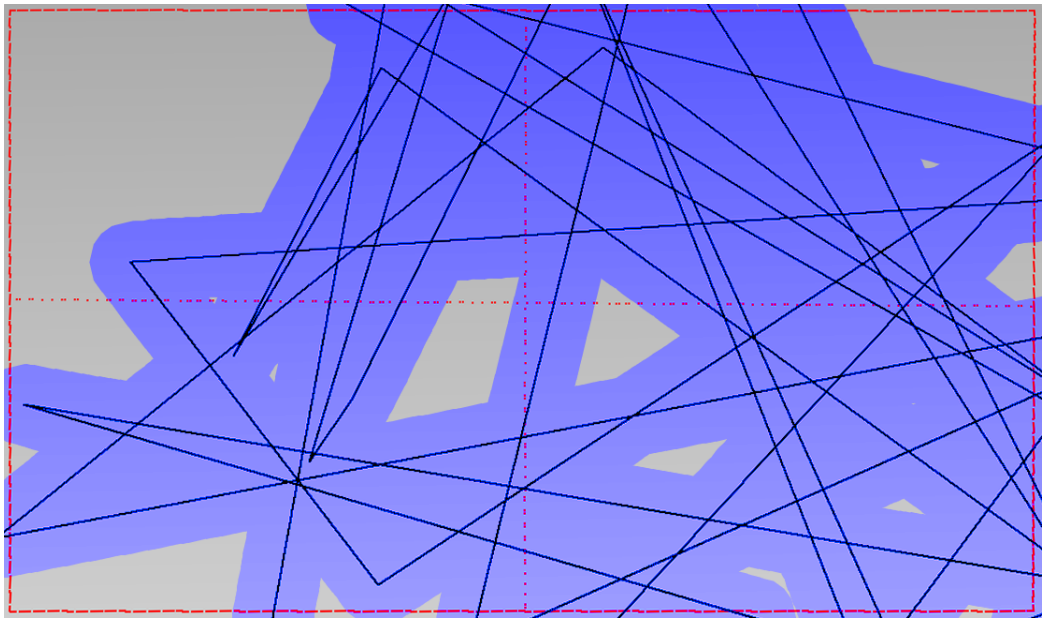


Figure 7.4: Convex Polygon Random CPP Actual Path and Coverage

7.4.1.3 Convex Polygon Parallel Swath CPP Test Results

Figure 7.5 represents the results of a sample of parallel swath CPP path with an initial starting point of (4.0 m, 0 m).

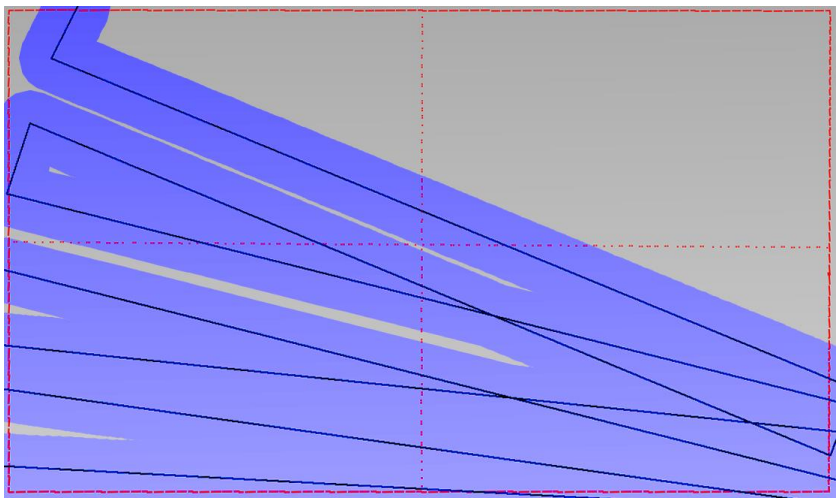


Figure 7.5: Convex Polygon Parallel Swath CPP Actual Path and Coverage

7.4.1.4 Convex Polygon Inward Spiral CPP Test Results

Figure 7.6 represents the results of a sample of inward spiral CPP path with an initial starting point of (4.0 m, 0 m).

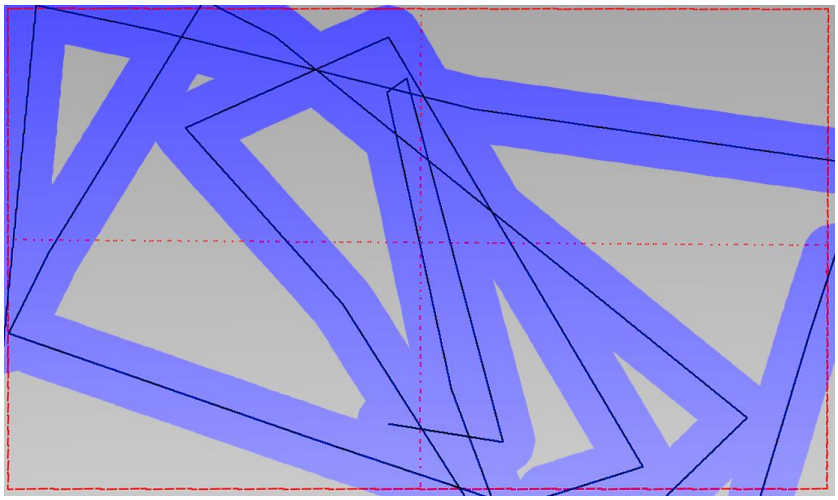


Figure 7.6: Convex Polygon Inward Spiral CPP Actual Path and Coverage

7.4.2 Concave Polygon Test Results

7.4.2.1 Concave Polygon CCPP Test Results

Figure 7.7 represents the results of a sample of CCPP path with an initial starting point of (3.0 m, 0 m).

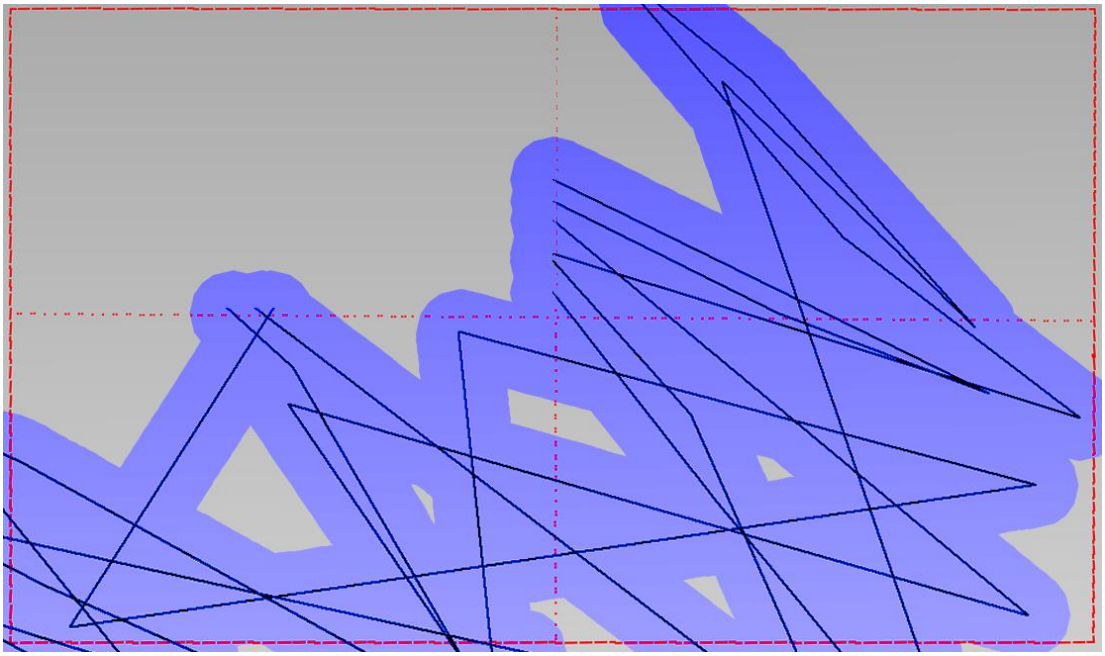


Figure 7.7: Concave Polygon CCPP Actual Path and Coverage

7.4.2.2 Concave Polygon Random CPP Test Results

Figure 7.8 represents the results of a sample of random CPP path with an initial starting point of (3.0 m, 0 m).

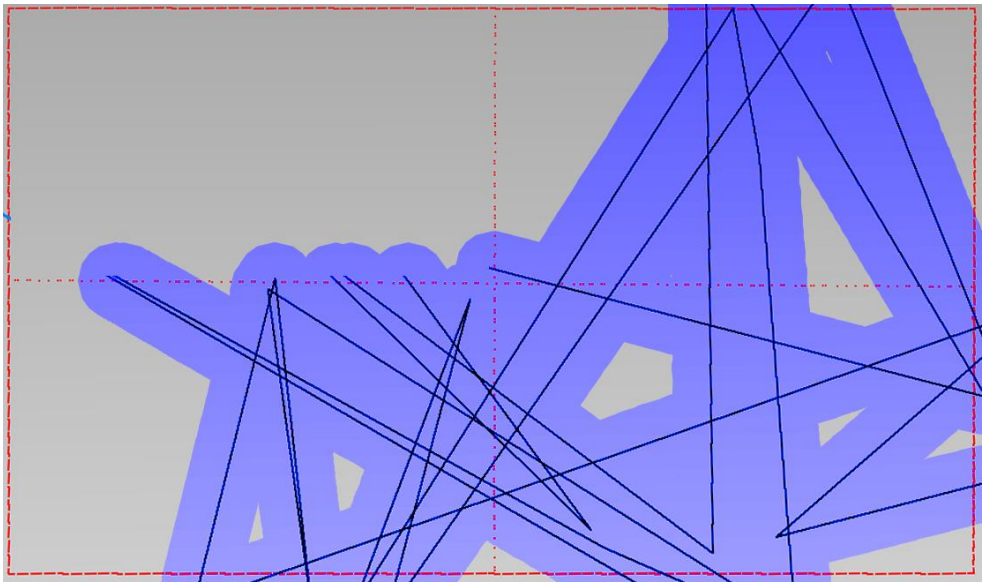


Figure 7.8: Concave Polygon Random CPP Actual Path and Coverage

7.4.2.3 Concave Polygon Parallel Swath CPP Test Results

Figure 7.9 represents the results of a sample of parallel swath CPP path with an initial starting point of (4.0 m, 0 m).

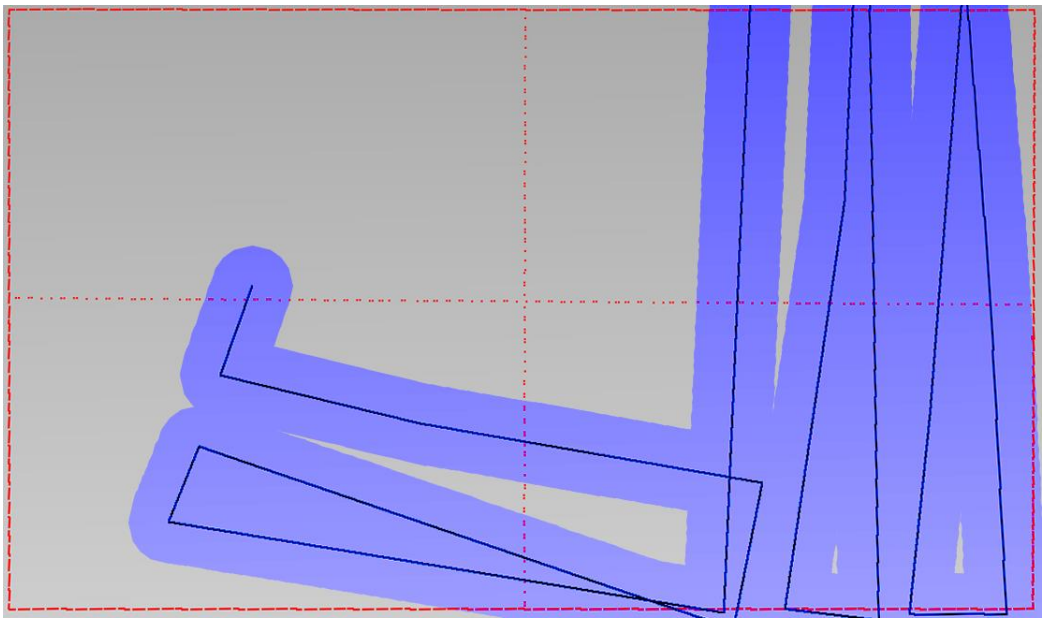


Figure 7.9: Concave Polygon Parallel Swath CPP Actual Path and Coverage

7.4.2.4 Concave Polygon Inward Spiral CPP Test Results

Figure 7.10 represents the results of a sample of inward spiral CPP path with an initial starting point of (4.0 m, 0 m).

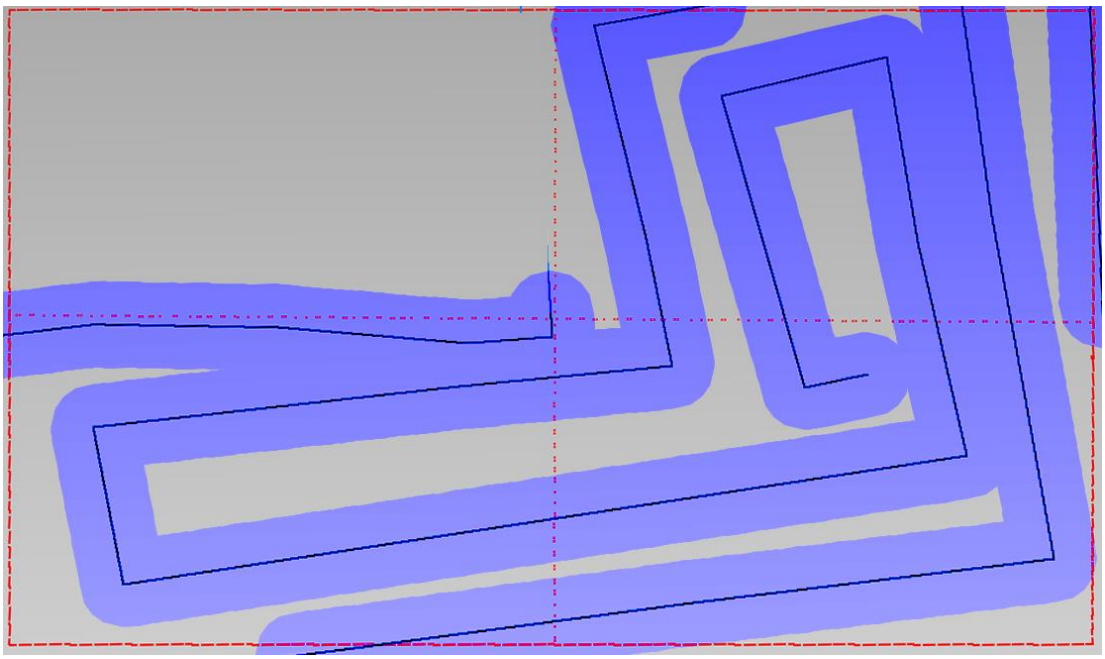


Figure 7.10: Concave Polygon Inward Spiral CPP Actual Path and Coverage

7.4.3 Test Results Summary

Same techniques with simulations are followed for physical test result evaluations. Movie recordings are processed to determine ALM's actual path. This path is used to determine the travelled distance. Afterwards, it is stroked by mowing blade width and coverage percentage is calculated. The number of encoder counts (and Driven Distance) remains same with ideal behavior simulations, since they represents actual commands.

7.4.3.1 Test Results Summary for Convex Polygon

Table 7.2 Test Result Summary for CCPP for Convex Polygon

Convex Polygon CCPP Results				
Test #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Actual	84.56	94.20	243.89
2	Actual	77.08	80.91	201.67
3	Actual	86.72	89.24	192.93
Average	**Actual**	**82.78**	**88.12**	**212.83**

Table 7.3: Test Result Summary for Random CPP for Convex Polygon

Convex Polygon Random CPP Results				
Test #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Actual	80.58	79.14	196.70
2	Actual	79.65	75.38	197.26
3	Actual	71.52	80.49	201.93
4	Actual	47.26	73.48	193.67
5	Actual	65.42	78.45	212.22
6	Actual	71.49	81.26	209.42
7	Actual	73.61	74.52	205.42
8	Actual	56.29	72.13	187.95
9	Actual	61.78	75.78	187.61
10	Actual	75.63	78.19	194.14
Average	**Actual**	**68.32**	**76.88**	**198.63**

Table 7.4: Test Result Summary for Par. Swath CPP for Convex Polygon

Convex Polygon Parallel Swath CPP Results				
Test #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Actual	70.45	32.95	79.28
2	Actual	54.06	30.57	79.28
3	Actual	56.85	31.87	79.28
Average	**Actual**	**60.43**	**31.79**	**79.28**

Table 7.5: Test Result Summary for Inw. Spiral CPP for Convex Polygon

Convex Polygon Inward Spiral CPP Results				
Test #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Actual	75.84	32.50	86.20
2	Actual	49.12	30.05	86.20
3	Actual	77.68	31.86	86.20
Average	**Actual**	**67.55**	**31.47**	**86.20**

7.4.3.2 Test Results Summary for Concave Polygon

Table 7.6: Test Result Summary for CCPP for Concave Polygon

Concave Polygon CCPP Results				
Test #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Actual	85.66	55.41	143.63
2	Actual	66.42	49.24	127.57
3	Actual	88.52	53.47	155.54
Average	**Actual**	**80.20**	**52.70**	**142.24**

Table 7.7: Test Result Summary for Random CPP for Concave Polygon

Concave Polygon Random CPP Results				
Test #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Actual	63.57	50.76	141.54
2	Actual	60.87	51.28	142.27
3	Actual	48.26	48.00	144.90
4	Actual	70.13	51.63	141.85
5	Actual	64.84	50.19	137.88
6	Actual	62.49	51.35	142.99
7	Actual	72.15	50.74	138.04
8	Actual	56.49	47.98	147.76
9	Actual	60.55	51.82	142.69
10	Actual	61.84	50.67	149.10
Average	**Actual**	**62.12**	**50.44**	**142.90**

Table 7.8: Test Result Summary for Par. Swath CPP for Concave Polygon

Concave Polygon Parallel Swath CPP Results				
Test #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Actual	55.88	24.57	72.76
2	Actual	65.17	25.96	72.76
3	Actual	63.53	24.64	72.76
Average	**Actual**	**61.52**	**25.05**	**72.76**

Table 7.9: Test Result Summary for Inw. Spiral CPP for Concave Polygon

Concave Polygon Inward Spiral CPP Results				
Test #	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
1	Actual	65.17	31.48	89.39
2	Actual	70.48	32.13	89.39
3	Actual	67.33	31.56	89.39
Average	**Actual**	**67.66**	**31.72**	**89.39**

7.4.3.3 Test Results Comparisons between CPP Methods

Table 7.2 - 7.9 presents individual test results. Result averages for convex and concave polygonal areas are presented in Table 7.10 and Table 7.11.

Table 7.10: Convex Polygon CPP Test Results Comparison

Convex Polygon CPP Test Results Average				
CPP Method	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
CCPP	Actual	82.78	88.12	212.83
Random CPP	Actual	68.32	76.88	198.63
Parallel Swath	Actual	60.43	31.79	79.28
Inward Spiral	Actual	67.55	31.47	86.20

Table 7.11: Concave Polygon CPP Test Results Comparison

Concave Polygon CPP Test Results Average				
CPP Method	Behavior	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
CCPP	Actual	80.20	52.70	142.24
Random CPP	Actual	62.12	50.44	142.90
Parallel Swath	Actual	61.52	25.05	72.76
Inward Spiral	Actual	67.66	31.72	89.39

Similarly with simulation results, coverage percentage increase by CCPP is presented in Table 7.12.

Table 7.12: Coverage Performance Increase by CCPP in Test Results

Compared Method	Polygon Type	Coverage Performance Increase by CCPP [%]
Random CPP	Convex	21.17
	Concave	29.10
Parallel Swath	Convex	36.98
	Concave	30.36
Inward Spiral	Convex	22.55
	Concave	18.53

Physical tests showed that CCPP covers 18 - 37 % more when compared with conventional CPP techniques.

Using Equation 5.3, overall physical test performances of compared CPP techniques are presented in Table 7.13.

Physical test comparisons reveals that CCPP actually improves overall performance up to 41% for convex polygons and up to 40% for concave ones. When compared with random CPP, CCPP provides a 29.61% overall performance increase for complex geometries and 35.52% for concave ones.

Table 7.13: Overall Test Performances for CPP Techniques

Compared Method	Polygon Type	Overall Performance
CCPP	Convex	0.343
	Concave	0.297
Random CPP	Convex	0.264
	Concave	0.219
Parallel Swath	Convex	0.242
	Concave	0.212
Inward Spiral	Convex	0.247
	Concave	0.240

In addition to simulation results, CCPP elapses physical tests in the first place. Detailed comparisons and summary of test results accompanied with simulation results are presented in the next section.

7.5 Conclusion

Section 5.3 presents simulation results and Section 7.4 reveals physical test results in detail. In both theoretical and physical comparisons, CCPP technique has proven its benefits over conventional CPP methods.

However, it must be taken into account that overall performance metrics for simulation results are derived over performances of erroneous coverage paths. Therefore the accuracy of introduced error model is important.

It can be seen in Figure 7.11 that erroneous trajectories in different CPP implementations are closely consistent with the actual path of the robot. The figure represents some sample comparisons for CCPP, parallel swath, inward spiral and random CPP methods.

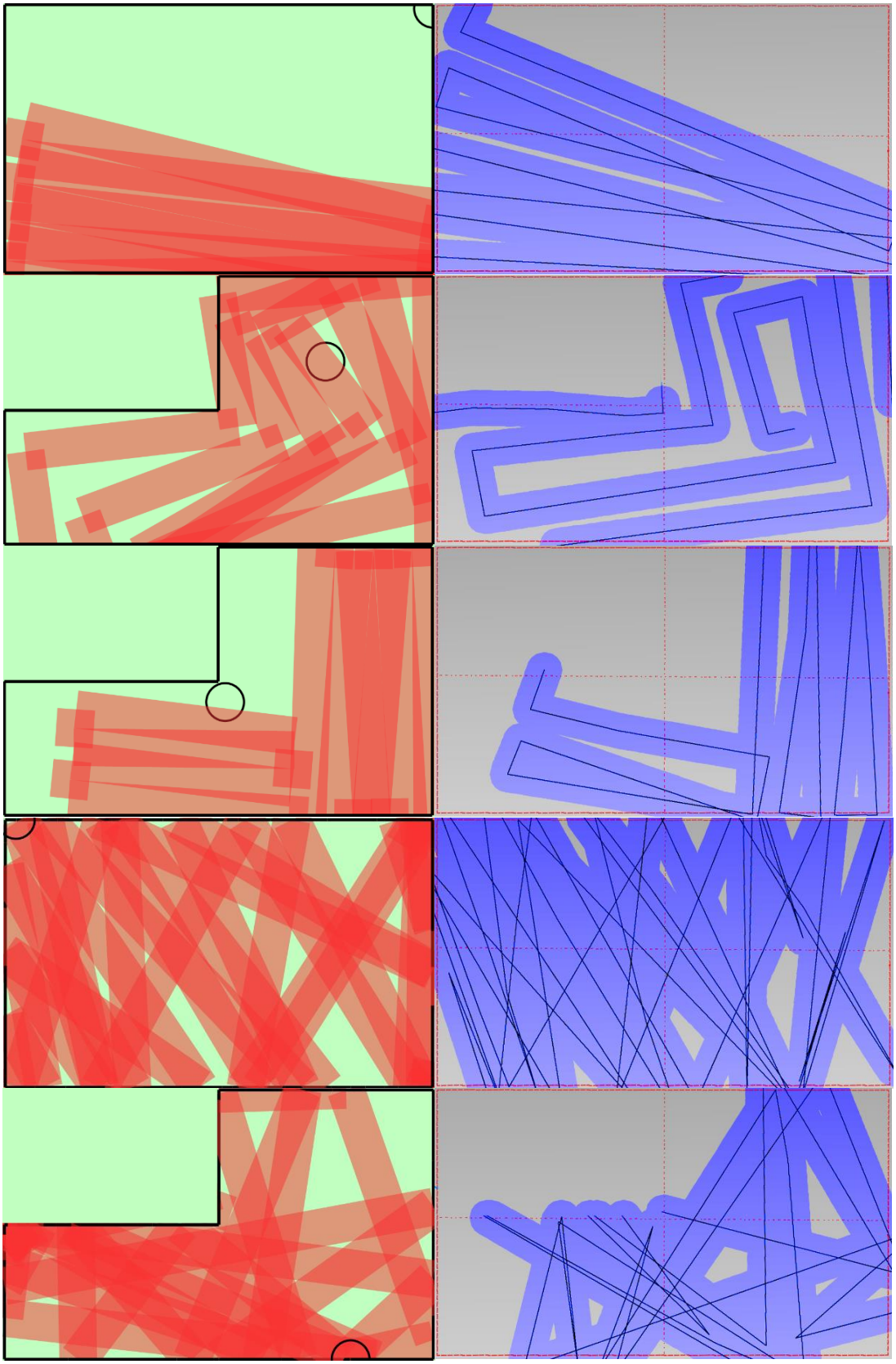


Figure 7.11: Erroneous (Left) and Actual Path (Right) Comparisons

Table 7.14 introduces the overall simulation and test result averages. In the table, simulation results represents the performance for erroneous paths and test results indicates actual output of ALM.

Table 7.14: Comparisons of Simulation and Test Results for CPP Methods

CPP Method	Polygon Type	Result Average	Coverage Percentage [%]	Distance Travelled [m]	Driven Distance [m]
CCPP	Convex	Simulation	91.75	91.78	219.06
		Test	82.78	88.12	212.83
	Concave	Simulation	89.83	48.94	141.98
		Test	80.20	52.70	142.24
Random CPP	Convex	Simulation	80.89	78.51	185.79
		Test	68.32	76.88	198.63
	Concave	Simulation	75.58	46.02	120.97
		Test	62.12	50.44	142.90
Parallel Swath	Convex	Simulation	58.66	34.45	87.65
		Test	60.43	31.79	79.28
	Concave	Simulation	71.66	26.22	77.87
		Test	61.52	25.05	72.76
Inward Spiral	Convex	Simulation	77.57	34.69	88.73
		Test	67.55	31.47	86.20
	Concave	Simulation	79.94	32.76	91.67
		Test	67.66	31.72	89.39

Table 7.14 reveals that the deviation between simulation and test results lies between 10% in most of the cases. When this information is evaluated with erroneous - actual path comparisons (Figure 7.11), it can be said that the introduced error model captures physical behavior of the ALM quite well. It must be remarked that creation of a "perfect" error model is impossible for such kind of variable, unstructured outdoor environment. Actual behavior of any robot will inevitably change over every garden or on every lawn.

Finally, 10 % deviation for corresponding motion paths is found sufficient for the error model. Therefore the introduced error model is validated which leads the simulation results to be validated also.

Table 7.15: Performance Increased by CCPP in Test and Simulation Results

CPP Method	Polygon Type	Source	Coverage Percentage Increase [%]	Overall Performance Increase [%]
Random CPP	Convex	Simulation	13.43	12.46
		Test	21.17	29.61
	Concave	Simulation	18.85	7.69
		Test	29.10	35.52
Parallel Swath	Convex	Simulation	56.41	66.73
		Test	36.98	41.44
	Concave	Simulation	25.36	28.33
		Test	30.36	40.29
Inward Spiral	Convex	Simulation	18.28	26.75
		Test	22.55	38.98
	Concave	Simulation	12.37	8.39
		Test	18.53	23.76

Table 7.15 introduces coverage percentage and overall performance addition of CCPP to conventional CPP techniques.

Deterministic CPP methods like parallel swath and inward spiral took last places when overall performances considered. As highlighted throughout this work, instantaneous localization is highly erroneous for those types of deterministic algorithms since they utilize no external references for navigation. As an internal comparison between these two methods, inward spiral succeeds as expected. Main reasons are; it rotates to same direction between trajectories and the number of rotations are less than parallel swath. However, in parallel swath, every two rotation direction is the opposite of the previous couple. This behavior highly accumulates error and ends up with a relatively poor performance.

Random CPP technique ended up with better performance when compared with previously mentioned deterministic methods. This also an expected situation. These two methods may accomplish better performances than random CPP in structured, smooth environments but as the complexity of environment increases, randomization becomes advantageous as seen. This is the main reason that nearly all of the commercial domestic mobile robots utilizes random CPP techniques.

Physical test and simulation result clearly showed that CCPP can definitely be a successor to random CPP technique. Regarding simulation and test results, CCPP increased coverage percentage more than 17 % (average) for convex polygon and more than 23 % (average) for concave one, when compared with random CPP. The overall performance has also increased by an average of 21 % for both convex and concave polygons.

CHAPTER 8

CONCLUSION

8.1 Conclusion

In this study, a state-of-art coverage path planning technique - CCPP is introduced. For its development and virtual validation, a simulator has been developed from scratch. In the simulator, various conditional behaviors and their consequences are investigated to form up the final shape of the technique. Besides all, a unique autonomous lawn mower is designed and produced regarding commercial market demands, where it became a ready-to-sell product at the end of this work.

CCPP is a deterministic CPP technique, which aims a complete coverage of its working area. The major motivation in creation of CCPP is to unite advantages of different CPP methods.

For a complete coverage, determinism is essential. CCPP is designed to operate same in every operation for a specifically defined working environment. However, drawing similar patterns on every working area is highly vulnerable to external effects, therefore it is inaccurate and inefficient. CCPP solves this problem with its capability to differentiate its overall path by decomposition of environment.

As environmental decomposition considered, many CPP techniques work with a similar manner - they use decompositions for partial area coverages. In this partial coverage path plans, they utilize simple motions such as back and forth as conventional methods do. As a result, even if they plan their overall paths prior to their operations, they suffer from same problems with custom conventional deterministic methods (parallel swath, spiral CPP, etc.) have.

When real-time environmental decompositions (by map building) are desired to be used, this concept additionally requires expensive sensory or computation hardware, which is not suitable for a commercial product.

Regarding those disadvantages, random CPP techniques usually provides the best performance-per-cost between conventional CPP methods. Unfortunately, due to its random behavior nature, this technique is incapable of providing a complete coverage. As also seen in our test results, performance of random CPP significantly decreased for concave polygonal working areas.

What CCPP mainly does is to utilize border information for trajectory generation. It is derived over a simple idea that every working environment for an ALM, must have borders.

CCPP creates its trajectories from border to border. Only straight line trajectories are used in CCPP for to reduce localization errors. These behaviors are actually inspired from random CPP methods. Contrary to random CPP methods, CCPP actually uses this border information for cellular decomposition and trajectory generation, whereas random CPP methods treats borders only as walls that must not to be crossed. Moreover, CCPP knows when job is completed, whereas random CPP did not have such kind of information.

CCPP calculates decompositions and generates its new trajectory when it reaches to a border. By doing this, it eases the duty of processors for mathematical computations. Having this specialty, CCPP requires less computational power or hardware then conventional cellular decomposition methods (trapezoidal decomposition, etc.).

CCPP cares about energy efficiency. The proposed cost function and conditional selection of unmowed regions decreases the travelled distance and reduces overpasses.

As a summary, CCPP is a deterministic, energy saving and highly adaptable complete CPP technique, which combines the unbounded acts of randomized algorithms and well organized behaviors of deterministic CPP techniques. It demands no expensive sensor or computation hardware.

Both simulations and tests revealed that CCPP technique is superior over conventional CPP methods. CCPP increased overall performance of ALM more than 20 % (average) when compared with the most significant rival - random CPP.

Physical test results have confirmed and validated simulations and the overall concept. As a result, CCPP is found highly beneficial for autonomous lawn mowing. When the complexity of the working environments compared, it can be concluded that CCPP is not solely an outdoor technique, it can also be a serious alternative to indoor WMRs (such as vacuum cleaners, etc.) when coverage path planning concerned.

During the physical tests, it is observed that the castor wheel assembly design of ALM introduces an additional non-systematic error to dead reckoning operation. Therefore their design is intended to be revised as a future work. Moreover, the obstacle avoidance algorithm of the ALM is found suitable for static or dynamic obstacle encounters; however it is a little slow and reluctant. This algorithm has been planned to be replaced with a faster, adaptive obstacle avoidance algorithm in a close future.

In sensor fusion, some simplifications were made in Kalman Filter in order to avoid losing the focus on the main scope equations ((w), (Q), (Z) and (R) matrices are taken as zero). As a further work however, all these matrices are intended to be characterized and adjusted depending on the reliable physical test results. Additional fine-tuning about sensor fusion would also be made by adjusting the initial values in the Process Covariance Matrix (P) in Kalman Filter equations.

Lastly, a learning algorithm, which utilizes inductive sensors and perimeter wire, has been decided to be implemented to ALM for environmental border identification and recognition.

REFERENCES

[1] A. Roberts, The History of Science Fiction, Palgrave Macmillan, 2006.

[2] "The Old Robots," [Online]. Available: http://www.theoldrobots.com/ElmerElsie.html. [Accessed March 2016].

[3] R. Siegwart, R. Nourbakhsh, D. Scaramuzza, Introduction to Autonomous Mobile Robots, MIT Press, 2004.

[4] "LAAS," [Online]. Available: http://homepages.laas.fr/. [Accessed March 2016].

[5] M. G. Spyridon, M. Eleftheria, "Classification of Domestic Robots," in *International Virtual Coference*, 2012.

[6] "LEGO," [Online]. Available: http://www.lego.com/. [Accessed April 2016].

[7] "Aldebaran Robotics," [Online]. Available: https://www.aldebaran.com. [Accessed April 2016].

[8] "WowWee," [Online]. Available: http://wowwee.com/products/robots. [Accessed April 2016].

[9] "MantaroBot," [Online]. Available: http://www.mantarobot.com/. [Accessed April 2016].

[10] "IEEE Spectrum," [Online]. Available: http://spectrum.ieee.org/automaton/robotics/home-robots/care-o-bot-4-mobile-manipulator. [Accessed April 2016].

[11] B. Gates, "A Robot in Every Home," Scientific American, 2007.

[12] "World Robotics 2014," International Federaion of Robotics, 2014.

[13] "Samsung," [Online]. Available: http://www.samsung.com/. [Accessed April 2016].

[14] "Robomower," [Online]. Available: http://robomow.com/. [Accessed April 2016].

[15] H. Choset, "Coverage for Robotics - A Survey of Recent Results," *Annals of Mathematics and Artificial Intelligence,* no. 31, pp. 113-126, 2001.

[16] J. Borenstein, H. R. Everett, L. Feng, "Where Am I?" Sensors and Methods for Mobile Robot Positioning, University of Michigan, 1996.

[17] J. Borenstein, L. Feng, "Measurement and Correction of Systematic Odometry Errors in Mobile Robots," *IEEE Transactions on Robotics and Automation,* vol. 12, no. 6, pp. 869-880, 1996.

[18] "Sparkfun," [Online]. Available: https://www.sparkfun.com/products/10736. [Accessed April 2016].

[19] "Sparkfun," [Online]. Available: https://www.sparkfun.com/products/8975. [Accessed April 2016].

[20] R. Byrne, "Global Positioning System Receiver Evaluation Results," 1993.

[21] K.Ohno, T. Tsubouchi, B. Shigematsu, "Differential GPS and Odometry-based Outdoor Navigation of a Mobile Robot," *Advanced Robotics,* vol. 18, no. 6, p. 611–635, 2004.

[22] "Trimble," [Online]. Available: http://www.trimble.com/unmanned/RTK_Positioning.aspx. [Accessed March 2016].

[23] "Hokuyo," [Online]. Available: http://www.hokuyo-aut.jp/. [Accessed April 2016].

[24] "Intro Robotics," [Online]. Available: http://www.intorobotics.com/. [Accessed April 2016].

[25] R. Talluri, J. Aggarwal, "Position Estimation Techniques for an Autonomous Mobile Robot - a Review," *Handbook of Pattern Recognition and Computer Vision*, pp. 769-801, 1993.

[26] E. Galceran, M. Carreras, "A Survey on Coverage Path Planning for Robotics," *Robotics and Autonomous Systems,* no. 61, p. 1258–1276, 2013.

[27] Z. L. Cao, Y. Huang and E. L. Hall, "Region Filling Operations with Random Obstacle Avoidance for Mobile Robots," *Journal of Field Robotics*, vol. 5, no. 2, pp. 87-102, 1988.

[28] T. Balch, "The Case for Randomized Search," in *IEEE International Conference on Robotics and Automation*, San Francisco, 2000.

[29] D. Gage, "Randomized Search Strategies With Imperfect Sensors," in *SPIE Mobile Robots VIII*, Boston, 1993.

[30] H. Choset, P. Pigon, "Coverage Path Planning: The Boustrophedon Cellular Decomposition," in *Proceedings of International Conference on Field and Service Robotics*, 1997.

[31] "Institute of Navigation," [Online]. Available: www.ion.org. [Accessed March 2016].

[32] "The Ninth Annual Robotic Lawnmower Competition. Competition Rulebook," Institute of Navigation, 2012.

[33] "The Quarterly Newsletter," Institute of Navigation, 2012.

[34] "Lawn Rider Competiton," Wright State University, 2012.

[35] R. Arkin, Behaviour Based Robotics, MIT Press, 1998.

[36] T. D. Larsen, K. L. Hansen, "Design of Kalman Filters for Mobile Robots; Evaluation of the Kinematic and Odometric Approach," in *International Conference on Control Applications*, 1999.