TOWARDS DEVELOPING EFFECIENT AREA COVERAGE APPROACH IN DOMESTIC ROBOTS

DISSERTATION

Prabakaran Veerajagadheswar

Department of Advanced Multidisciplinary Engineering

Graduate School of Advanced Science and Technology

Tokyo Denki University

Supervisors:

Professor, Masami Iwase, Tokyo Denki University

Professor, Mohan Rajesh Elara, Singapore University of Technology and Design

March 8, 2019

Acknowledgements

I would like to take this opportunity to extend my sincere thanks and appreciation to my supervisor, Professor Masami Iwase, Tokyo Denki University, for his constant guidance and encouragement.

I want to take this opportunity to extend my sincere thanks and appreciation to my supervisor, Professor Mohan Rajesh Elara, Singapore University of Technology and Design, for his unwavering support, collegiality, and mentorship throughout this project.

I am indebted to Assistant Professor Shunsuke Nansai, Department of Advanced Machinery Engineering, Tokyo Denki University, for Helping and coauthoring my first journal paper.

I would like to show my appreciation to Thejus, Takuma, Vinu, Vengadesh, Ping Cheng, Ang Vu le, and the Researchers at ROAR Lab, Singapore University of Technology and Design for their support and assistance in this project.

Finally, I am most grateful to my wife, Suganya Sampath Kumar, my Daughter, Senthamizhini, my parents, and my in-laws for their continuous motivation, strong moral support, understanding and love.

Table of Contents **Page No**

List of Tables

List of Figures

1. Introduction:

1.1 Motivation:

Domestic robots are becoming an integral part of every household that could improvise the productivity and quality of life through performing repetitive, and time-consuming task. It is estimated that Domestic robots could reach a market value of USD 4.4 Billion in near 2025. Wherein, the robots such as, floor cleaners, lawn moving robot, and pool cleaning robots will hold a larger share. Even though, the dominant market player iRobot, Neato, Samsung, Aquabot, and Husqvarna Automower claims that they have sold millions of units, there exists numerous research challenges and open questions to be addressed. Such research initiatives have constantly pushed the limits of performance in floor cleaning robots in the areas of mechanical design, autonomy, human-robot interaction, and performance benchmarking. Although, the diverse literature on Area coverage robot demonstrates its benefits, and the translation from theoretical design to commercial technology, the conventional robots endure from a series of performance degradation. One major attributes that curtail the efficacy of such robots is their fixed morphology design, which restrains their accessibility in constrained spaces, and weakening its area covering performance. Due to its degraded area coverage performances, the purpose of those root's deployment was not realized essentially. These are some of the facts that motivates to study and contribute a thesis on alternative designs and approaches in order to solve these difficulties in commercial area coverage platforms.

One viable approach to overcome this bottlenecks in area coverage robot is to develop a next-generation designs where a system can adopts its morphology with respect to perceived environment. This thesis introduces a reconfigurable design, where the developed platform adopts or reconfigure its morphology proportionate to its traversed surroundings. In spite of the fact that numerous studies including our prior research efforts in the literature address reconfigurable robotics, they are largely limited to mechanism design, often with a loose connection to an end application. In addition, none of the previous work in reconfigurable robotics targets the area coverage task, which presents significant opportunities for research and development. In the process of developing an area coverage robot, it is critical to address its capacity to demonstrate autonomous and efficient coverage path planning. In most cases with Coverage Path Planning (CPP) strategies, the commonly used motion planning algorithms are spiral motion, backtracking spiral motion, and boustrophedon (back-and-forth) motion. Even though, there are numerous literature demonstrating the advantages of existing motion techniques in the context of CPP, there is a huge gap in implementing those techniques on robots with the reconfigurable base. In particular, the considered reconfigurable design techniques requires a unique motion planning algorithm in order to achieve a better performance. In this thesis, I have proposed a novel coverage motion planning techniques that can be implemented for the proposed reconfigurable class of area coverage platforms.

1.2 Objectives:

- The first main objective of this thesis is to develop a reconfigurable floor cleaning robotic platform in order to maximize the area coverage. The robot will be developed under the inspiration of Tetris game. Tetris is a type of polyominoes which consist of four equivalent square blocks connected at edges. There are seven different shapes defines the Tetris characteristics. In order to achieve each form of Tetris the platform will be developed with hinged joints. Each hinged joints will be connected with a servo motor which aids the robot to reconfigure its morphology with an objective of maximizing the area coverage.
- The second objective is to put forward a novel motion planning technique for a Tetris inspired reconfigurable floor cleaning robot named "hTetro" that can reconfigure its morphology to any of the seven one-sided Tetris pieces. The proposed motion planning technique adapts polyomino tiling theory to tile a defined space, generate reference coordinates, and produces a navigation path to traverse on the generated tileset with an objective of maximizing the area coverage.
- The final objective is to implement the proposed motion planning technique on the developed hTetro robot and validate the coverage performances. The robotic floor cleaner (hTetro) that was developed as part of this research leverages the polyomino tiling theory to automatically generate the global tiling set required to ensure the full area of a given space is covered. In this study, three Tetris tiling theorems were validated using our developed hTetro robot. Also, another objective of this study is to validate the modularity concept on

developed hTetro robot. Wherein we implement Tiling theory, through the medium of a reconfigurable robot cleaner, inspired by Trominoes. The purpose of this theory application is to tackle the issue of cleaning area coverage. For this work, a robot floor cleaner (hTromo) was created, employing polyomino tiling theory to auto-generate a universal tiling set for full coverage of a specified area.

1.3 Organization of the thesis:

CHAPTER 1: Introduction

This Chapter introduce the different types of area coverage platforms and existing literatures on those platforms. Also, this chapter discuss briefly about the existing articles on coverage path planning techniques. At the end, the chapter proposes the novel techniques for better area coverage performance in domestic robots.

CHAPTER 2: hTetro- A Tetris Inspired Shape shifting floor cleaning robot

Although several precedents have demonstrated the benefits of using floor cleaning robots for the maintenance of constructed buildings, traditional platforms suffer from a series of performance issues. One major factor that contributes to their performance deficit is their fixed morphology design, which highly constrains their navigation and access. To this end, a novel Tetris-inspired reconfigurable floor cleaning robot called hTetro is presented in this chapter. The developed robot is capable of reconfiguring its morphology to any of seven one-sided tetrominoes in response to its perceived environment to maximize its coverage area. Also, this chapter presents the coverage area performance of the h-Tetro robot and systematically benchmarks its performance with two commercially available fixed morphology robot platforms. Our experiments indicate that h-Tetro achieves significantly higher coverage performance than the other platforms due to its shape-shifting ability in response to navigating its environment

CHAPTER 3: hTetro- Novel Motion planning techniques for a Tetris inspired reconfigurable robot

Coverage path planning technique is an essential ingredient in every floor cleaning robotic systems. Even though there are numerous approaches that demonstrate the benefits of conventional coverage motion planning techniques, they are mostly limited to fixed morphological platforms. In this chapter, I put forward a novel motion planning technique for a Tetris inspired reconfigurable floor cleaning robot named “hTetro” that can reconfigure its morphology to any of the seven one-sided Tetris pieces. The proposed motion planning technique adapts polyomino tiling theory to tile a defined space, generate reference coordinates, and produces a navigation path to traverse on the generated tileset with an objective of maximizing the area coverage. I have summarized all these aspects and concluded with experiments in a simulated environment that benchmarks the proposed technique with conventional approaches. The results show that the proposed motion planning technique achieves significantly higher performance in all considered parameters (total distance travelled, area re-covered) than the traditional methods.

CHAPTER 4: the implementation of proposed motion planning technique on Tetris inspired reconfigurable robot

Although numerous studies have focused on the development and application of polyomino tiling theories, research of this nature is typically limited to the graphics and gaming fields. In this chapter, I am presenting an innovative application of the polyomino tiling theory which is applied to Tetris-inspired reconfigurable robotic cleaning device as a means of solving the area coverage problem. The robotic floor cleaner (hTetro) that was developed as part of this research leverages the polyomino tiling theory to automatically generate the global tiling set required to ensure the full area of a given space is covered. In this study, three Tetris tiling theorems were validated using our developed hTetro robot. The results of the research clearly indicated that the proposed approach offers a strong area coverage performance across all experimental cases. This chapter includes an outline of the system architecture that underpins the hTetro robot and a comprehensive overview of the three tiling theorems that were applied in this research.

CHAPTER 5: The modularity concept and implementing motion planning technique on other class of polyomino robots

Where the proposed coverage motion planning technique was applied to a Tromino inspired robot called hTromo which is a class of Polyomino robot with three hinged blocks. This chapter will offer a creative application of Tiling theory, through the medium of a reconfigurable robot cleaner, inspired by Trominoes. The purpose of this theory application is to tackle the issue of cleaning area coverage. For this work, a robot floor cleaner (hTromo) was created, employing polyomino tiling theory to auto-generate a universal tiling set for full coverage of a specified area. Three separate Trominoes tiling theories were applied to the hTromo robot, with the corresponding results demonstrating excellent area coverage across all tests, when using the proposed approach. This research also provides an overview of the system framework supporting hTromo, and a summary of the three theories applied to this study.

CHAPTER 6: Conclusion and Future Work
At the end of each chapter are drawn a series of conclusions, concerning the investigations performed and the results obtained. The conclusions are summarized in this capter, where the possible future investigations are also indicated. The PhD thesis ends with a set of references that aim precisely on literatures of area covering domestic robots, reconfigurable robots and its mechanism, Tetris tiling theorems, and coverage path planning techniques.

1.4 Literature Review:

Every household has at least a vacuum cleaner around the house. However one might argue that vacuum cleaners are always underutilized as the users are too busy. Cleaning is an unwelcoming chore that was not enjoyed by many in decades. Vacuuming, a form of cleaning has always been “considered as an “annoying”, “boring”, “time consuming” repetitive task that needs to be done over and over again.” [1]. Inventors hopes to solve the issue of making the vacuum cleaners function independently as no one likes to spend time on it. “Market for household robots are expected to be worth $100B per year by 2020” [2]. To this date, many different types of cleaning robots were invented and “several million units of domestic service robots have already been sold” [3].

1.4.1 Examples of cleaning robots (General overview)

General overviews of the different types of cleaning robots are shown in Table.1.1. Since there are thousands of different cleaning robots in the market currently, the products are chosen based on the area of application in which they specialized in. It is interesting to note that almost all chores can be done by robots in the future. Currently, there are robots that are able to do the dirty work for humans in areas such as floor cleaning (indoor), automatic lawn mowers, swimming pool cleaners, window cleaners and even deep sea cleaners. These intelligent robots may be available for commercial usage, but not everyone can afford them currently, due to high cost in production.

Company	Types of robotic system	Area of application	Basic functions
iRobot Corporation	Roomba	Floor	Independent cleaning robot that automatically recharge and resume scheduled program until it finishes the job.
Robomow	RS612	Lawn mower	Automatic lawn mower that cuts grass in a uniform length within an area surrounded by perimeter wires.
Aquabot	Rapids XLS	Swimming pool	Mobile robot that cleans pool floor and walls with water jets and scrub brushes. The complete cycle includes filtering of dirt and mixing new chemicals for the swimming pool.
Viu Global	Sky Pro Brush standard	High rise building windows	Self-climbing window cleaning robot specially designed for tall and flat buildings
Senseable City Lab	Seaswarm	Oilspill	Autonomously navigate water surface for the purpose of ocean-skimming and oil removal.

Table. 1.1 Examples of different types of cleaning robots for different purposes.

1.4.2 Bibliometric Analysis

A. *Study Methodology*

In this study, cleaning robots are analyzed by having the breakdown of the entire industry and how one constant might affect another. It is well expected in every consumer products that companies are looking into ways to make their products as cheap as possible without compromising its performance [4]. It is always good to keep things simple and maintain the consistency of robotic performance, to avoid it becoming unreliable [5]. The most important subjects when introducing a new development can be defined by the following notes:

- Application
- Cost
- Burnishing
- Keep it simple

B. *Technologies*

The different types of technologies adopted by a typical cleaning robot will be discussed in this chapter. "Indoor cleaning robots have been advanced to a stage of automatic cleaning and battery recharging, and are commercially available" [6]. Early cleaning robots are found to be equipped mainly with *Floor sensors, Collision Sensors, People sensors and Distance sensors.* The sensors were mainly different type of ultrasonic sensors which allows object detection, wall alignment, map building and path finding. The mechanical design on the other hand was kept flat and small so as to allow easy manoeuvrability around obstacles [7]. In the past, researches on mobile robot were focused on various kinds of point-to-point specific tasks, and were unable to make decision when unexpected obstacles appeared [8]. Even the original Roombas cleaned almost entirely randomly [9]. At present, researches on autonomous cleaning robots focused on navigation and map building. Cleaning robots that has a system which creates a digital map offers a larger potential for the future. With such features, a robot would be able to do the following:

- Selection of a start location
- Self-localization (in case it gets lost)
- Automatic planning of cleaning path
- Representation of cleaning area
- Locate different rooms
- Locate charging points or bin disposals

"Sensors are technology that links the real world with digital world by sensing various environmental conditions" [10]. However developments are still needed to perfect the real-time system which will offer the robots more choices when it comes to overcoming an unexpected obstacle. Therefore the most important part of this process will be the development of an environmental map [11].

C. *Advantages*

Robots exist to benefit humans beyond labor substitution. Current technologies allowed the robots to perform difficult tasks with better performance and lesser errors [12]. The advantages of having robots assist us in our routine cleaning will be discussed in this study.

- Cost Vs time efficiency
- Simple to use

- Independent
- Less noise
- Compact and portable

The cost price of any one of the cleaning robots mentioned in Table 1.1.1 would make every family think twice about purchasing it. Although these robots do not come cheap, its lifespan, consistency and amount time users will save can be justified. Users commented that"the whole idea is that you can be doing other things while the robot is cleaning." [13]. Modern cleaning robots are able to identify dangers and thus reducing human errors when it comes to "near misses or accidents that humans tend to make while manually operating floor machines" [14]. Studies have also shown that at least 85 percent of water and chemical were reused by cleaning robots and thus benefiting both the user and environment [15]. Cleaning robots are relatively simple to use and straightforward as it has only one purpose, which is to clean. Most are pre-programmed with schedules so that human interaction is kept at a minimum. Its hardware were also designed to allow free movements across the homes without any help going over door sills or under the couch [16].

D. Disadvantages

By studying the possible limitations a cleaning robot might have, users and researchers alike can decide if a cleaning robot is practical assistive robot to have around at home.

- Small dirt bin
- Takes a long time to complete tasks
- Battery deterioration
- Noisy
- Unable to reach in difficult to access areas
- Preparation of house is needed

Although cleaning robots is much quieter than a standard vacuum cleaner, they still produce unwanted noise [17].One of the biggest concern users have is the constant emptying of bins and recharging of batteries by looking for sockets if they do not have automatic functions [18]. Rechargeable batteries have a lifespan of about 3 years but after constant usage, the battery power dips down to as little as 30–40 minutes when fully charged [19]. Another limitation that might cause unsatisfactory reaction from people with high standards of cleanliness is the limitations in cleaning area covered. Even though the robot may have cleaned the floor of the entire room, it can still miss some spots. Cleaning robots are also unable to move furniture out so as to clean behind them. If the users has pets in the house, the robot might also spend the entire day chasing around pets which are seen as a source of dirts [20]. Hence preparation of the house will be required for a robot to be able to function in its fullest potential.

E. Conclusio: Bibliometric Analysis

The findings from this study will aid researchers and inventors in finding new ways to improve the existing products. Many has argued that robots will never be better than humans when it comes to cleaning as "the human arm and the human eye are still very competent in doing this work" according to Israel Berger the Chairman of Vidaris, an architectural consulting agency [21]. Cleaning robots are something that can be improved upon as the demand is great while the benefits are already outweighing the disadvantage. Therefore, in conclusion, cleaning robots will be around for as long as new technologies are being developed to improve them.

1.4.3 Patent Analysis

A. Keywords

Automatic Cleaning Robot; Autonomous Cleaning Robot; Intelligent Cleaning Robot; Domestic Cleaning Robot; Indoor Cleaning Robots.

B. Study Methodology

Data used in the study of this thesis were retrieved from the Google Patent and Patent Inspiration (PI) database. Filters were used in the gathering of patent data obtained between the period of 1996 and 2016. Keywords were used along with the "AND", "NOT" and "OR" functions to narrow down the results for cleaning robots to be used only for commercial usage. To ensure the relevance of patent results to the scope of study, manual analysis will be done on each patent. The major International Patent Classifiers (IPC) for cleaning robots is provided in Table 1.2. IPC results are determined by researches made by inventors to further improve their products. From the table, we are able to understand that focus was more on "suction cleaners" and "control of position" for the development of cleaning products.

No.	IPC	Short Definition
1.	A47L11/00	Machines for cleaning floors
2.	A47L9/00	Details or accessories of suction cleaners
3.	G05D1/00	Control of position
4.	A47L5/00	Structural features of suction cleaners
5.	B25J9/00	Programme-controlled manipulators
6.	B25J11/00	Manipulators not otherwise provided for
7.	A47L1/00	Cleaning windows
8.	B08B1/00	Cleaning by methods involving the use of tools
9.	B08B9/00	Cleaning hollow articles by methods or apparatus specially adapted thereto
10.	B25J5/00	Manipulators mounted on wheels or on carriages
11.	B08B3/00	Cleaning by methods involving the use or presence of liquid or steam
12.	B25J13/00	Controls for manipulators
13.	A47L7/00	Suction cleaners adapted for additional purposes
14.	G05B19/00	Programme-control systems
15.	E01H1/00	Removing undesirable matter from roads or like surfaces
16.	A01J5/00	Milking machines or devices
17.	G06F19/00	Digital computing or data processing equipment or methods
18.	B25J19/00	Accessories fitted to manipulators
19.	B08B7/00	Cleaning by methods not provided for in a single other subclass or a single group in this subclass
20.	E04H4/00	Swimming or splash baths or pools

Table. 1.2. Main group of IPC code used for identifying a cleaning robot

C. Annual Patent Activities

The summary data representing annual patenting activities for Cleaning Robots are shown in Table 1. 3. During a period of 20 years from 1996 to 2016, a total of 1077 patents have been filed and published. According to the dataset retrieved from PI, the Peak year for research and development of a Cleaning Robot is in 2016 with a total of 367 patent published. China leads the ranking for the most patents contributed by a country with a total of 297 patents published. American company, iRobot Corporation was identified as the company with the most IPC contribution to keyword “Cleaning Robots” Coming up in second and third place is Samsung Electronics (35 patents) and LG Electronics (23 patents).

Time frame	1996 to 2016
Number of years	20
Number of patents filed and published	1077
Peak year	2016 (367 patents)
Top country	China (297 patents)
Top Company	iRobot Corporation (136 patents)

Table. 1.3. Summary of dataset retrieved on Cleaning Robot

“Patents filed” can be described as the study researchers had done to explore more options to improve an existing product. However not all theoretical theories are feasible in the real world, hence patents filed are a work in progress and may not be published. Since the early stages of its development, cleaning robots has been restricted by limitations. The challenges inventors constantly faced were the “lack of available or appropriate sensors and safety during the interaction with humans” [22]. Technology may have been around and available, but the cost of parts may be too much of a burden for researchers who are usually on a strict budget funded by external organisations. It is a known fact that “expensive cost in production and maintenance will always hinder the development of robots” [23]. From Fig 1.1, we can observe that patents filed from 1996 to 2002 had been stagnant. During this period, it can be argued that parts were not readily available for inventors to use commercially. Starting from the year 2002 onwards, the values were observed to be relatively increasing. Coincidentally, 2002 was also the year Roomba, an autonomous floor vacuuming robot, was released by iRobot [24]. Following the success of Roomba, it is understandable that the trend starts to increase with more patents filed in every year after that. The steep increase may be due to available parts in the market and the demand for more of such products. It is worth noting that the slight dip in numbers from 2007 to 2009 was affected by the global recession in 2008 [25]. The major dip in 2016 on the other hand may be due pending submissions filed only a few months ago at the point of writing, hence it should not be a cause of concern.

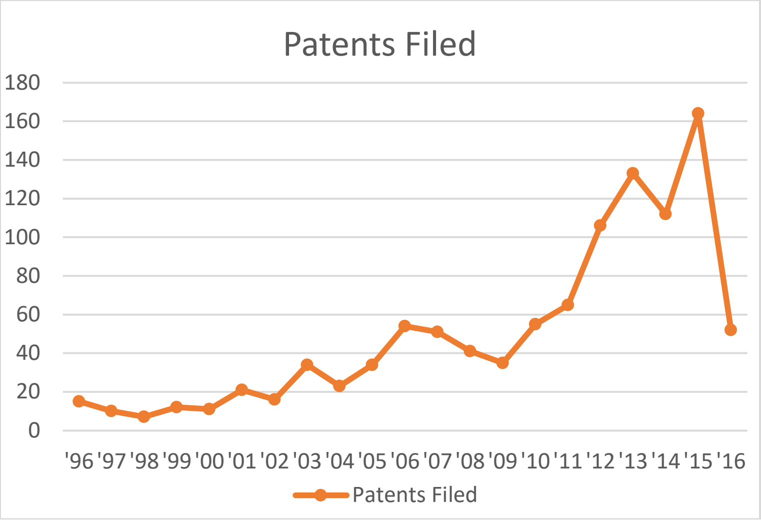


Figure 1.1. Distribution of patents filed over time

Patents published, on the other hand are defined as the successful and accepted applicants. Similarly with patents filed, patents published were also stagnant during the early years building up towards 2002. Records observed from Fig 1..2 were observed to be relatively exponential over time. It is important to understand the relationship between Patents filed and Patents published. A total of 164 patents were filed in 2015 but only 141 patents were published in that same year. By understanding the relationship between the 2 graphs, it can be deduced that not all patents filed in 2015 were published in that same year. This can be proven by observing the year 2016, where only 52 patents were filed in contrast to the 144 published. This meant that most of the patents published in 2016 were filed a year before; with the publishing process taking a long time. The great demand of robotic vacuum cleaners over the years signals a healthy competition among companies to produce new and improved cleaning robots. This encourages new technologies to be developed at a much rapid rate. Intellectual Property Rights (IPR) ensures companies that had filed under them will have their creation and development protected from infringement. To avoid legal issues and complications with external parties, technology companies such as iRobot tend to file and publish every breakthrough technology developed. Companies who are aware of the IPR can also be assumed to be one of the reasons for the recent increase in patents published.

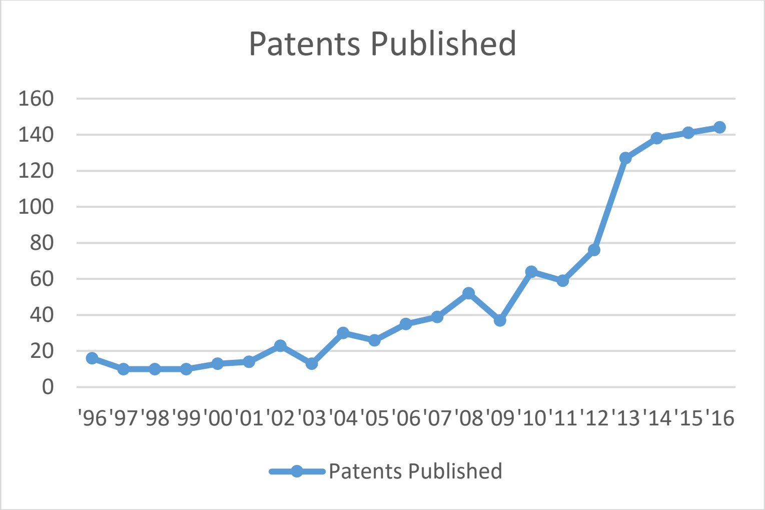


Figure 1.2. Distribution of patents published over time

D. Geographical distribution of patents

The study of geographical distribution of patents will provide a better understanding of the global trends. Technology advancement in different countries are driven by different source of motivations and the level of acceptance by the community. Fig 1..3 illustrates the clear distribution of patents filed and published across different regions. From this data, we can observe that Asian countries are more adventurous in exploring new ideas for cleaning robots. “Asian communities appreciate robots because they are cool; America do not like robots unless they do something useful” (Wilson, 2006).

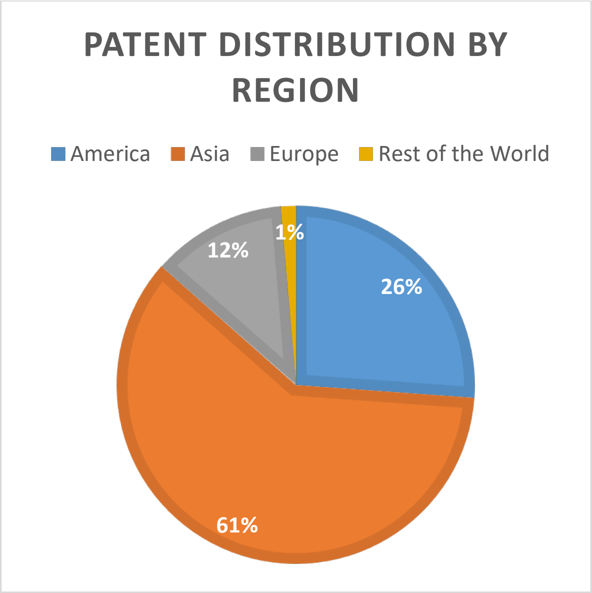


Figure. 1.3. Distribution of patents filed across the regions

Fig 1..4 illustrates a segregation of patents filed and published by individual countries. China is observed to have contributed with the most patents. This is somewhat expected since parts required to develop new inventions can be manufactured locally and at a much cheaper rate compared to the rest of the countries. America has contributed with 26% of the total patent published globally, but only iRobot Corporation can be considered as a major player in the industry locally in terms of amassed contribution. This number can be proven with up to 76% of the patents from USA were contributed by iRobot Corporation alone. China on the other hand has more than 25 companies who are trying to claim a spot in this already proven market. Hence making production from companies in China seem unreliable in terms of quality as they are still new to the industry when compared to iRobot Corporation. Although Republic of Korea only contributed a humbling 18% of the total patents internationally, they are worth looking into due to the impact made in the development of new technologies for cleaning robots. Samsung Electronics and LG electronics are some of the major player in the industry. They were one of the first to implement a self-emptying robot and a low noise robot respectively.

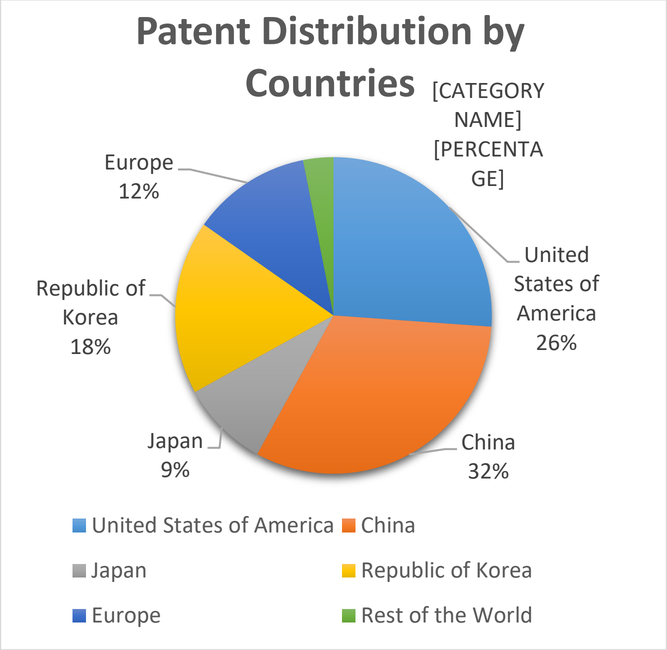


Figure. 1.4. Distribution of patents filed internationally

E. Distribution of top IPC

By referring to major IPC codes used in patents, we can conclude that companies are constantly working on existing issues instead of adding unnecessary add-ons on cleaning machines. Some of the main IPC codes mentioned in Table 1. 2. includes “movement control” and “suction cleaners”. These are some of the basic requirements a cleaning robot should be able to satisfy [26]. Roomba has been rewarded for putting customer’s interests first [27]. By maintaining and improving the basic functionality of the Roomba, iRobot Corporation has been able to keep up with the industry without making many changes to its products. The comparison between different products available in the market will be explored in the later part of the study. However it is worth noting that even though products available in the market were made with the same concept, there will be some distinct difference in terms of additional functions such as air purification system available in some cleaning robots.

F. Conclusion: Patent Analysis

In Conclusion, we can observe that the overall patent activities for cleaning robots have generally been increasing over the period of 20 years. The numbers increases at a much larger rate from 2002 onwards due to the introduction of Roomba from iRobot Corp. One of the reasons for the sudden increase in patents for autonomous cleaning robots are the readily available parts and technology in the modern era. Another possible reason may be due to the increase in demand of cleaning robots to satisfy the fast paced lifestyle of current generations. “The need to be efficient and instant” is what drives millennials into succumbing for help when it comes to the mundane and time consuming chores [28]. Availability of parts was also one of the key factors that allowed China to produce the most patents in the World. However, more patents does not mean better products as proven by USA and South Korea which houses the top 3 Companies publishing patents on domestic cleaning robots. Patent analysis for this study was done using data mining to understand the relationship of patents filed and published over time, countries, and the different IPR for autonomous cleaning robots. The results and information presented may have been restricted by the limitations of the free version of Patent Inspiration. Further study was needed to ensure the numbers tally with social and civil events happening around the world.

1.4.4 Commercial Product Analysis

A. Study methodology

Existing products available in the market can be used as a benchmark or case study for every future development of the automatic cleaning robots. Data obtained in this study were retrieved from Google search with the main purpose of exploring the way every product were perceived and adopted by users. "Cleaning robots was one of the first members of the service robot family to be available" commercially to the public [29]. To ensure products are being judged fairly, we will see based on how each products faired in their respective areas of specialty. Indoor cleaning robots can be classified in many different sub-groups as technology has "advanced to a stage of automatic cleaning and battery recharging" [30]. For this study, focus will be stressed upon a more specialized area of floor cleaning which also includes vacuum cleaning and mopping. Secondly, product features will be discussed in terms of importance in the development of an intelligent cleaning robot. To keep up with the trends and relate to consumers, products are often found to have unnecessary features that do not compliment the main purpose of the machine.

B. Types of Domestic Cleaning robots

Different types of products in the market can be compared by referring to Table 1. 3. The Roomba is one of the "most widely deployed vacuum cleaning robots" globally [31]. The advance model features functions such as locating the charging point (to perform self-recharge) and then resuming where it had stopped cleaning until the job is completed. To maintain relevance in the market, different companies develop additional features to compete with the Roomba. For example, Samsung's SR8900 ROBOT VC has the unique function of self-emptying its dustbin and is able to memorize locations of permanent obstacles such as furniture. LG Electronic managed to develop a low noise cleaning robot that does not compromise its ability to do its job effectively. These are examples of patented technologies that would make the creators of Roomba envious. Other products mentioned in Table 1..3 may not be equipped enough to compete with the big players in the domestic cleaning robots industry. Milagrow on the other hand focused its development on a wet cleaning robot to compete with "Scooba – a mop version of Roomba" by iRobot Corp. Electrolux, "the world's largest household appliance manufacturer" was the first to "unveil the prototype for robotic vacuum cleaner" [32]. It went on to develop the Trilobite, but was unable to compete in the market and has since been discontinued. Trilobite was a randomly navigated cleaning robot that bumps into walls as it goes around the house to clean. It also required the cleaning area to be defined by a magnetic strip so as to act as boundary for the robot. Even the most basic Roomba model had a pre-scheduled cleaning option that operates automatically with minimum adjustment needed around the house. Without an updated navigation system, cleaning robots would not be able to recall where it had last cleaned if its battery died or was picked up by humans. It would be safe to assume that other companies might just follow suit into discontinuing their versions of cleaning robots if they are unable to keep up with the ever changing requirements of an intelligent robot.

Company	Model	Area of Specialty	Features
IRobot Corporation	Roomba	Floor cleaning	Basic models: Pre-scheduled independent robots that clean, brush and vacuum room assisted by dirt sensors and real time path visualization. Advance model: Cleans up to 3 rooms and automatically recharge and resume itself until job is done. Stronger vacuum suctions (captures up to 10 micron particles) and connected to smart phone for easy accessibility.
Neato robotics	Botvac	Floor cleaning	Pre-scheduled independent robot which will automatically recharge and resume until its job is done. Scans and maps the rooms, plans, and methodically clean the room. Its unique shape and larger brush allows easy access to corners.

SAMSUNG	SR8900 ROBOT VC	Floor cleaning	Self-charging and self-emptying cleaning robot that will not stop until pre-scheduled task has been completed. Dust sensors and real time mapping system ensures it overcome obstacles by memorizing the best cleaning path.
LG Electronics	VR65710LVMP	Floor and corner cleaning	Low noise, pre-scheduled cleaning robot that focuses on corners. Scans ceiling with built in camera to identify room/area to be cleaned. Digital bumper avoids obstacles and memorizes location of obstacles in the room.
Milagrow Business & Knowledge Solutions	IMap Water	Wet cleaning	Offers wet cleaning, air purification, real time path planning, and recharges and resume cleaning automatically.
Electrolux	Trilobite*	Vacuum cleaning	Random navigation method with magnetic strip on the floor (requires pre-arrangement around the house) which acts as its boundary.
METAPO Inc.	Cleanmate QQ3 LCD	Vacuum cleaning	Clean and vacuums cleaning area confined by sonic wall. Search and locate charging base when battery is low.
SHARP Corporation	RX-V100-W	Vacuum cleaning	Equipped with ultrasound sensors to avoid obstacles, this automatic vacuum cum air purifier will return to charging dock to recharge when its battery is low.
Moneual	RYDIS	Vacuum cleaning	Semi-automatic vacuum which navigates around the room by bumping into walls. A remote control can also be used for manual manoeuvre. Status indicator lights blinks for full dustbin or low battery status.

Table. 1.4. Different types of domestic cleaning robots available in the market

C. Reliability

After a 6 month study was conducted by a group of researchers in Switzerland on “Living with a Vacuum Cleaning Robot” [33]. The effectiveness of a product will be judged by cross referencing the requirements and adoption methods humans has on the different types of cleaning robots. Perceived effectiveness inventors expected of their robots are usually over estimated. In fact, “a lot of studies of the working robots reported have not achieved them” [34]. However, the product will be deemed successful if it is able to satisfy the basic functions of autonomous movement and cleaning. According to the study, “only three out of nine household adopted the robot” and fully utilising its functions. The usage of cleaning robots was found to decrease over time [35]. This is mainly due to the level of attachment one has on an exciting, new technology and are curious to try them out. Hence the decline in usage does not mean that the machines are not reliable. Although the cleaning robots were able to complete their tasks successfully, many users still prefer having a personal touch while doing chores. Humans are able to adapt to different situations which robotic intelligence still lacks in capacity, hence “Cleaning robot are not a substitute for a cleaning human” [36].The main objective of having cleaning robots around is to allow people to be “doing other things while the robot is cleaning”, thus increasing human efficiency in getting more thing done [37]. While there are mixed reviews in the actual acceptance of the robots by the public, it can be argued that the cleaning robots are actually fulfilling their purpose of automatic pre-scheduled cleaning. To keep it simple, a product is useful as long as it is able to move independently and clean upon commands. However, products with at least one of following features can be classified as a highly reliable product:

- Pre-scheduled cleaning
- Self-recharging of battery
- Self-emptying of dirt bins
- Real-time mapping system

D. Conclusion: Consumer Analysis

We have explored the different types of products available in the market for autonomous cleaning robots in this chapter. By understanding the acceptance and thoughts humans have towards robots, we can look into ways to improve future developments positively. One of the concerns people have when they hear the word ‘service robots’, “people immediately assume worker displacements” [38].Those who make a living by cleaning will generally be expected have poor perceptions on such robots. Cleaning robots usually do take a much longer time to clean an entire room as compared to a human being [39]. Humans are able to visually identify dirt and proceed to execute the removal of unwanted objects by using a broom or a vacuum cleaner. Problem solving skills of a human brain gives us an added advantage over the robots while maneuvering of cleaning equipment into tight spaces [40]. Through the results from this study, we can conclude that the society, especially the older generations, are not ready to accept cleaning robots as part of our household. However the robots do benefitted those who were able to adapt in having a cleaning robot work for them. Even though doing chores with a vacuum cleaner may be more effective, cleaning robots offers users a more frequent cleaning schedule which is a good return for efficiency driven millennials [41].

.

1.4.5 Technological analysis

The various aspects of floor cleaning robots, including mechanism design, human interaction, and autonomy, have been researched in depth over the last two decades [43].

A. Mechanical Design

Gao et al. present a floor cleaning robot with Swedish wheels that attains highly efficient locomotion in crowded places such as railway stations and airports [44]. In another work, Kakudou et al. presents a novel staircase cleaning robot equipped with L-shaped legs on both sides of its body frame for staircase ascent and descent locomotion [45].

B. Autonomy:

An innovative neural network approach towards robot autonomy is conceived and validated by Yang and Luo in [46]. Their method covers path planning and obstacle avoidance for cleaning robots in dynamic environments. In their approach, an evolving action scene is generated by a neural network, and the robot’s last position is used for path planning. Siop et al. [47] discuss a new technique called triangular cell map representation that helps a cleaning robot identify shorter paths, offering increased navigational flexibility compared with rectangular cell maps. They also propose complete coverage navigation and methods for map construction to enable a cleaning robot to perform navigation on an entire workspace given basic information regarding its environment. In [48], the authors experimentally investigate the coverage performance of a chaotic autonomous mobile robot. The proposed approach uses a microcontroller to generate a chaotic bit sequence and suitable robot trajectory. A SLAM-based approach for improving the coverage of floor cleaning robots is presented in [49]. This method uses a fusion of magnetic field and odometry data to synthesize an accurate map for efficient autonomous navigation and improved area coverage.

C. Other Technological Analysis:

Several human-robot interaction studies involving floor cleaning robots have been reported in recent years. Fink et al. [50] report on an ethnographic study of floor cleaning robots in separate households over a period of approximately six months. In their study, user perception, usage analysis, and social activities with respect to cleaning robots are extensively studied. Sakamoto et al. present a stroke gesture-based computer screen interface for interacting with a floor cleaning robot [51]. Their experiments demonstrate the validity of the proposed interface in controlling a cleaning robot and capturing multiple views from ceiling cameras. Luo and Yang propose multi-robot cooperative sweeping based on a bio-inspired neural network approach [52]. The authors use multiple cleaning robots to study the complete coverage and path planning issues associated with unstructured workspaces. Another significant study [53] presents a cellular decomposition method to divide a cleaning area into cells and provide an effective coverage plan to two indoor floor cleaning robots over the cells. With the market becoming increasingly flooded with floor cleaning robotic products, interest in benchmarking various robotic systems has also grown. A set of performance metrics for autonomous cleaning robots is reported in Rhim et al. [54], who define autonomous mobility, dust collection, and operation noise as critical performance indices for the majority of cleaning robots. Wong et al. propose two metrics to capture robotic cleaning performance and coverage efficiency and validate them with real-world trials [55]. This study also presents experimental approaches using computer vision techniques to benchmark the performance of cleaning robots. Zheng et al. in [56] present the evaluation of path planning algorithm for multiple floor cleaning robot platforms using a Mocap system. This work also put forward and validate three simple metrics to benchmark the overall performance of the area coverage algorithm. Prassler et al. in [57] share their experiences in organizing the First International Contest for Cleaning Robots, which took place jointly with IROS 2002 in Lausanne, Switzerland. This work provides extensive details on the efforts made to ensure a well-defined setup, which could be replicated for every run of the competition contest and participating team. Also, this work put forward a set of fair criteria for benchmarking the performance of the participating cleaning robots.

In Conclusion, results from all three major analyses done in this study suggests that cleaning robots are here to stay. Despite requiring humans to adapt to the existing products in the market, the reviews have been positive. “The robots encouraged people to make a variety of changes to their home” [42]. Once a user understands the limitations of the cleaning robots, they were able to use the machines more effectively. The cleaning robots had fulfilled its purpose of independently completing the chore that nobody likes to do. Bibliometric analysis done in the first part has helped us gain the overview knowledge of the cleaning robot industry. Patent analysis was crucial in providing specific insights and trends taken by major corporations to stay relevant in the cleaning robot industry. Finally, Commercial product analysis was done to observe the relationship between the products and users.

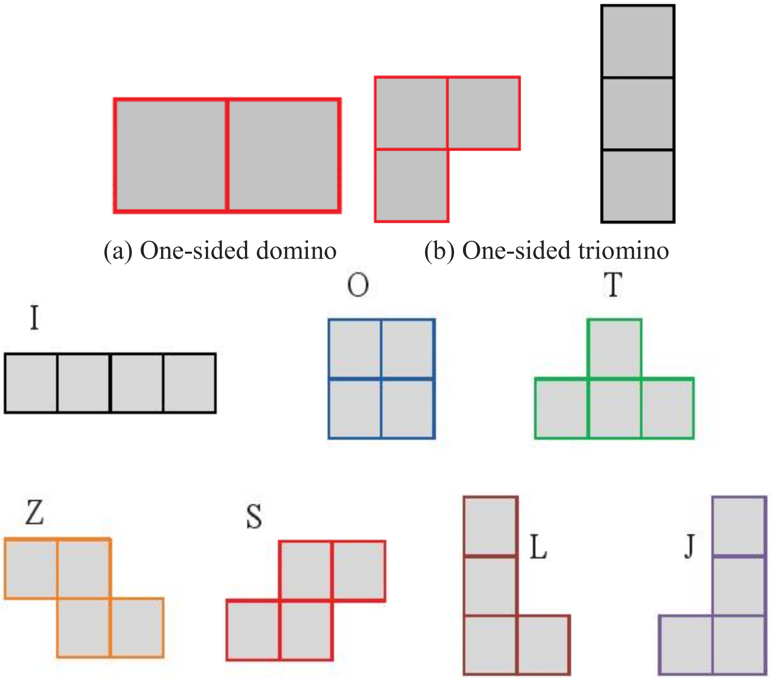
Although numerous studies in the literature demonstrate the use of floor cleaning robots for the maintenance of constructed building spaces, conventional robots suffer from a series of performance issues that curtail their full potential. One major factor that contributes to the performance degradation associated with cleaning robots is their fixed morphology design, which highly limits their navigation and access. One viable approach to overcome this bottleneck is to develop a next-generation cleaning robot that can reconfigure its morphology to maximize cleaning performance.

One viable approach to overcome this bottleneck is to develop a next-generation cleaning robot that can reconfigure its morphology to maximize cleaning performance.
Since the early 1980s, the design philosophy of reconfiguration has been actively considered and applied to robotic systems. As a result, a number of reconfigurable robotic platforms have been proposed. It is a proven fact that reconfigurability has widened the scope and diversity of robot design. In our previous work, we classified reconfigurable robots broadly into three categories, namely, inter-, intra- and nested reconfigurable platforms [58]. An intra-reconfigurable robot comprises a combination of sensors, actuators, mechanical parts, power and a controller and acts as a single entity capable of changing its internal morphology. Precedents to this end include a versatile robot that can switch between amphibious and terrestrial gait mechanisms [59], an anthropomorphic robotic hand capable of reconfiguring its palm to generate changeable topologies [60], a reconfigurable, underactuated, legged robot that generates a variety of locomotion gaits [61] and a reconfigurable bio-inspired robot that can switch between three locomotion modes: rolling, climbing and crawling [62]. Robots in the inter-reconfigurable category consist of modular homogeneous or heterogeneous units that can assume a variety of morphologies via assembly and/or disassembly processes [63]. One example of an inter-reconfigurable robot is Sambot [64]. Multiple Sambots can assemble and disassemble to form new robot morphologies. Another case involves an underwater platform capable of splitting into independent rigid modules and joints to swim like an eel [65]. In [66], a robot is described as being capable of multiple locomotion modes via assembly and disassembly processes. CEBOT, Poly Bot, Crystalline, M-TRAN, ATRON, Molecube and CKBot are other relevant examples of inter-reconfigurable robots. Robots in the third class, nested reconfigurable robots, are capable of performing both inter- and intra-reconfigurability, thereby expanding their reconfiguration options. Hinged-Tetro, presented in [67], is a nested reconfigurable robot capable of reconfiguring its morphology as an individual as well as assembling/disassembling itself to generate global morphologies. In spite of the fact that numerous studies including our prior research efforts in the literature address reconfigurable robotics, they are largely limited to mechanism design, often with a loose connection to an end application. In addition, none of the previous work in reconfigurable robotics targets the floor cleaning task, which presents significant opportunities for research and development.

2. hTetro- A Tetris Inspired Shape shifting floor cleaning robot

2.1 Design Principle of the hTetro Robot

Polyominoes are the inspiration for the design of our h-Tetro robot and its reconfiguration strategies. Polyominoes are two-dimensional geometric structures joining one or more similar squares edge to edge in varying arrangements. A polyomino consisting of n squares is termed as n-omino or n-polyomino [68]. A polyomino with a combination of two congruent squares is called a domino, a combination of three squares forms a triomino, and a combination of four squares forms a tetromino, as shown in Fig 2.1. Polyominoes have been popularly adopted in several entertainment puzzles since the 18th century [69].



(a) One-sided domino (b) One-sided triomino

(c) One-sided tetromino

Figure 2.1. Examples of polyominoes.

Polyominoes can be classified as free polyominoes, one-sided polyominoes, and fixed polyominoes based on their geometry and chirality. The free polyomino can be considered as a subset of the n-polyomino domain such that each element exhibits chirality in terms of its spatial orientation. One-sided polyominoes exhibit similarity in terms of their geometry, irrespective of right-angled rotational transformations. Unlike free polyominoes, one-sided polyominoes do not show chirality. Hence, the mirror image of the structure of each one-sided polyomino can be treated as another distinct element belonging to the same subset. Fixed polyominoes can be treated as another subset of the polyomino domain wherein each element (lattice animal) is distinct in terms of its rotational and flip transformations. Based on [70], single one-sided, single free and two fixed dominoes form the domino group of polyominoes. Similarly, two free, two one-sided, and six fixed triominoes (3-ominoes) form the triomino group. In [70] it is documented that, for a group of tetrominoes, there exists five free, seven one-sided, and nineteen fixed tetrominoes. Because the tetromino accommodates a large number of polymorphs, we adopted the tetromino geometrical structure as the morphology of our self-reconfigurable cleaning robot. Our hTetro platform is capable of reconfiguring its morphology into any of seven one-sided tetrominoes in response to its perceived environment with the objective of achieving optimal coverage area and, thereby, cleaning performance. The strategy adopted in our research is analogous to the gap filling approach in conventional Tetris games. Fig 2.1(c) illustrates the seven free tetromino structures adopted for dynamic morphology generation in the hTetro robot.

The main challenge in designing the hTetro robot was the placement of hinged dissection points to enable shape-shifting from one configuration to another. A hinged dissection is a geometric methodology that divides a planar structure into a finite number of pieces connected by “hinged” points in such a way that a new shape can be created by rearranging the dissected pieces without breaking the chain structure formed by the hinged points [71]. Fig 2.2 illustrates the hinged dissection approach. Several studies have reported on the concept of hinged dissection since 1984. For instance, hinged dissection can be used to transform an equilateral triangle into a polygon [72]. In addition, [73] demonstrates the usage of 3D hinged dissection for polyhedra involving connected 3D solids formed by merging several rigid copies of the same polyhedron along identical faces. In [74], a hinged dissection based approach to pattern generation is presented. The patterns are created so that, when they are rotated from one shape of the dissection to another, the patterns also change along with the shapes.

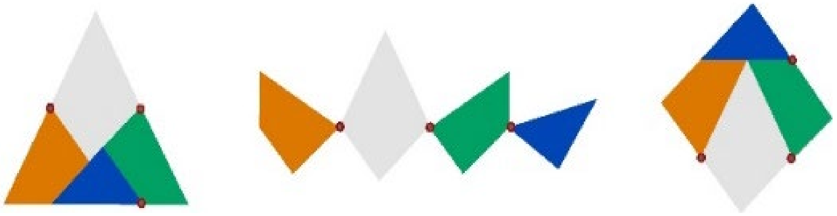


Figure 2.2. Hinged dissection.

The current literature on hinged dissection is generally confined to solving a typical dissection problem in planar and spatial geometry. We presented an innovative application of hinged dissection in a previous study wherein we developed a nested reconfigurable robot module called hinged-Tetro to prove that LLR or LLL hinged dissection can be used to achieve all seven one-sided tetrominoes [58]. In this Chapter, we build on our past work [75] to quantifying the performance superiority of our shape-shifting approach to floor cleaning robots. For the work presented in this Chapter, we adopted the LLR hinged dissection shown in Table 2. I to enable its shape-shifting capabilities.

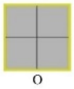
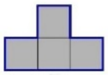
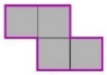
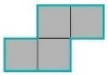
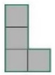
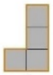
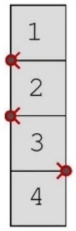
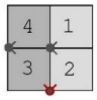
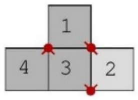
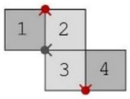
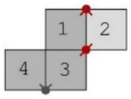
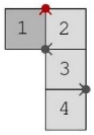
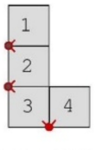
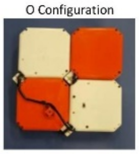
Hinged Dissection	 O	 T	 Z	 S	 L	 J
 (LLR)	 1) $^{2}R_{3} = -180^{0}$	 1) $^{2}R_{3} = -180^{0}$ 2) $^{3}R_{4} = +180^{0}$ 3) $^{2}R_{1} = +90^{0}$	 1) $^{2}R_{1} = +180^{0}$ 2) $^{3}R_{4} = +180^{0}$	 1) $^{2}R_{1} = +180^{0}$ 2) $^{2}R_{3} = -90^{0}$	 1) $^{2}R_{1} = +180^{0}$	 1) $^{3}R_{4} = +180^{0}$
Robot Prototype (LLR) I Configuration 	O Configuration 	T Configuration 	Z Configuration 	S Configuration 	L Configuration 	J Configuration

Table 2.1. Hinged dissection of tetrominoes in LLR configuration.

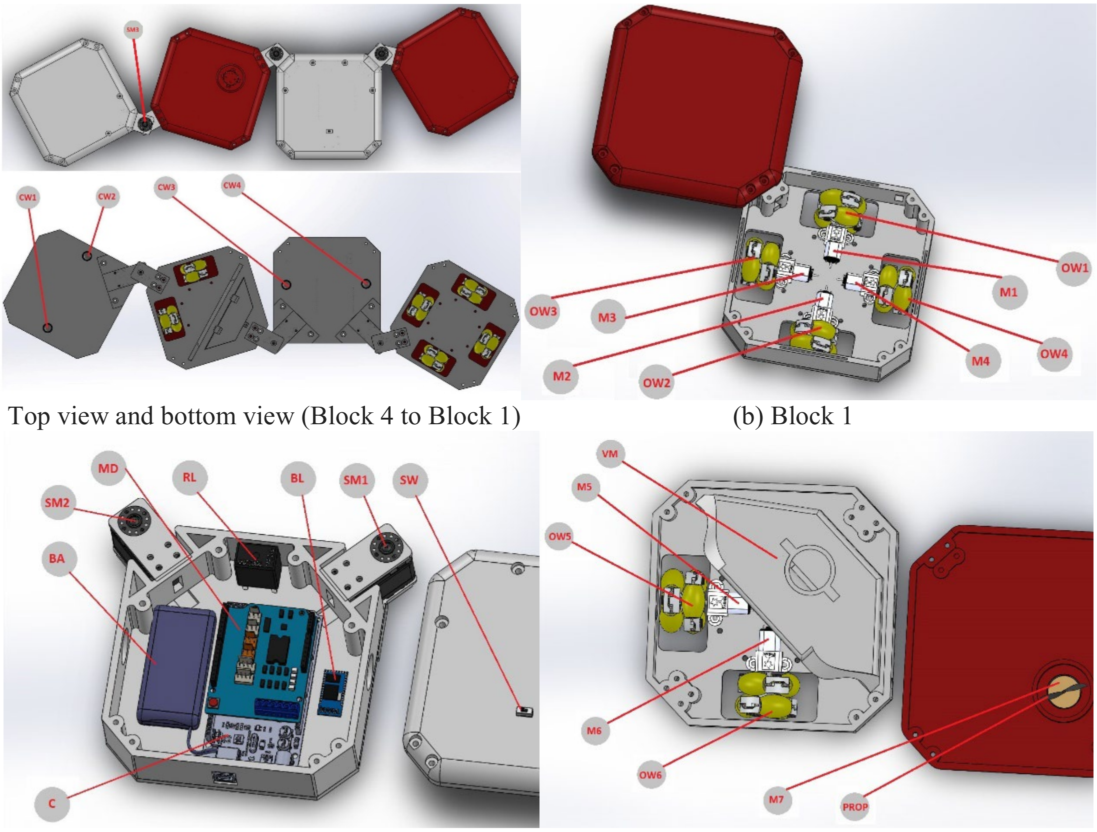
2.2 hTtetro System Architecture

As shown in Fig 2.3(a), our hTetro robot consists of four square blocks with three (LLR) hinged actuation points. Because our hTetro robot is compact, each block is assigned a particular function. For instance, Block 1 is responsible for mobility functions, Block 2 houses the control and power modules and Blocks 3 and 4 hold the cleaning functions.

2.2.1 Mechanical Design

The enclosure of the robot has four square blocks connected by three active hinges. Each block of the hTetro robot has the following dimensions: 140 mm length, 140 mm width and 55 mm height. The blocks are designed with no sharp edges to avoid collisions between the blocks during transformation. The walls of the robot enclosure are fabricated in honeycomb patterns with a thickness of 4 mm to minimize weight and maximize tensile strength. The total weight of the hTetro robot, including all its peripheral devices, is approximately 3 kg. The hTetro robot is equipped with a set of six motors for mobility and three motors for transformation. For the robot's locomotion, six Pololu DC geared motors with a working voltage of 7.4 V are used. As it is responsible for mobility, four of the six DC motors (M1 to M4) are mounted in Block 1. The remaining two motors (M5 and M6) are placed in Block 3, which ensures smooth navigation for the hTetro robot. Each motor is mounted on a spring suspension to overcome bumps and traverse uneven terrain. Because we use multiple DC motors for locomotion under seven different robot morphologies, each motor is programmed to operate differently under specific configurations. For instance, motors M1, M2 and M5 are used for forward and reverse motions in the 'I', 'S', and 'J' configurations and perform a lateral motion in the 'O', 'T', 'Z', and 'L' configurations. Motors M3, M4 and M6 act in the opposite manner. We use omnidirectional wheels (OW1-OW6) for our hTetro robot to realize multi-directional locomotion. In addition, four caster wheels (CW1-4) are mounted in Blocks 2 and 4 to improve robot's mobility. Seven different configurations of hTetro can be realized by controlling three actuated hinged points. Three Herkulex DRS-0101-7 V smart servos activate the hinges and control reconfigurability. The smart servo motors can rotate to a maximum of 270 degrees and have an inbuilt encoder feedback system for precise control. The servos SM1 and SM2 attached to Block 2 (Fig 2. 3(c)) act as an anchor and drive both Block 1 and Block 3. SM3 is placed in Block 4 (see Fig 2. 3(a)) and drives Block 4. Each servo has a stall torque of approximately 12 kg, which allows the motor to lock the positions of the blocks at the end of every transformation. When the robot starts moving, the blocks are automatically locked to maintain the morphology of the robot throughout its locomotion. Additionally, A separate vacuum module VM is integrated into Block 3 (Fig 2. 3(d)), wherein a high RPM motor M7 is used for realizing the vacuuming function shown in Fig 2. 3(d), which performs vacuuming functions and collects dirt particles. The path of the suction module is designed to avoid vacuum or dust leakage during cleaning.

The locomotion and transformation of the robot are achieved by coordinating its actuators and microcontroller. An Arduino Atmega2560 16-Bit microcontroller is used to control the robot's locomotion, shape transformation and human-robot interaction. It has 54 digital input/output pins, 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. The microcontroller is located in the second block of our robot, as seen in Fig 2. 3(b). The microcontroller is programmed to carry out 3 major functions: (1) generate control signals to the motor driver unit MD, which controls the speed of the M1- M6 motors, (2) control signal generation for the servo motors (SM1-SM3) for reconfiguration and receive feedback from them and (3) receive user commands from a remote smartphone device. The motor driver used has four 4 H-Bridges TB6612 chipset provides 1.2A per bridge (3A for brief 20ms peaks) with thermal shutdown protection, internal kickback protection diodes, and can run motors on 4.5 VDC to 13.5 VDC with individual 8-bit speed selection. For each of the six motors, the speed and time period for the motor's rotations are defined by pulse-width modulation (PWM) signals from the microcontroller. Because the robot is deployed with six motors, a 5v DPDT (double pole double through) relay switch, RL, with 1A of switching load current is used to switch between the motors for different configurations. In every transformation, the microcontroller gives a serial command (trough RX1-TX1 pin) to SM1-3, and at the end of the transformation, the controller receives feedback on the motor's position from SM1-3 to cross-check its position. The controller receives wireless commands from a remote user (through RX0-TX0 pins) to perform a task, for which an HC-06 (5 V) Bluetooth device BL with 2.4GHz ISM band frequency is used. The entire circuit is powered using a LIPO battery BA with 7.4 V and 900 mAh capacity. A toggle switch (SW) is connected to the main power supply branch, which enables convenient circuit assembly and breakage. The detailed system architecture of electrical connections is shown in Fig 2. 4. Even though our hTetro robot would require increased number of actuators and computational needs to realize the shape shifting features in comparison to traditional floor cleaning robotic platforms. The performance superiority in area coverage that comes with our hTetro platform largely outweighs the additional needs for actuators and computational needs. In addition, the advancement of actuation and computing technology over the last two decades have resulted in improved sophistication, small footprint, high precision and lower cost of such systems.



(a) Top view and bottom view (Block 4 to Block 1) (b) Block 1

(c) Block 2 (d) Block 3

Figure 2.3. hTetro's CAD and the component used in each block.

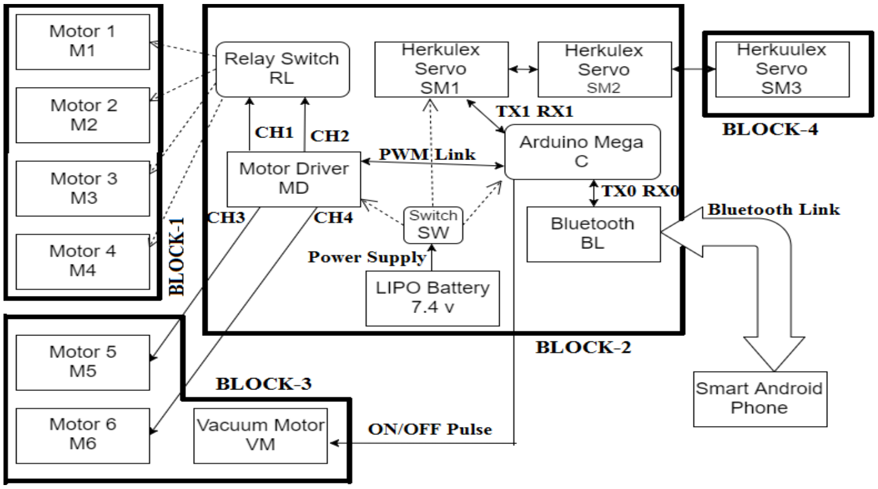


Figure 2.4. System architecture of the hTetro robot.

2.2.2 Software Design

hTetro receives manual commands to achieve transformations between seven tetromino morphologies. The user can choose the most appropriate morphology for the perceived environment with the objective of achieving maximum coverage. The current hTetro prototype is an initial design in the direction of an autonomous shape-shifting robot requiring no human intervention. A mobile application was developed and installed on an Android/IOS smartphone device. The developed application contains navigation and seven transformation buttons on the control page as shown in Fig 2. 5 (right). Before sending commands from the developed app, a Bluetooth connection must be established between the application and the Bluetooth device BL mounted on the hTtero. The user can select the required device from the "Available Device" box as shown in Fig 2. 5 (center); this box appears after selecting the "Plus" symbol in Fig 2. 5 (left). Once a connection has been established, the robot control page appears onscreen, and the hTetro robot can react to the commands received from the application. In the application, the arrow keys are used for navigational commands, and the play icon is used to pause or start the current operation. The ON and OFF buttons are used to perform the vacuuming function.

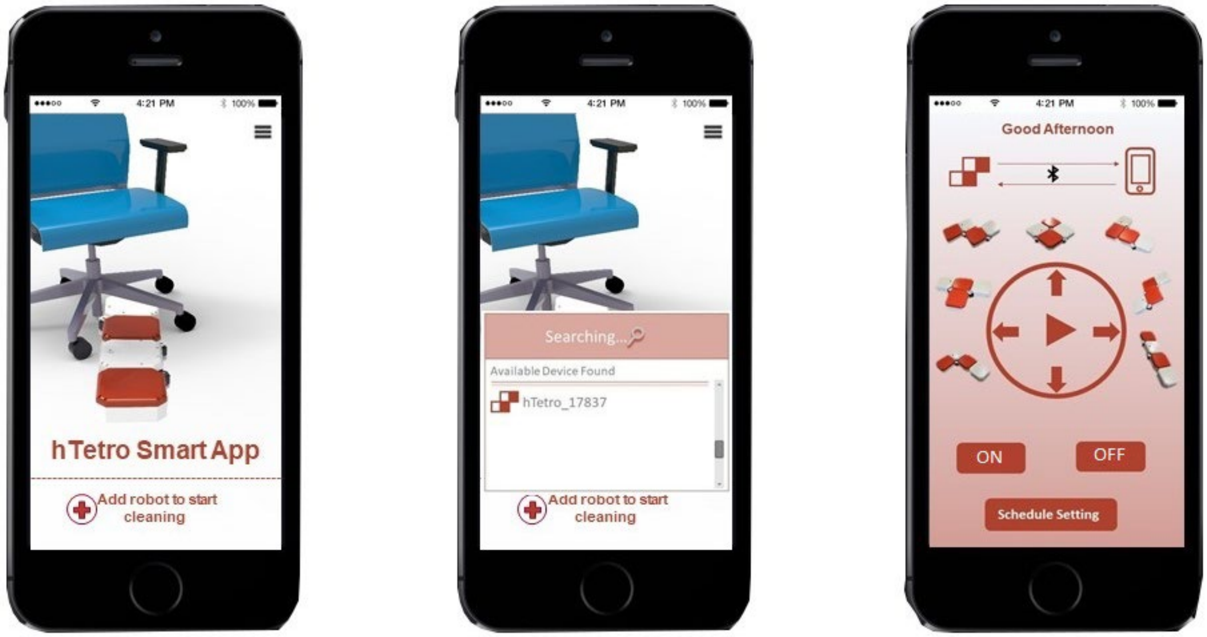


Figure 2.5. hTetro graphical user interface.

2.3 Experiments

We evaluated the performance of our hTetro robot against two commercially available fixed morphology robots under two experimental conditions. In the first set of experiments, the robots were deployed to clean a fixed area with a single chair as an obstacle. The experiment was performed using ten different types of chairs with five trial runs of each chair. The coverage areas of hTetro and the other two commercially available robots were computed using all ten chairs, and we compared their coverage performance. The chair types used in the first experiment are shown in Fig 2. 6. Each chair was assigned a complexity rank based on its ease of accessibility to the cleaning robots. Due to the fact that a chair with more legs occupies greater space, thereby decreasing its accessibility to cleaning robots, a chair's complexity rank was defined by a score of 1-10 based on its number of legs and the distance between them. During the second set of experiments, hTetro's performance was compared with two fixed morphology robots with distinct morphologies in an office-like environment. One of the commercial robot platforms was circular and the other was 'D'-shaped. These two fixed morphologies are the most commonly encountered morphologies in existing commercial floor cleaning robots. Within the same experimental setting, the coverage area performance of the considered robots was computed across varying degrees of obstacle density. To standardize the comparison, the commercial robots were reverse engineered to be manually controlled using an Android smartphone application. The two fixed morphology experimental platforms used in our experiments are shown in Fig 2.7. The experimental testbed consisted of a predefined floor area to be cleaned, support frames, an image acquisition device and a vision based benchmarking scheme that computed the coverage area automatically by tracking the robot within the operating space.

2.3.1 Predefined Floor Area

The predefined floor area considered in this Chapter is a rectangular, polished laboratory floor with boundaries defined by extendable metal frames. The surface area of the cleaning space can be varied by reconfiguring the arrangement of the extendable metal frames. For the first set of experiments, we used an area of 93 cm x 168 cm, and, for the second set of experiments, we used an area of 200 cm x 168 cm. A camera was placed perpendicular to the cleaning surface with a fixed focal length. The altitude for fixing the camera was selected so that the entire cleaning area was covered in the camera's field of view. We also ensured that the movement of the robot would not be eclipsed by any of the obstacles. For instance, the seats of the chairs were removed so that the benchmarking scheme could track the robot's movement under chairs (see Fig 2.8).

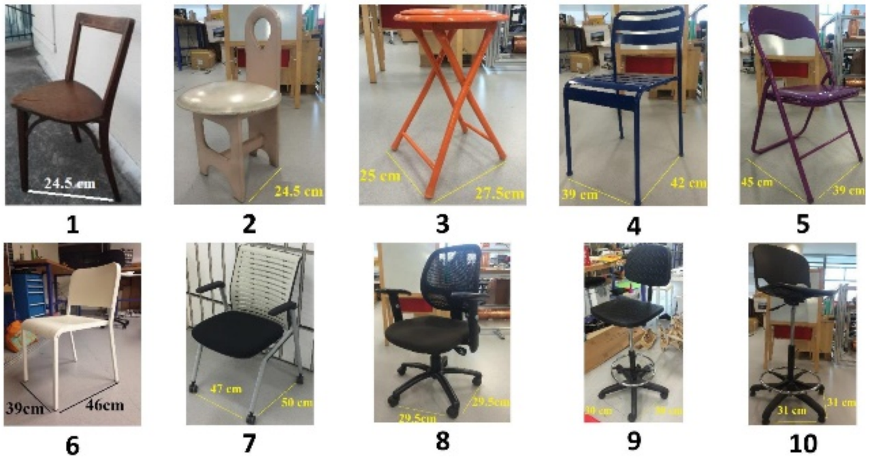
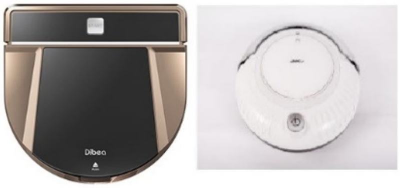


Figure 2.6. Chair types used in the experiments.



Specifications of Fixed morphology robots

Size of the Circular Robot	280 (Diameter) X 75 (height) mm
Size of the 'D' shaped Robot	290 (Diameter) X 76 (height) mm

Figure 2.7. Experimental fixed morphology platforms.

2.3.2 Support Frame

Using a T-slot aluminum extrusion profile, a parallelepiped structure was built to provide vibration-immune support for the camera mount over the cleaning space (see Fig 2. 9). The support frame has the following dimensions: 2000 mm length, 2000 mm breadth, and 2500 mm height. The camera was fixed exactly at the midpoint of the upper face of the supporting frame.

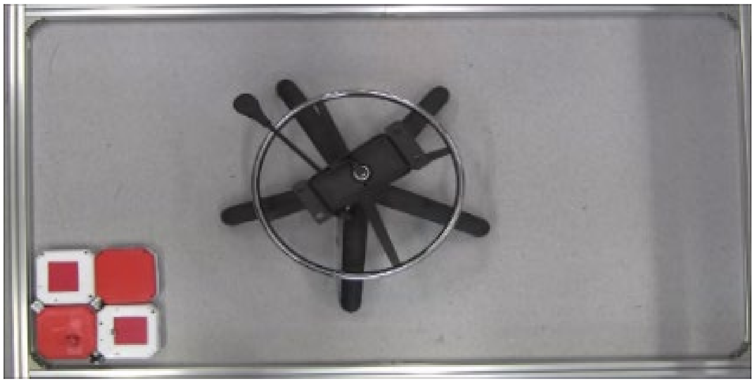


Figure 2.8. Experimental setup after removing the seat of the chair.

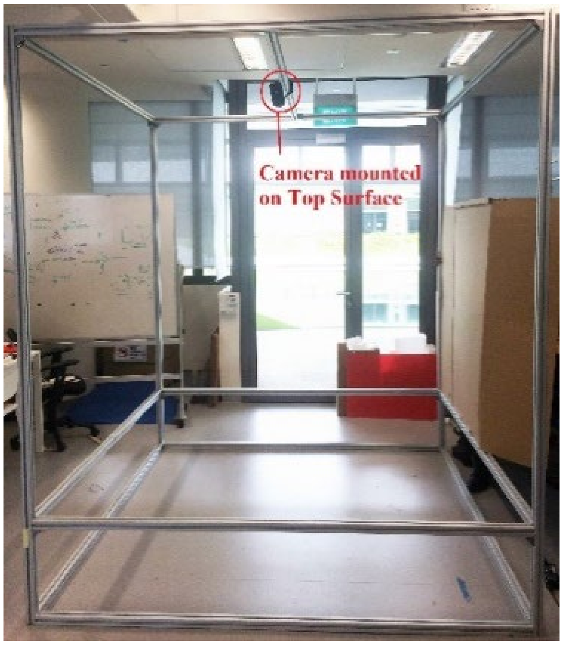


Figure 2.9. Complete experimental setup.

2.3.3 Image Acquisition Device

We used a digital HD camcorder with a resolution of 2.07 megapixels as an image acquisition device. The employed camcorder has a built-in storage memory for storing all the recorded raw video footage. The camera used has a frame rate of 24 fps with a fixed focal length of 61 mm. The test arena was manually focused to record the experimental trials of each robot's cleaning process. The raw video files were then copied to a computer and converted into mp4 files. The converted files were then used as the input to a vision based benchmarking scheme.

2.3.4 Vision-based benchmarking scheme

The vision-based benchmarking scheme automatically computes the total area coverage performance of the tested robots by continuously tracking the area covered. The first processing step saves the first frame as a reference image for track map generation. The second phase of the benchmarking scheme detects the deployed robot and its position in every frame. The third step generates a track map by plotting the identified position of the robot on the reference image. Coverage area performance is then calculated using the overlapped positions of the robot in the reference image. This scheme is completely implemented in MATLAB (R2015a). Because the circular-shaped commercial cleaning robot possesses a round geometry, we used Hough transformation-based circle detection to determine the position of the geometric center of the robot. Once the robot's center was identified, a green colored circle with the same radius was replicated on the reference image corresponding to the center coordinates to generate the tracking map. Because hTetro has dynamic morphology, we used color-based blob detection and tracking approach wherein four red blobs detect the hTetro platform in each video frame. Using the detected blobs, the hTetro's coverage area can be identified, and green squares are plotted on the reference image over the complete video sequence. After generating the tracking maps on the reference images, the percentage of the area covered was calculated using Equation (1) for the first set of experiments and Equation (2) for the second round of experiments. In equations (1) and (2), the pixel area of the test robot can be computed by identifying the green colored pixels on the track map on the reference image. The images were purely used for benchmarking purposes to compare the area coverage performances of the considered robotic platforms, and our future work is set to explore the use of these images for global and local path planning.

$$Percentage\ of\ area\ covered = \frac{\text{Pixel area of the robot}}{\text{Total pixel area of the testing field}} \times\ 100 \tag{2.1}$$

$$Percentage\ of\ area\ covered = \frac{\text{Pixel area of the robot}}{\text{Total pixel area of the testing field} - \text{Total pixel area of the obtacles}} \times 100 \tag{2.2}$$

2.4 Results and Analysis

The experimental trials were started by placing the chairs and test robot at their initial positions. After the tests were completed, the recorded videos were post-processed to generate a track map using a stepwise vision based benchmarking scheme. This scheme used to track the commercial circular-shaped robot uses three steps to generate the robot's track map. The first step is a filtering process in which the frames are passed through a sliding window median filter for suppressing the effect of noise. The next step is to find the circle presented in every frame using a Hough transformation as shown in Fig 2. 11a (right), which also identifies the center of the circular-shaped robot. In the third step, a green colored circle with the same radius is duplicated on the reference image using the detected center as shown in Fig 2. 11a (left). These steps continue to the final frame, converting the reference image into a track map of the considered robot. The benchmarking scheme used for the hTetro and D-shaped robots comprised five steps to generate a track map. The first step is the filtering process used in the previous benchmarking scheme for the circular-shaped robot. In the second step, the red components in the image are separated by extracting the red layer from the RGB color space representation of the filtered image as shown in Fig 2. 10 (bottom right). Once the red layer is extracted, in the third step, a binary image is generated by mapping the 8-bit encoded pixel values in the red layer matrix to either zero or one based on a fixed threshold. The binary image generated in this manner contains white blobs that represent the presence of red-colored objects as shown in Fig 2. 10 (bottom left). The next step is blob detection and analysis of the binary image using a computer vision toolbox. To improve efficiency with prior knowledge and avoid false alarms, we excluded tiny blobs with an area of less than or equal to 200 pixels. The centroid obtained from the remaining blobs represents each block of the hTetro and 'D'-shaped robots as shown in Fig 2. 10 (top left) and Fig 2. 11b (left). The output of the previous step becomes the coordinates of the centroids. Using the detected blobs, the hTetro and D-shaped robots' areas are identified, and green squares are plotted on the reference image over the complete video sequence as shown in Fig 2. 10 (top right) and Fig 2. 11b (right).

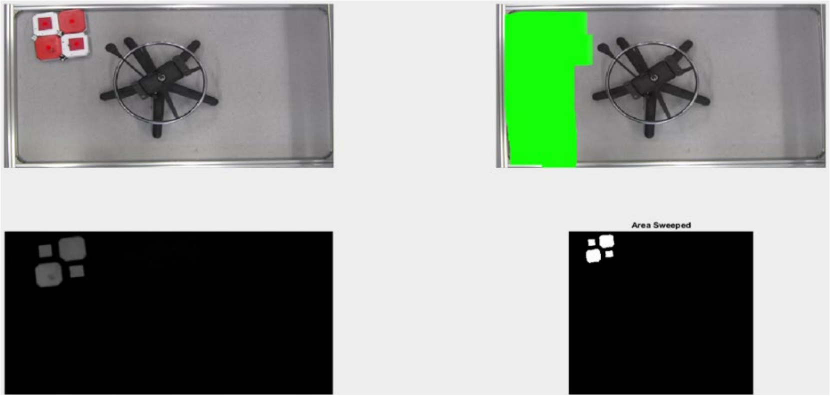
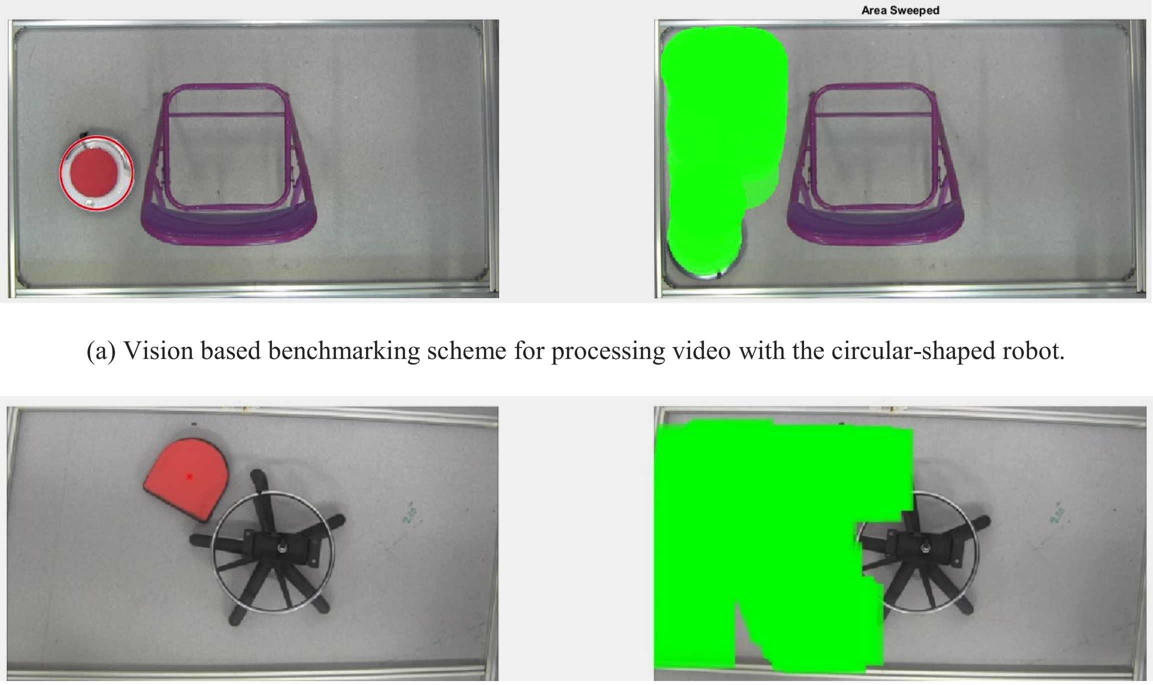


Figure 2.10. Vision based benchmarking scheme for processing video with hTetro.

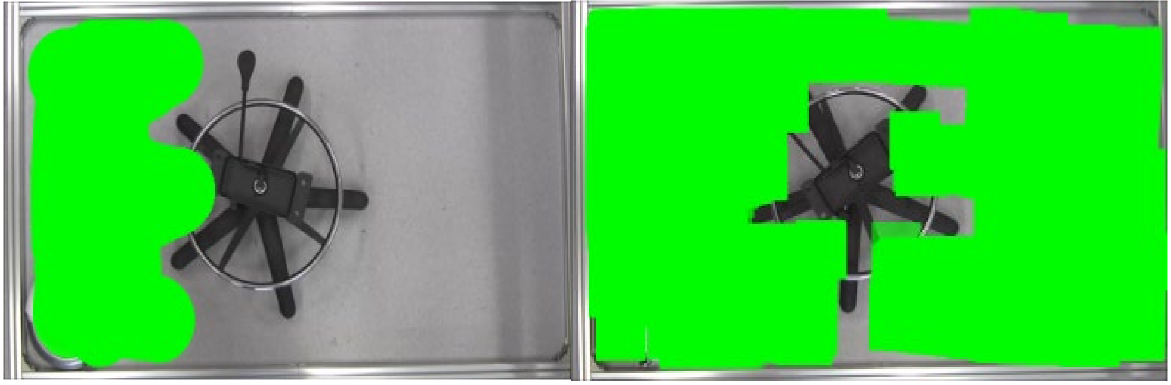


(a) Vision based benchmarking scheme for processing video with the circular-shaped robot.

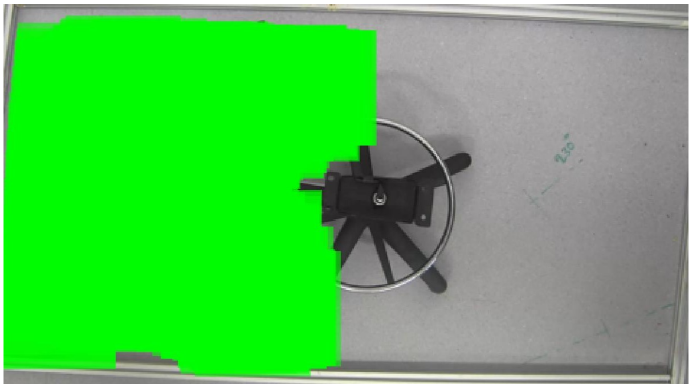
(b) Vision based benchmarking scheme for processing video with the D-shaped robot.

Figure 2.11. Vision based benchmarking scheme for processing video with fixed morphology robots.

Figs. 2.12(a), (b) and (c) illustrate the track map images of all the considered robots in the first set of experiments. The green colored shading represents the area covered by the robots. The percentage of the area covered was calculated with Equations 1 and 2. Fig 2. 13 shows the average percentage of the area covered for all the tested robots under each ranked chair in the first set of experiments. The processed images clearly indicate that hTetro has a significantly larger coverage area than the fixed morphology robots. Using its shape-shifting mechanism, hTetro covered more than 80% of the testing area of the highly-rated chair with a score of greater than eight, whereas both of the fixed morphology robots covered less than 50% of the set area. The overall coverage performance of hTtero across all the chair types was greater than 95%, whereas the circular-shaped robot achieved 68.27% coverage, and the 'D'-shaped robot achieved 77.64% coverage.



(a) Track map of the circular-shaped robot. (b) Track map of the hTetro robot.



(c) Track map of the D-shaped robot.

Figure 2.12. Resulting tracking maps on the reference image for the circular-shaped robot (a), hTetro (b) and the 'D'-shaped robot (c) around a piece of furniture ranked as 10.

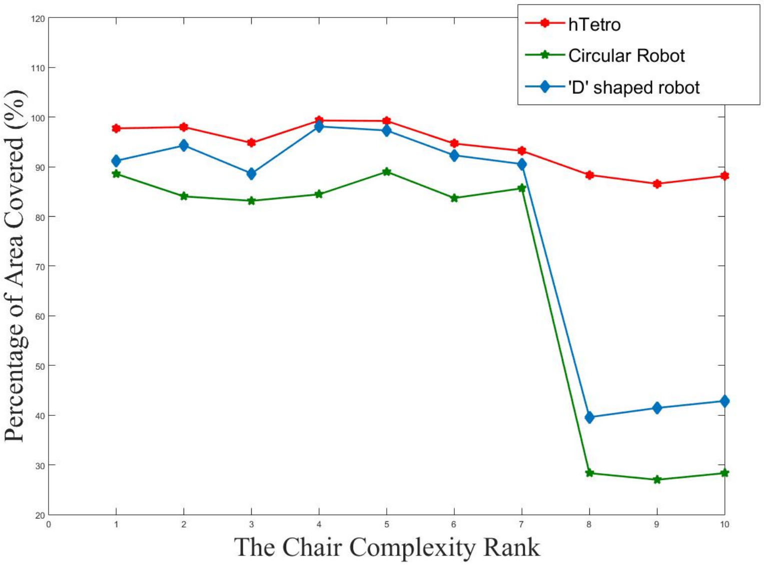
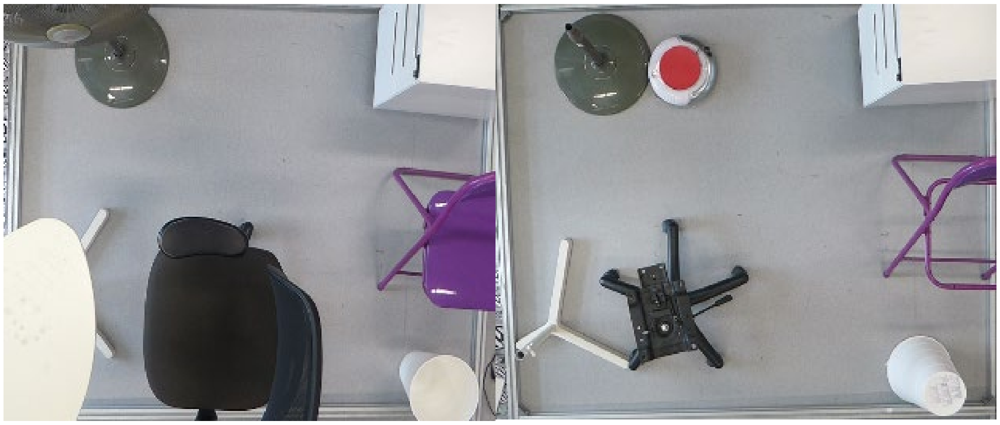


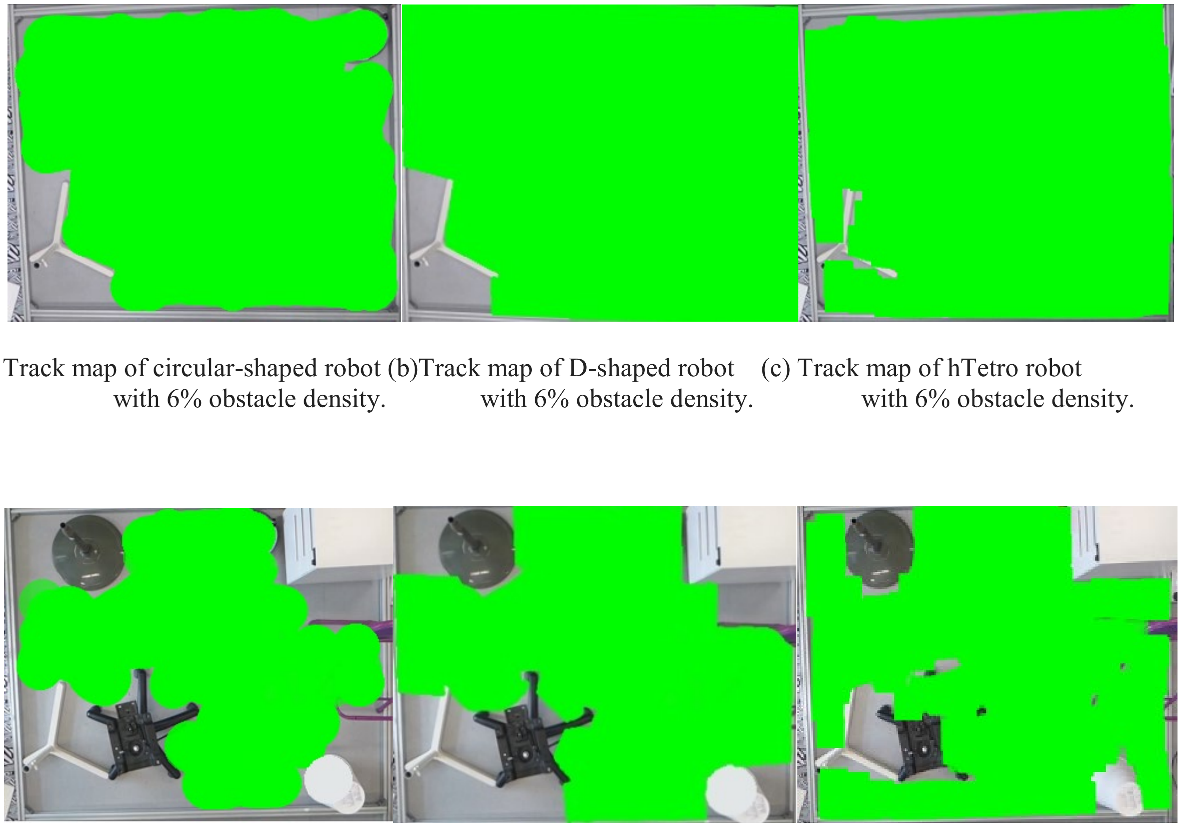
Figure 2.13. Percentage of the covered area of all the robots for first set of experiments.

In the second scenario, the boundaries of the testing field were extended, and the camera lens was adjusted to cover a new enlarged field of area 3.36 m^2. We created a mock office setup with six static obstacles inside the test scenario as shown in Fig 2. 14. The obstacle density was calculated by computing the total pixels occupied by all the obstacles placed in the testing field. The track maps of all the considered robots with an obstacle density of 6% are shown in Figs 2. 15 (a), (b) and (c). The percentage of the area covered in the second set of experiments was calculated only for the accessible area (ignoring the area occupied by the obstacles placed inside the scenario). The obstacle density was gradually increased, and the associated coverage performance of the robots was captured experimentally. Figs 2. 15 (d), (e) and (f) give the coverage performance of the robots with an obstacle density of 66%. The results indicate a significant drop in performance for both of the fixed morphology robots with increasing obstacle density. Fig 2. 16 shows the average percentage of the covered area for all the tested robots based on five runs for each level of obstacle density. The results indicate the superior coverage performance of hTtero in comparison with both the fixed morphology robots across all the experimental cases. The higher coverage performance of hTetro is mainly attributed to its ability to navigate narrow corridors and tight spaces by changing its morphology. The average coverage area for hTetro in the second set of experiments was 94.56%, whereas the average coverage area of the circular- and 'D'-shaped robots were only 80.52% and 85.98%, respectively.



(a) (b)

Figure 2.14. Testbed setup for the second set of experiments before (a) and after (b) removing the top surface of the furniture.



(a) Track map of circular-shaped robot with 6% obstacle density.

(b)Track map of D-shaped robot with 6% obstacle density.

(c) Track map of hTetro robot with 6% obstacle density.

(d) Track map of circular-shaped robot with 66% obstacle density.

(e)Track map of D-shaped robot with 66% obstacle density.

(f) Track map of hTetro robot with 66% obstacle density.

Figure 2.15. Resulting tracking map on the reference image for the circular-shaped robot (a & d), the 'D'-shaped robot (b & e) and hTetro (c & f), under increasing obstacle density.

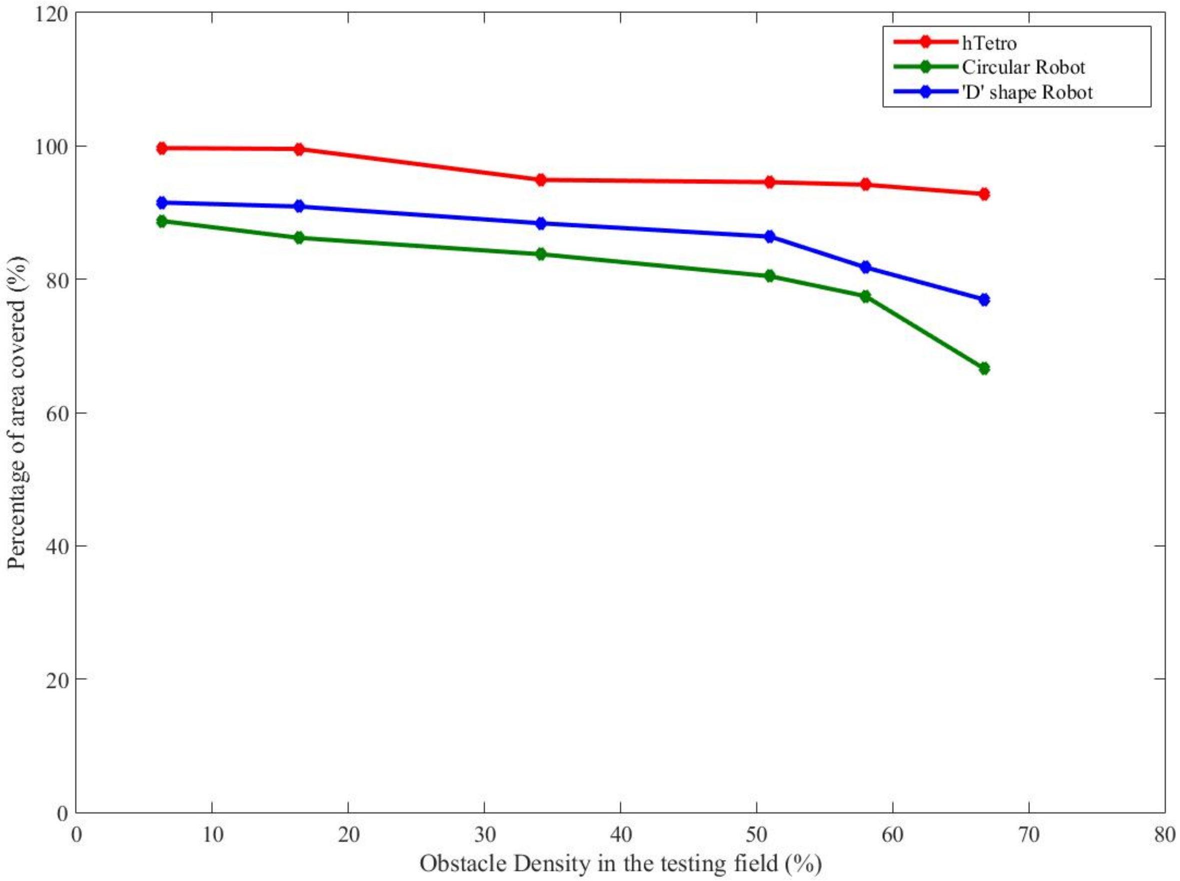


Figure 2.16. Percentage of area covered in the second set of experiments

2.5 Conclusion

In this Chapter, we have presented a novel reconfigurable Tetris-inspired floor cleaning robot called hTetro that implements the hinged dissection of polyominoes. We introduced the mechanical design, electrical system and HMI modules of the hTetro robot and successfully validated its ability to transform between any of seven one-sided tetromino morphologies to maximize floor coverage. Experiments were performed in two different settings to systematically compare the coverage area performance of the developed hTetro robot with two commercially available circular and 'D'-shaped fixed morphology robot platforms. We tracked the test robots using an overhead camera on top of a constructed testbed to generate the robots' track MAPS. The track maps generated from each set of experiments were post-processed to measure the area covered by the robots. In both sets of experiments, the hTetro robot exhibited a significant performance advantage in terms of floor coverage area due to its ability to assume optimal morphologies while navigating its environment.

Future research work is set to mainly focus on the following areas: the first is to integrate range and bump sensors to enable autonomous navigation in the hTetro robot. The second is to study the application of Polyomino tiling theory towards facilitating global path planning through automatic generation of global tiling set required to cover a defined space. Moreover, we intend to implement highly adaptive locomotion control that allows for smooth trajectories at low computational costs across all seven configurations.

3. Novel Motion planning techniques for a Tetris inspired reconfigurable robot.

In the process of developing a floor cleaning robot, it is critical to address its capacity to demonstrate autonomous and efficient coverage path planning. Similarly in the case of hTetro, it is crucial to establish its coverage path planning intelligence. Coverage path planning is an interesting field of study for robotic scientists with numerous research literature. One frequent technique that has been extensively utilized for coverage path planning task is cellular decomposition. This method breaks down the workspace into small cells and applies motion planning to each cell to cover the entire region. There are numerous other methods have been proposed to decompose the given space that includes trapezoidal decomposition [76], boustrophedon decom- position [77], Morse-based cellular decomposition [78] among others. S.C. Wong and MacDonald proposed a topological area coverage technique that could point landmarks as nodes to cover the area [79]. In another work, a sensor-based coverage technique was proposed by Butler et al [80]. The proposed method uses sensed robot data to generate paths that could reach most possible point to achieve maximum area.

Grid-based area coverage methods is in use over the last three decades wherein, the workspace is treated as uniform grid cells, and each cell contains a value which replicates presence of an obstacle. There are numerous algorithms proposed to- wards realizing grid-based coverage such as wavefront algorithm [81], spanning tree technique [82], hexagonal grid de- composition [83], and neural network-based coverage [84]. Xu [85] presented a graph based coverage path planning technique in which, he considers the mapped area as a graph and applies robot motion planning to reach every coordinate point in the graph. In the recent years, a number of 3D area coverage techniques have been proposed and validated in the context of service robotic platforms. For instance, Cheng et al. applied 3D coverage to inspect urban structure [86], E. Galceran, M. Carreras proposed a bathymetric 3D map to inspect ocean floors [87], Jin and Tang proposed a 3D coverage algorithm for agriculture purpose [88]. Other coverage path planning techniques like optimized 3D coverage, and multi-robot coverage strategies were discussed in [89]. Even though there are numerous literature that demonstrates the application of coverage path planning techniques in robotics systems, the current approaches are mostly limited to fixed morphological platforms. Also, none of these existing techniques were realized on a reconfigurable robot in the context of a meaningful application.

In most cases with Coverage Path Planning (CPP) strategies, the commonly used motion planning algorithms are spiral motion, backtracking spiral motion, and boustrophedon (back- and-forth) motion. For instance, Jian Jin and Lie Tang proposed a decomposition method in order to cover the farming field using boustrophedon paths [90]. Similarly, I. A. Hameed et al proposed a genetic algorithm for coverage path planning and used boustrophedon paths for coverage [91]. A time and energy efficient online coverage path planning technique is proposed in [92] wherein, they adopt a high-resolution grid map representation and utilized spiral path motion to perform efficient coverage. In another work, Y. Gabriel and E. Rimon applied the spiral motion in a cellular decomposition coverage technique [82]. E. Gonzalez et al utilized backtracking spiral motion in an approximate cellular decomposition area coverage method [93]. Also, they proposed an improved spiral algorithm in order to consider the obstacles inside the grid space [94]. Even though, there are numerous literature demonstrating the advantages of existing motion techniques in the context of CPP, there is a huge gap in implementing those techniques on robots with the reconfigurable base. In particular, hTetro robot requires a unique motion planning algorithm in order to achieve a better performance.

In this Chapter, we are proposing a novel motion planner for the Tetris mimicked reconfigurable floor cleaning robot that creates waypoints to navigate on the generated tileset with an objective of covering the unified area. The process of the pro- posed motion planner and its execution of path generation to cover the entire area is divided into a set of different stages. The first stage is the planning stage where the high-level global coverage planning will be performed based on the tiling theory. The second set is the generation stage which produces the trajectory (i.e. Motion planning) to cover the area. The last stage set is the execution stage where the robot performs navigation with appropriate morphology. In this chapter, we reviewed the fundamental challenges of the proposed scheme such as realizing the generation of the tileset, motion pattern, reconfigurable ability and translating the tiling theoretical approach through the analytical process into demonstrable systems. This chapter discus all these aspects and concludes with experimental results that validate the efficiency of proposed approach through systematically benchmarked its performance with few conventional motion planning techniques used in CPP (i.e. Spiral, boustrophedon motion, greedy search, and random search [95] - [99]) concerning total distance traveled, and are recovered. The motion planner herein presented is a critical effort towards designing a self-reconfigurable robot that is capable of autonomously produces a global tileset for the deployed environment, determining correlated global and local feasible trajectories, and generating relevant motor signals based on inverse kinematics and dynamics model of the robot's each configuration.

3.1 hTetro: A Reconfigurable Cleaning Robot

hTetro is a Tetris inspired robotic floor cleaner which is able to change its morphology to any of the seven one-sided tetromino pieces as shown in Fig 3. 1. The concerned robot was developed under the principle of hinged dissection of polyominoes [70], [71]. The hTetro robot consists of four equivalent square blocks which are connected by three hinged points with LLR (Left Left Right) configuration. Each hinge point is horsed with a smart servo which enables the shapeshifting capability. Two hinge servos are mounted on block-2 and one more connected to block-4. Each block is attached with a set of loco- motion components and vacuuming modules. Since block-2 is connected with two hinged motors, it acts as an anchor point for the robot. Also, block-2 consists of all control circuits, controller, processor, and Lidar sensor. In this chapter, the local and global references of the hTetro robot were fixed on block-2. The following section details the hTetro robot's system model setup on a workspace.

Variables	Description
q	Commands to hTetro without orientation information
Q	Commands to hTetro with orientation information
θB^G	Global Orientation
T_v	Tile Value
M	Empty Matrix
M^t	Tiled Matrix

3.1.1 Workspace Model

Let's consider the workspace as W wherein the concerned robot A is navigating. The reference frame F for the workspace

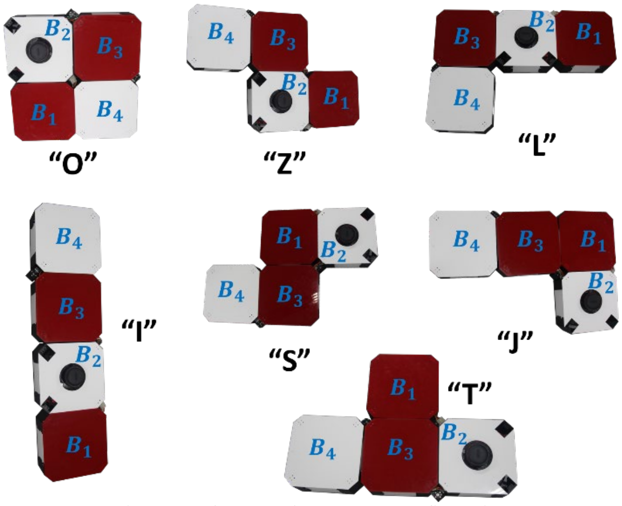


Figure 3.1. hTetro robot's seven configurations

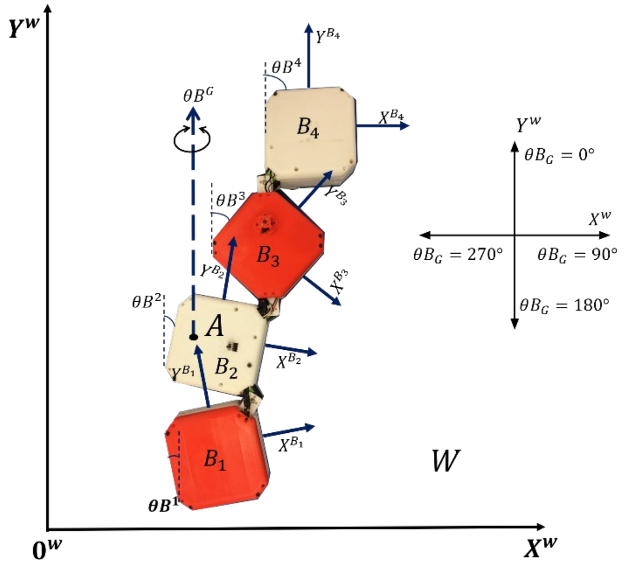


Figure 3.2 hTetro Robot's System Model on Work Space

and the robot is expressed as FW ,and FA. The dimension of the Cartesian work space is denoted as YW for Y coordinate, and XW for X coordinate. The defined workspace is then disintegrated into square grids, wherein each grid cell size (denoted as dgrid) is identical to the size of each block of the hTetro robot. The total row and column size of the decomposed grid be like nrow and ncol. Finally in the grid space, the obstacles presence in the each cell are indicated as O. The figure of the model is illustrated in Fig 3. 2.

3.1.2 hTetro On A Workspace

Since the hTetro robot consist of four blocks and each of those occupies a single cell in the grid, we have given a geometric representation for all four blocks as B1, B2, B3, B4. Where θBn(n = 1 to 4) serve as a rotational angle for each block with respect to the workspace frame. The possible rotational angles that can be achieved by each block locally are θBn = 0, $\frac{\pi}{2}$, π. However, the angle θB_2 will be constant whereas the block 2 act as a foothold for the whole hTetro platform in which the *FA* is tagged. The variations in θB_n aids the hTetro robot to achieve reconfiguration between any of the seven one-sided Tetris pieces. Unlike the traditional definition, it requires a unique representation for the hTetro to identify its position or grid coordinates (*X*, *Y*) on a workspace. Robot Pose: (1) The pose q (1) of an hTetro robot is a six-element array (2) The Orientation of the robot θB_n with respect to the workspace frame.

$$q = [X, Y, \theta B_1, \theta B_2, \theta B_3, \theta B_4]^T \qquad (3.1)$$

Where:

(X; Y) = coordinate of hTetro Block 2 (B2) center in workspace frame F^W
θB^G = Global orientation of the hTetro robot which is also centered to (B2) with respect to F^W.

Similarly, it is critical to acquire the robot's grid position information which provides the i^{th} row, and j^{th} column. While considering the hTetro robot, we get four grid position which is four block information at any point of the operation. Let's consider G as the total grid generated with n_{row}, and n_{column}, then each grid information is represented as $G_{i,j}^W$ with respect to the workspace. The relation between the grid representation and the workspace coordinates is given in equation (2) & (3).

$$G_{i,j}^W = [X_{i,j}^W, Y_{i,j}^W]^T \qquad (3.2)$$
$$G_{i,j}^{B^n} = [X_{i,j}^{B^n}, Y_{i,j}^{B^n}]^T \qquad (3.3)$$

$G_{i,j}^W$ is the reference point which the robot A will clear while navigating.

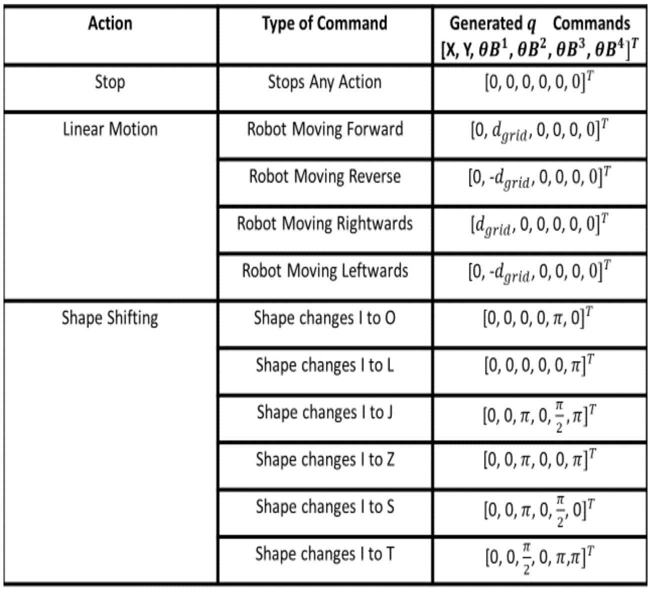
When there is an obstacle O_l introduced on the work space, each $G_{i,j}^{W}$ contains a value that replicates the obstacle presence. For instance, O_l = $G_{i,j}^{W}$ $\in \{0, 1, -1\}$ i.e. free cell, obstacle cell, and undefined cell. The concern robot can only navigate on the free grid space.

3.1.3 hTetro Robot Navigation On The Workspace:

The hTetro robot consists of navigational components in almost each block which benefit smooth locomotion. Since the concern robot's base is reconfigurable, achieving differential movement is very challenging. Hence, we developed linear locomotive gaits which cause the robot to traverse forward, backward, leftward, and rightward motions. In order to achieve the mentioned motion capability, we established omnidirectional wheels in each block of the hTetro robot. The proposed motion planning strategy produces a series of waypoints WP in order to accomplish the motion planning to cover the entire area. Dissimilar to the traditional robotic pose, hTetro has its own definition to represents its current stare. For instance, it requires position (X, Y), orientation θB^{G}, and configuration q to define its state. The each generate WP points consist of all the mentioned parameter. Once the information is passed to the controller, it produces necessary motor signals to achieve action in order to complete the area coverage task.

In order to reduce unblocked areas that could potentially be identified as obstacle grids while constructing the grid map, the smallest grid size is defined to match the size of a hTetro block ($d_{grid} = d_{block}$), which provides the highest resolution of the grid map with each grid being geometrically coverable by hTetro blocks. Under this definition, the robot is capable of performing either a linear motion or an angle adjustment of a block. Performing a linear motion at a time instant moves all robot blocks simultaneously in one of the four directions for a grid length d_{grid} with respect to the workspace frame F^{W} neighbor block's local frame. The hTetro robot's action command q to accomplish linear and angular motion is shown in table 3.1.

Table 3.1. hTetro Robot's Command q .



Action	Type of Command	Generated q Commands $[X, Y, \theta B^1, \theta B^2, \theta B^3, \theta B^4]^T$
Stop	Stops Any Action	$[0, 0, 0, 0, 0, 0]^T$
Linear Motion	Robot Moving Forward	$[0, d_{grid}, 0, 0, 0, 0]^T$
	Robot Moving Reverse	$[0, -d_{grid}, 0, 0, 0, 0]^T$
	Robot Moving Rightwards	$[d_{grid}, 0, 0, 0, 0, 0]^T$
	Robot Moving Leftwards	$[0, -d_{grid}, 0, 0, 0, 0]^T$
Shape Shifting	Shape changes I to O	$[0, 0, 0, 0, \pi, 0]^T$
	Shape changes I to L	$[0, 0, 0, 0, 0, \pi]^T$
	Shape changes I to J	$[0, 0, \pi, 0, \frac{\pi}{2}, \pi]^T$
	Shape changes I to Z	$[0, 0, \pi, 0, 0, \pi]^T$
	Shape changes I to S	$[0, 0, \pi, 0, \frac{\pi}{2}, 0]^T$
	Shape changes I to T	$[0, 0, \frac{\pi}{2}, 0, \pi, \pi]^T$

3.2 Motion Planner Framework For hTetro Robot

This chapter presents a novel motion planner framework (Fig 3. 3) for the hTetro robot over complete coverage tasks where the choice of feasible global, local trajectory, and morphology of the robot is critical for achieving the set performance targets. The proposed framework could be scalable across intra-, inter- and nested reconfigurable systems with minimal modifications according to its application. It allows a concerned robot to evaluate the geometry of the environment, compute desired body morphology, selects associated local and global optimal trajectories, and generate appropriate motor primitives. The proposed framework subsist of stages, and each of those architectures is detailed in the following sections.

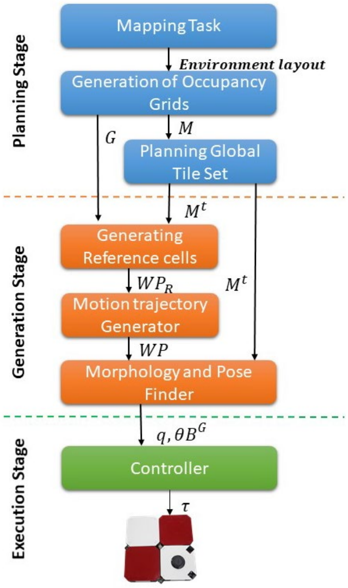


Figure 3.3. The Proposed Motion Planning Frame Work

3.2.1 Planning stage:

Sensing the environment accurately is critical to perform a better area coverage path planning. The process of generating the map of test space is the initial task in our proposed framework. Once the map is generated, it will automatically give out the total no of grid cells it occupies. Each cell ($G^W_{i,j}$) in the produced grid contains a value which replicates the mapped environment. We accumulate all the cell value in order to generate matrix M^t. The algorithm that generates the matrix from the occupancy grid space is detailed in Algorithm 1. The produced Matrix is then passed to the next sub stage in order to plan the global tileset. Before placing the Tetris tiles on the generated matrix, the Global tile planner adopts a definite tiling theorem to reduce the no of pieces that are utilized from the available one-sided Tetris set. The process of placing the tile pieces will be explained in section 3.
$M^t_{i,j} = G^W_{i,j}$ (i.e. the row and column information are same in both matrix and grid)

Algorithm 1 Convert Grid Space to Matrix M^t

```
Function GET MATRIX ($M^t$)
  $m \leftarrow n_{row}$, $n \leftarrow n_{column}$, $M^t[] \leftarrow [n_{row}, n_{column}]$
  $i \leftarrow 0, j \leftarrow 0$;
  for all $i$, $i \leftarrow 0$ to $n_{row}$ do
    for all $j$, $j \leftarrow 0$ to $n_{column}$ do
      if ($G^W[i][j] = 0$] then
        $M^t[i][j] \leftarrow 0$
      else
        $M^t[i][j] \leftarrow -1$
      end if
    end for
  end for
End function
```

3.2.2 Generation Stage:

Next, we use the tileset matrix to generate the navigation path that achieves maximum coverage. This stage is further divided into subtask that simplifies the process of area coverage for the concern robot. Since the hTetro robot could cover 4 grid cells at any point in time, it is critical to find the reference cells which acts as a waypoint for the system to navigate. The reference generator marks the reference cells which can be a possible waypoint for the hTetro robot. The motion trajectory generator executes the plan of sequencing the reference cells. The accomplishment of the reference point arrangements produces navigation plan for the hTetro robot. Also, this task produces the waypoints WP that consist of pose and morphology information of the robot and passes it to the next level. The next task is finding the current pose and morphology of the robot and generate appropriate actions to achieve the targeted state.

3.2.3 Execution Stage:

In the execution stage, the controller receives the series of states from the generation layer. After receiving the state information, the controller executes navigation and morphology shifts. We use a feedback controller which could generate the motor primitives with respect to the current pose of the robot in order to achieve the received state. Similarly, the controller could perform the same set of actions for all provoked states which ultimately leads the robot to cover the entire area. In this chapter, we are mainly discussing on the planning, and generation stage. Execution stage is considered as a control theory problem which we discuss in our extension of this work.

3.3 Strategy For Generating Tile Set Based On Tiling Theory

Implementing existing area coverage approaches on the reconfigurable platform conceives complications due to its geometrical changes in every configuration. To overcome this bottlenecks in reconfigurable platforms, it requires a unique approach for area coverage task. Specifically, the case with the hTetro robot we used polyomino tiling concept as an area coverage technique. A polyomino tiling is a process of placing tiles in a plane using any number of polyominoes pieces which is widely used in gaming [100] and rendering purposes [101]. Since the hTetro robot has seven different shapes, it is inefficient in terms of power and computation, when we use all seven configurations to cover the given area. So, we implemented Tetris tiling theorems to reduce the no of shapes that are used to cover the given area. Tetris tiling theorems is a branch of mathematics which provides a possible tileset to cover a given A x B rectangular grid space. We discussed the details of a few tiling theorems, and its application on the hTetro robot in terms of complete coverage in our previous work [93]. However, the details of the algorithm that generates the tile set in a given 2D rectangular grid space were not addressed. Next section introduces the algorithm to generate tiles in the produced matrix M^t.

3.3.1 Choosing a Theorem:

Before going to the tiling process, it is crucial to choose the possible Tetris shapes to fill the given area. In order to achieve that, we have to first choose the best tiling theorem. The best way to choose a tiling theorem is to find the size of the provoked matrix M^t**.** Let m and n be the total no of rows and columns of matrix M^t. Since we are considering Tetris pieces which consist of four equivalent squares, any rectangular grid that is divisible by 4 can be tiled fully with any combination of Tetris pieces. For instance, in the considered matrix, if mxn is divisible by 4, then we can have complete tile. In order to reduce the no of tile pieces, we used the theorem (TH1) stated below. Also, it is important to count the no of elements in the matrix M^t which consist of obstacles. If the no of elements that consist of obstacles is also divisible by four, then we can apply the below theorem. Fig 3. 4 illustrates the proof of the theorem (TH1).

An $a \times b$ gridded rectangle can be tiled by $'T','S','Z'$ Tetrominoes if, and only if, $a, b \geq 4$ and either
1. One side is divisible by 4, or
2. $a, b \equiv 2\ (mod\ 4)$ And $a + b > 16$. **(TH1)**

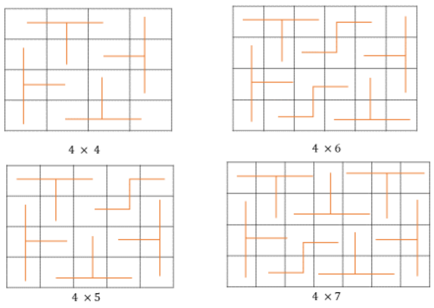


Figure 3.4. Proof of Theorem 1

While mapping an unknown environment, it is impractical to expect the total grid occupied could be a factor of 4. In such cases, we must use other relevant theorems to tile the defined space. If the dimensions are not exactly a factor of 4, then we will not obtain a complete tile. We may have to leave one to two cells untiled in all possible situations. When the theorem is applied to the generated matrix M^t one to two matrix element's values are neglected to default. These elements may also be considered as an obstacle. Other relevant Tetris tiling theorems that can be chosen in the above-mentioned condition are furnished below. Fig 3 5. & 3.6. details the theorem-proof for (TH2) and (TH3)

Let a and b be integers that are strictly greater than 1. As such, the $'T',' S',' Z'$ Tetrominoes can tile a modified rectangle of side a and b, if, and only if, either:[102]
1. a≡2 (mod4)and b is odd, or
2. b=2 and a≡1 (mod 4). **(TH2)**

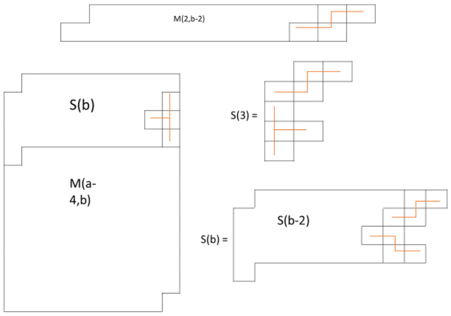


Figure 3.5. Proof of Theorem 2

The $'O',' L',' J'$ tetrominoes can tile a deficient rectangle of side a and b, only if
1. The a and b is odd (2n+1)X(2n+1), n≥1, and
2. The missing cell has to be on the odd position if the rectangle is (4m + 1) × (4m + 1) m ≥ 1and in an even position if the rectangle is (4m + 3) × (4m + 3) m ≥ 1. [103] **(TH3)**

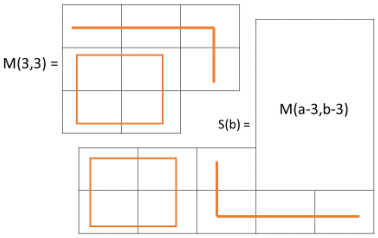


Figure 3.6. Proof of Theorem 3

A. Placement of Tiles:

The first step in the process of tiling the matrix M^t is to find the possible placements of all selected pieces. When the selected pieces are more, then it requires excess time and memory to complete the tiling action. However, this chapter only concentrates on the feasible solutions than the optimal ones. In order to obtain the appropriate tileset, we utilized bipartite relation between the placement and the matrix elements [104]. For instance, Fig 3. 7 describes from association of the placement graph wherein we restricted to Tromino (a type of polyomino that contains three equivalent square blocks) for better understanding. The figure illustrates a $2x3$ matrix m_e, and all possible placements of L-Tromino shapes. Each L-Tromino piece will occupy three matrix elements, and produce eight possible placement combinations.

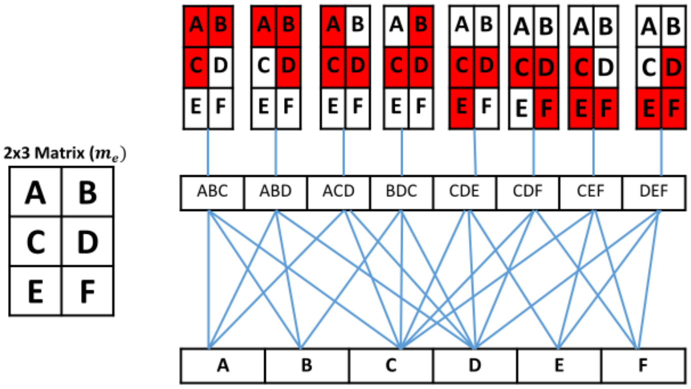


Figure 3.7. Tile Placement Map

The next step is to search the best element to place the tile which illustrated in Fig 3. 8. We use a backtracking algorithm to search the best element $m_{e_{i,j}}$ in the matrix to place the selected tile. Once the placement of tile is established, the algorithm adds the implanted matrix items $m_{e_{i,j}}$ in the solution set and tries to tile the rest of the matrix items. If the algorithm couldn't manage to place the next tile, then it tries other possible placements. Even after trying all viable placements, when the algorithm departs from tile combinations, then it backtracks to the previous matrix element and continues the same procedure with the new tile piece. The same process continues until we get the fully tiled matrix solution. The concern tile placement procedures are also applicable to the grid space with obstacles.

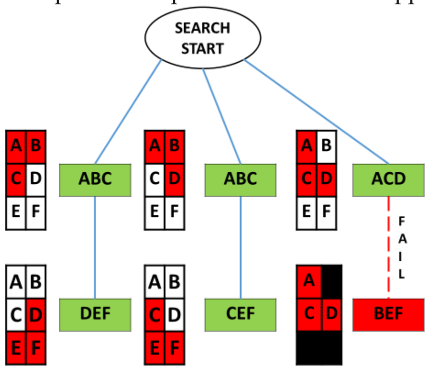
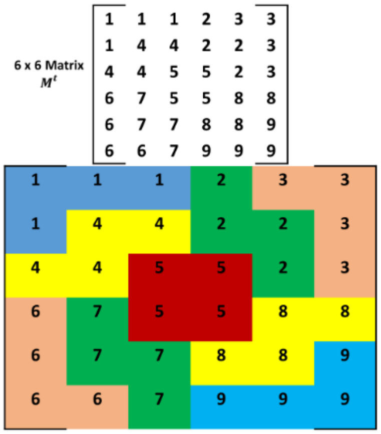


Figure 3.8. Tile Placement Searching

B. Generation of tile on matrix M^t:

The matrix $M^t$was generated in the planning stage after the stage of occupancy grids. As explained previously, each element of M^t holds a value (0, & -1) that replicates the mapped environment. When the algorithm starts to tile placing process, it will check the elements value $M^t_{i,j}$ before placing the tile. The algorithm only considered the elements that are $M^t_{i,j} > -1$ $and < 1$ in order to establish the placement of tile. If the element $M^t_{i,j} < 0$, then the proposed placing method will eliminate the item for tile placements and proceed to search for the next possible placement item. After placing each tile pieces on the matrix, the algorithm will number (P_n) the placed tiles in ascending order until the algorithm exit by ran out of placements. The detailed algorithm of tile placement on matrix M^t is presented in Algorithm 2. Random tile matrix that was generated using the above algorithm is shown in Fig 3. 9.


6 x 6 Matrix M^t

$$\begin{bmatrix} 1 & 1 & 1 & 2 & 3 & 3 \\ 1 & 4 & 4 & 2 & 2 & 3 \\ 4 & 4 & 5 & 5 & 2 & 3 \\ 6 & 7 & 5 & 5 & 8 & 8 \\ 6 & 7 & 7 & 8 & 8 & 9 \\ 6 & 6 & 7 & 9 & 9 & 9 \end{bmatrix}$$

Figure 3.9. The Random Tile Set Generated

Algorithm 2 Placing tile in matrix M^t

```
function PLACETILE($M^t$)
    $c_t \leftarrow m \times n; c_n \leftarrow 0$
    $p_n, n_t \leftarrow 1$
    if $mod(c_t, 4) = 0 \wedge mod(c_t, 4) = 2$ then
        Choose Theorem TH1
        $T_1[] \leftarrow tileT$
        $T_2[] \leftarrow tileZ$
        $T_3[] \leftarrow tileS$
    else if $mod(c_t, 4) = 1$ then
        Choose Theorem TH1
        $T_1[] \leftarrow tileO$
        $T_2[] \leftarrow tileL$
        $T_3[] \leftarrow tileJ$
    end if
    while $c_n > c_t$ do
        for all $i \leftarrow 0$ to $n_{row}$ do
            for all $j \leftarrow 0$ to $n_{col}$ do
                if $M^t[i][j] = 0$ then
                    while $p_n = p_n + 1$ do
                        Try place tile $T_{n_t}[]$
                        if (!Placed) then
                            Try place tile $T_n[]^T$
                            if (!Placed) then
                                $n_t = n_t + 1$
                            end if
                        else
                            Try place tile $(T_n[]^T)p_n$
                            $p_n = p_n + 1$
                            $c_n = c_n + 1$
                        end if
                    end while
                end if
            end for
        end for
    end while
    $V_n \leftarrow p_n$
end function
```

3.4 Strategy for Motion planning:

In most grid-based coverage path planning techniques, each cell represents the full robot (i.e. each cell is robot sized). So the path planning scheme just needs to pass the grid coordinates to achieve the full coverage. Since the hTetro platform is unique in its reconfiguration ability and it covers four cells in a grid at any point of time, the path planning technique should generate reference coordinates. The generated coordinates will be tagged with block-2 ($B2$) for all seven hTetro's configurations. These coordinates will acts as a waypoints WP for the robot that aids to achieve full coverage. On the other hand, the previous coverage motion planning techniques never concern the robot's orientation by reason of their fixed morphology. Positioning the heading angle in every area coverage platforms is treated as a control problem. Whereas with the hTetro robot it also requires an orientation reference, due to its directional shift that takes effect after every morphological movement. Acquiring the tiling matrix M^t and G information as an input for this stage of operation to produce a series of WP which consist of hTetro robot's state Q. The formulation to generate state Q is given in equation (4).

Where, $Q = [q[] + \theta B^G]$ (3.4)

3.4.1 Generation of Reference Cell:

For better reference cell identification we required two information that includes, the tile type and its placement orientation. This two information we can acquire from the produced M^t matrix. As we know, each tile pieces that were placed in the matrix M^t holds a unique number P_n in ascending order which is also the R value for each element on the tile. Every R value should occupy four different elements $M^t_{i,j}$ in the tiled matrix. Let $i \; and \; j$ be the i^{th} row and j^{th} column of matrix M^t. The first step of our process is to get the all four element's data that holds the value R by performing row-wise search, from which we can bring in the two basic information. At the completion of searching process, we get four sets of data which is denoted as $M^t_{i_k j_k}$ where $k \in \{1, 2, 3, 4\}$. In each element set $M^t_{i_k j_k}$, there will be a row and column values which are stored on another variable as i^R_k and j^R_k. Using those values we calculated the tile value T_v (5) from which we can get tile type and orientation information Shown in Fig 3. 10. All viable T_v and its corresponding Q values are given in Table 3. 2. Once the process is completed, all the information will be stored in array WP_R. The Algorithm. 3 again starts the same procedures by increasing the R value until it runs out of tiles. The generated reference cells for the random matrix which was illustrated in the last section is shown in Fig 3. 11.

$T_v = ((i^R_1\text{-}i^R_2) + (i^R_2\text{-}i^R_3) + (i^R_3\text{-}i^R_4) + (j^R_1\text{-}\, j^R_2) + (j^R_2\text{-}j^R_3) + (j^R_3\text{-}j^R_4))$ (5)

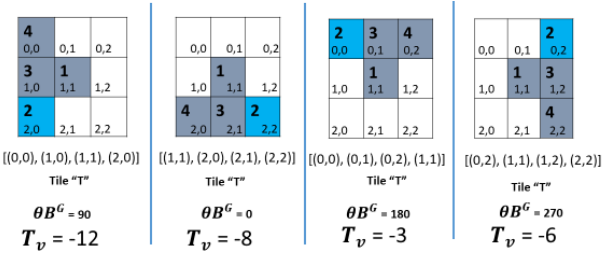
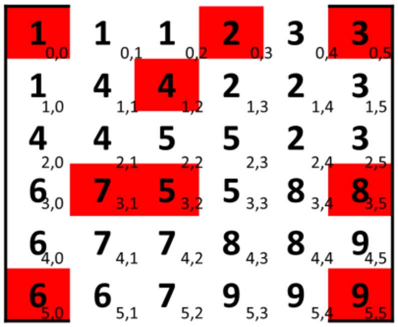
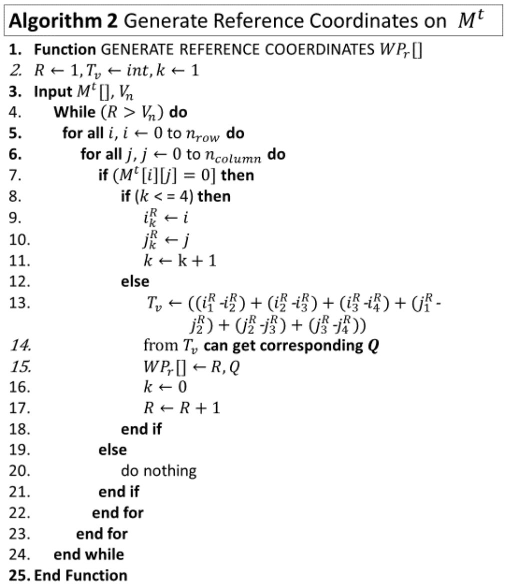


Figure 3.10. The T_v and θB^G Value for Tile 'T'



🟥 1 0,0	1 0,1	1 0,2	🟥 2 0,3	3 0,4	🟥 3 0,5
1 1,0	4 1,1	🟥 4 1,2	2 1,3	2 1,4	3 1,5
4 2,0	4 2,1	5 2,2	5 2,3	2 2,4	3 2,5
6 3,0	🟥 7 3,1	🟥 5 3,2	5 3,3	8 3,4	🟥 8 3,5
6 4,0	7 4,1	7 4,2	8 4,3	8 4,4	9 4,5
🟥 6 5,0	6 5,1	7 5,2	9 5,3	9 5,4	🟥 9 5,5

Figure 3.11. The Produced Reference Point on Random Tile set



```
Algorithm 2 Generate Reference Coordinates on $M^t$
Function GENERATE REFERENCE COOERDINATES $WP_r[]$
$R \leftarrow 1, T_v \leftarrow int, k \leftarrow 1$
Input $M^t[], V_n$
    While $(R > V_n)$ do
      for all $i, i \leftarrow 0$ to $n_{row}$ do
        for all $j, j \leftarrow 0$ to $n_{column}$ do
          if $(M^t[i][j] = 0]$ then
            if $(k <= 4)$ then
              $i_k^R \leftarrow i$
              $j_k^R \leftarrow j$
              $k \leftarrow k + 1$
            else
              $T_v \leftarrow ((i_1^R - i_2^R) + (i_2^R - i_3^R) + (i_3^R - i_4^R) + (j_1^R - j_2^R) + (j_2^R - j_3^R) + (j_3^R - j_4^R))$
              from $T_v$ can get corresponding $Q$
              $WP_r[] \leftarrow R, Q$
              $k \leftarrow 0$
              $R \leftarrow R + 1$
            end if
          else
            do nothing
          end if
        end for
      end for
    end while
End Function
```

3.4.2 Sequencing Of Reference Cells:

This section discuss the strategy of sequencing the reference points generated so that the hTetro robot just follows the order as a roadmap. In order to sequence the generated reference points, we are using a technique which is very similar to the grid-based wavefront algorithm. In this algorithm, we are using the WP_R array as an input wherein we get the initial reference point. Further, we are using the R value from the matrix M^t to complete this process. Altogether we are arranging the series of R values and its represent reference elements. The algorithm assumes that the initial value of R is 1 and its corresponding reference element as a default starting point for the hTetro robot. Once we have the initial reference element, with respect to that item, the algorithm looks for the smallest R value in the neighboring elements. So the algorithm literally performs a R value comparison between every neighboring element Fig 3.12(Top Left). The total set of R values that are compared can be determined by the current reference location. There are three possible value sets could be compared which is determined as shown in Fig 3.12 (Top, and Bottom Left). While the performing the search, the algorithm naturally eliminates the R value which is already visited. Also, if the proposed algorithm couldn't find the smallest value at some point in the middle of the process, it will start the row-wise, and column wise research to find the next R value and its reference element as in Fig 3.12(bottom Right). After finding the smallest value and its corresponding reference point the value is listed in waypoint array WP. The algorithm will continue until when it runs out of R Values. The detailed algorithm of sequencing the waypoints is elaborated in Algorithm. 4.

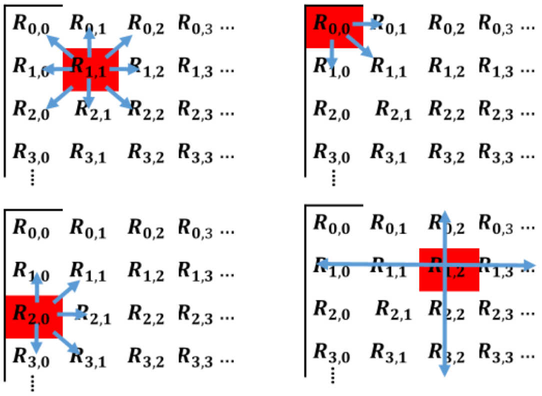

$R_{0,0}$ $R_{0,1}$ $R_{0,2}$ $R_{0,3}$ ...
$R_{1,0}$ $R_{1,1}$ $R_{1,2}$ $R_{1,3}$ ...
$R_{2,0}$ $R_{2,1}$ $R_{2,2}$ $R_{2,3}$ ...
$R_{3,0}$ $R_{3,1}$ $R_{3,2}$ $R_{3,3}$ ...
$R_{0,0}$ $R_{0,1}$ $R_{0,2}$ $R_{0,3}$ ...
$R_{1,0}$ $R_{1,1}$ $R_{1,2}$ $R_{1,3}$ ...
$R_{2,0}$ $R_{2,1}$ $R_{2,2}$ $R_{2,3}$ ...
$R_{3,0}$ $R_{3,1}$ $R_{3,2}$ $R_{3,3}$ ...
$R_{0,0}$ $R_{0,1}$ $R_{0,2}$ $R_{0,3}$ ...
$R_{1,0}$ $R_{1,1}$ $R_{1,2}$ $R_{1,3}$ ...
$R_{2,0}$ $R_{2,1}$ $R_{2,2}$ $R_{2,3}$ ...
$R_{3,0}$ $R_{3,1}$ $R_{3,2}$ $R_{3,3}$ ...
$R_{0,0}$ $R_{0,1}$ $R_{0,2}$ $R_{0,3}$ ...
$R_{1,0}$ $R_{1,1}$ $R_{1,2}$ $R_{1,3}$ ...
$R_{2,0}$ $R_{2,1}$ $R_{2,2}$ $R_{2,3}$ ...
$R_{3,0}$ $R_{3,1}$ $R_{3,2}$ $R_{3,3}$ ...

Figure 3.12. Different Search Techniques to Identify the Next R Value

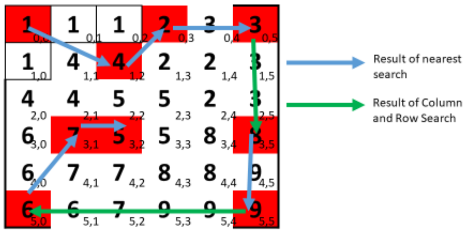


Figure 3.13. Navigational path generated on the Random Tile set

3.4.3 Navigation Strategy

Given a sequenced robot $WP[]_{nav}$, we have developed a navigation algorithm that generates the connection between way- points and create a fully connected navigation route for robot navigation within the entire navigation process. The navigation algorithm is described in detail in Algorithm 5. In the case of our proposed tiling motion planning and sequencing, waypoint array $WP[]_{tm}$ will be used; while during the simulation phase, several other sequencing algorithms and waypoint arrays will be taken as input in Algorithm 5. In the developed navigation strategy, we refer to the connection between every waypoint pairs as "paths"; while we call the connection between all paths "route". In Algorithm 5, we generate paths using function pathGen, and we connect them in specific sequences utilizing function routeGen, which eventually generates an entire route in the workspace for the hTetro to follow once the navigation begins.

In function pathGen, a virtual robot located at the position of ($rbtX$, $rbtY$) with velocity vel is created to generate the path between two waypoints. The robot initializes its position at the initial waypoint (Q_i) and updates its morphology according to the starting configuration as well. The algorithm first checks whether any morphology reconfiguration is required by com- paring its current morphology to the morphology at the final waypoint (Q_f). If a different configuration is detected, the robot will prioritize the robot transformation compared to the linear movements. The platform will check whether the surrounding environment is providing enough space by executing function canTransform check. If no nearby obstacles are presented and the robot transformation command is valid, the transformation command, which is represented as ’transform(θ_1, θ_2, θ_3, θ_4)’ in the algorithm, will be pushed to $Path[]$. If the hTetro transformation command is not valid at the current virtual robot position, it will proceed to the second movement command, and execute the transformation check again once it reaches a new position in the workspace. The movement command is the crucial portion of the pathGen function, which will move the virtual robot in a small distance towards the next waypoint either in X or Y direction, whichever yields a larger distance difference between the two waypoints, until the robot arrives at the positon of the final waypoint. The moved distance is determined based on the robot velocity vel. Once the robot path segment is selected, it will be pushed to the $Path[]$ as well, issuing a movement command, represented as ’move(x, y)’ to the robot once the navigation be- gins. The process will repeat until the virtual robot arrives at the position of the final waypoint with the corresponding robot morphology, and the generated path will be utilized to construct an entire robot route.

Finally, function routeGen will execute a loop which search for the correct sequence that connects the generated paths from the pathGen function and push them into a route array ($route[]$) as output. During robot navigation, the robot will continuously follow the commands in $route[]$ and will eventually cover the entire workspace area with the desired waypoint sequences.

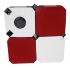
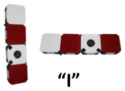
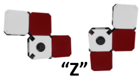
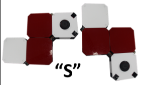
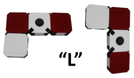
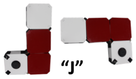
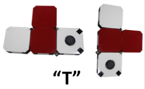
Tile Type	Orientation	T_v	Reference element	Q [X, Y, $\theta B^1, \theta B^2, \theta B^3, \theta B^4, \theta B^G$]
 “O”	0	-7	$M^t_{i_4j_4}$	[i^R_4,j^R_4, 0, 0, π, 0, 0]
	90		$M^t_{i_3j_3}$	[i^R_3,j^R_3, 0, 0, π, 0, 90]
	180		$M^t_{i_1j_1}$	[i^R_1,j^R_1, 0, 0, π, 0, 180]
	270		$M^t_{i_2j_2}$	[i^R_2,j^R_2, 0, 0, π, 0, 270]
 “I”	0	-6	$M^t_{i_2j_2}$	[i^R_2,j^R_2, 0, 0, 0, 0, 0]
	180		$M^t_{i_3j_3}$	[i^R_3,j^R_3, 0, 0, 0, 0, 180]
	90	-15	$M^t_{i_3j_3}$	[i^R_3,j^R_3, 0, 0, 0, 0, 90]
	270		$M^t_{i_2j_2}$	[i^R_2,j^R_2, 0, 0, 0, 0, 270]
 “Z”	0	-1	$M^t_{i_3j_3}$	[i^R_3,j^R_3,π, 0, 0, π,0]
	180		$M^t_{i_2j_2}$	[i^R_2,j^R_2,π, 0, 0, π,180]
	90	-17	$M^t_{i_3j_3}$	[i^R_3,j^R_3,π, 0, 0, π,90]
	270		$M^t_{i_2j_2}$	[i^R_2,j^R_2,π, 0, 0, π,270]
 “S”	0	-11	$M^t_{i_1j_1}$	[i^R_1,j^R_1, π, 0, $\frac{\pi}{2}$, 0,0]
	180		$M^t_{i_4j_4}$	[i^R_4,j^R_4, π, 0, $\frac{\pi}{2}$, 0,180]
	90	-2	$M^t_{i_2j_2}$	[i^R_2,j^R_2, π, 0, $\frac{\pi}{2}$, 0,90]
	270		$M^t_{i_3j_3}$	[i^R_3,j^R_3, π, 0, $\frac{\pi}{2}$, 0,270]
 “L”	0	-9	$M^t_{i_2j_2}$	[i^R_2,j^R_2,0, 0, 0, π,0]
	90	0	$M^t_{i_2j_2}$	[i^R_2,j^R_2,0, 0, 0, π,90]
	180	-10	$M^t_{i_3j_3}$	[i^R_3,j^R_3,0, 0, 0, π,180]
	270	-4	$M^t_{i_3j_3}$	[i^R_3,j^R_3,0, 0, 0, π,270]
 “J”	0	-5	$M^t_{i_2j_2}$	[i^R_2,j^R_2, π, 0, $\frac{\pi}{2}$, π,0]
	90	-16	$M^t_{i_4j_4}$	[i^R_4,j^R_4, π, 0, $\frac{\pi}{2}$, π,90]
	180	-14	$M^t_{i_4j_4}$	[i^R_4,j^R_4, π, 0, $\frac{\pi}{2}$, π,180]
	270	-13	$M^t_{i_1j_1}$	[i^R_1,j^R_1, π, 0, $\frac{\pi}{2}$, π,270]
 “T”	0	-8	$M^t_{i_4j_4}$	[i^R_4,j^R_4, $\frac{\pi}{2}$, 0, π,π,0]
	90	-12	$M^t_{i_4j_4}$	[i^R_4,j^R_4, $\frac{\pi}{2}$, 0, π,π,90]
	180	-3	$M^t_{i_1j_1}$	[i^R_1,j^R_1, $\frac{\pi}{2}$, 0, π,π,180]
	270	-6	$M^t_{i_1j_1}$	[i^R_1,j^R_1, $\frac{\pi}{2}$, 0, π,π,270]

Figure 3.14. hTetro robot’s T_v value and corresponding Q values

Algorithm 4 Tiling Motion Sequencing of Generated Waypoints

```
function $\textsc{TilingMotionSeq}(WP[], M')$
    $R \leftarrow 1, x_f \leftarrow 0, WP_{tm} \leftarrow [], dist \leftarrow 1$
    $Step_{nearby} \leftarrow \{(-1,-1),(-1,0),(-1,1),(0,-1),(0,1),$
    $(1,1),(1,0),(1,-1)\}$
    $Step_{rc} \leftarrow \{(-1,0),(0,-1),(1,0),(0,1)$
    while $\textsc{length}(WP_{tm}[])! = \textsc{length}(WP[])$ do
        for all $i \leftarrow 0$ to $\textsc{length}(WP)$ do
            if $\textsc{length}(WP_{tm}[]) + 1 = R$ then
                $x \leftarrow WP[i][1]$ ; $y \leftarrow WP[i][2]$
                $G[] \leftarrow []$ ; $minRValue \leftarrow \infty$
                // Perform nearby grid search.
                for all $(x_{step}, y_{step}) \in Step_{nearby}$ do
                    $x_{next} \leftarrow x + x_{step}$ ; $y_{next} \leftarrow y + y_{step}$
                    $G[] \leftarrow \textsc{push}G(x', y', n_{row}, n_{col}, G[])$
                end for
                while $\textsc{is}E\textsc{mpty}(G[])$ do
                    // Perform row and column search
                    $dist \leftarrow dist + 1$
                    for all $(x_{step}, y_{step}) \in Step_{rc}$ do
                        $x' \leftarrow x + x_{step} \times dist$
                        $y' \leftarrow y + y_{step} \times dist$
                        $G[] \leftarrow \textsc{push}G(x', y', n_{row}, n_{col}, G[], WP_{tm}[])$
                    end for
                end while
                for all $(x, y) \in G[]$ do
                    $RValue \leftarrow \textsc{get}RV\textsc{alue}(x, y)$
                    if $RValue < minR$ then
                        $minRValue \leftarrow RValue$
                        $Q \leftarrow \textsc{get}R\textsc{ef}C\textsc{oord}(x, y)$
                    end if
                end for
                $WP_{tm}[].\textsc{push}(Q)$
            end if
        end for
    end while
    return $WP_{tm}[]$
end function

function $\textsc{push}G(x, y, n_{row}, n_{col}, G[], WP_{tm}[])$
    if $x > n_{row} \lor x <= 0 \lor y > n_{col} \lor y <= 0$ then
        $Q \leftarrow \textsc{get}R\textsc{ef}C\textsc{oord}(x, y)$
        if $Q \notin WP_{tm}[]$ then
            $G[].\textsc{push}(x, y)$
        end if
    end if
    return $G[]$
end function
```

3.5 Simulated Experiments

As part of this study, we have evaluated the proposed motion planning on hTetro robot in a simulated environment. In order to analyze the performance of the proposed motion planning algorithm, we will compare the results with several algorithms that tackle the waypoint sequencing as well. In the simulation, the simulator adopts all the algorithms to guide the robot to traverse within the workspace and to clear the waypoints. The simulations are conducted using MATLAB Simulink software. This section will firstly introduce the six path planning strategies that we have selected as our benchmarking algorithms.

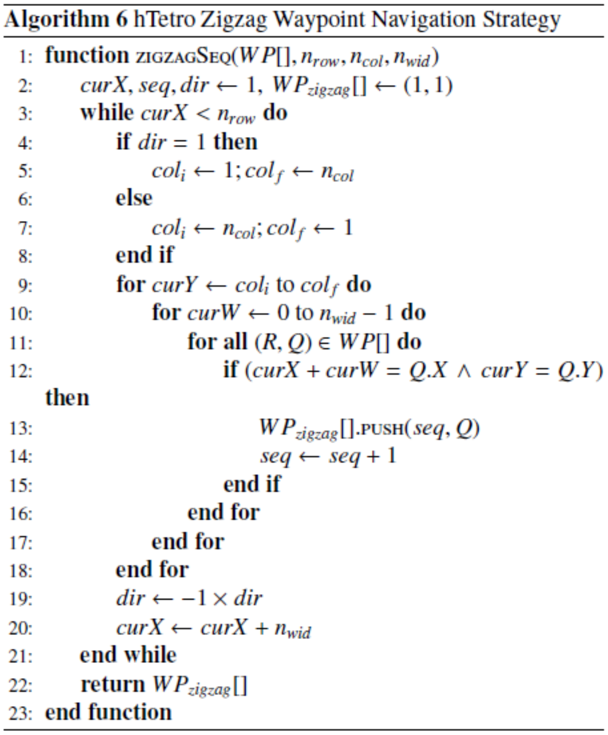
The evaluation criteria will then be introduced. Finally, the workspace setups under three scenarios and the navigation results in these scenarios will be presented.

3.5.1 Benchmarking Path Planning Strategies

In order to accomplish CPP tasks, many algorithms have been proposed for robots with a single fixed morphology, each of which is constructed based on a specific cellular decomposition method. Among all path planning algorithms that adopted the grid-based cellular decomposition technique, zigzag path (or boustrophedon path) and spiral path are the two popular navigation strategies for the robot to successfully accomplish area coverage. The conventional zigzag path algorithm explores the area with a constant sweeping width, and performs an 180◦ turn when an obstacle is encountered and move on to the next sweeping lane so as to avoid the swept areas. While the conventional spiral path algorithm works under a similar framework, where the robot performs 90◦ turns in a fixed direction when an obstacle is presented or when the robot enters areas that have already been covered, resulting a spiral-shaped navigation path with diminishing width as time goes on. Both algorithms are capable of producing efficient results in terms of minimizing the amount of area revisited by the robot platform.

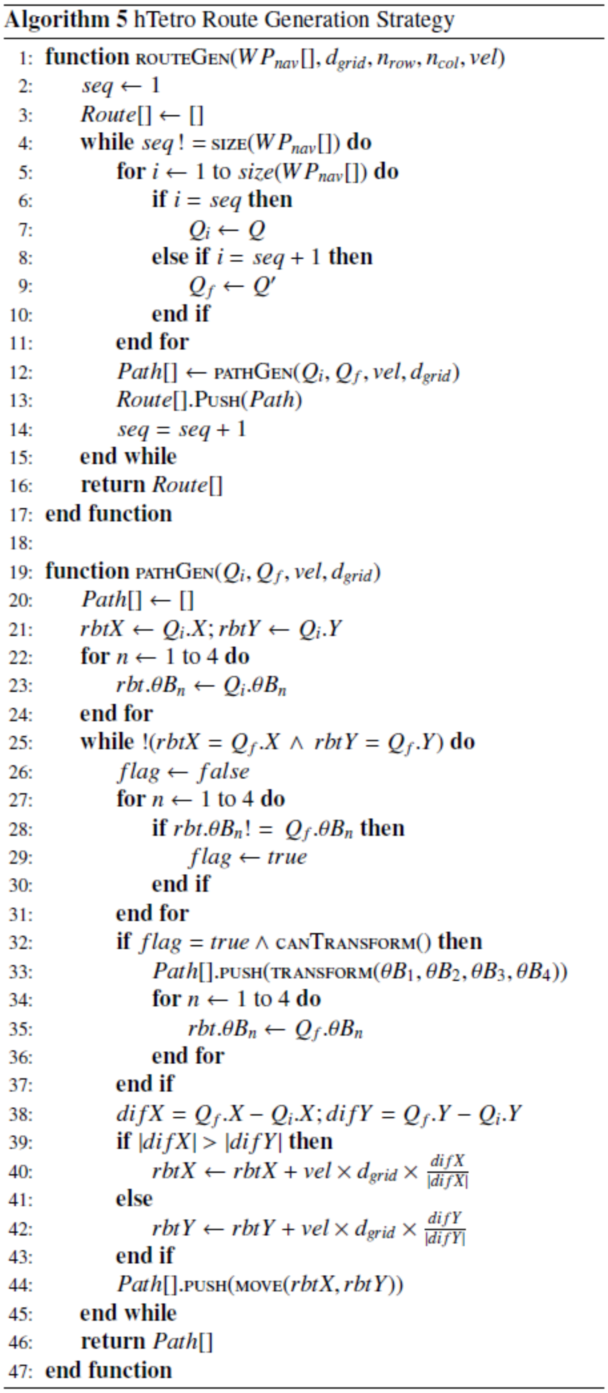
Nevertheless, unlike the proposed motion planning algorithm which makes use of waypoint sequencing to navigate within the workspace, the conventional zigzag and spiral algorithms do not guarantee complete coverage of the entire area, so minor adjustments to the existing conventional zigzag and spiral path algorithms have to be made for our hTetro robot simulation model. In order to make meaningful comparison between different navigation strategies, we consider accomplishing complete area coverage as a baseline criteria for a valid benchmarking algorithm. To achieve this goal, we make use of the unsequenced waypoint array *WP*[] that was created previously and produce different sequencing patterns for robot navigation. In this chapter, six sequencing patterns are being introduced to make comparison with the proposed tiling motion algorithm, which includes random path algorithm, greedy path algorithm, two zigzag-pattern based algorithms, and two spiral-pattern based algorithms.

Here we adopt greedy search [105] and random search algorithms [106] directly into our model without making any adjustments to the algorithm. Starting from the first robot configuration in the navigation process, the greedy search algorithm will continuously explore the workspace and assign the nearest configuration as the following waypoint; while the random search assigns the next waypoint from the unsequenced waypoint set at random. The zigzag-pattern and spiral-pattern based algorithm are being described in Algorithm 6 and Algorithm 7, respectively. These modified algorithms now take the unsequenced waypoint series *WP*[] and a sweeping width n_{wid} as input, and produce sequenced waypoint series (WP_{zigzag}[] or WP_{spiral}[]) as output. The modified algorithms will sweep the space based on conventional zigzag and spiral path algorithm. Nevertheless, the robot will no longer follow the swept path in the adjusted algorithms, instead, it will follow the sequence of the waypoints that are being swept by either zigzag or spiral algorithm. By making this minor adjustment, we can ensure full area coverage of the generated routes in zigzag-pattern and spiral-pattern based algorithms. Considering that traditional zigzag and spiral algorithms are sweeping the environment in single unit grid width, we introduce the sweeping width n_{wid} so as to better describe the scenario during the navigation process of hTetro robot. In all seven feasible hTetro morphologies, the average width that are being swept by hTetro robot is around 2 unit grid width, so we have selected $n_{wid} = 1$ and $n_{wid} = 2$ for both zigzag-pattern and spiral-pattern based navigation as benchmarking algorithms.



Algorithm 6 hTetro Zigzag Waypoint Navigation Strategy

```
function zigzagSeq($WP[], n_{row}, n_{col}, n_{wid}$)
    $curX, seq, dir \leftarrow 1,\ WP_{zigzag}[] \leftarrow (1,1)$
    while $curX < n_{row}$ do
        if $dir = 1$ then
            $col_i \leftarrow 1; col_f \leftarrow n_{col}$
        else
            $col_i \leftarrow n_{col}; col_f \leftarrow 1$
        end if
        for $curY \leftarrow col_i$ to $col_f$ do
            for $curW \leftarrow 0$ to $n_{wid} - 1$ do
                for all $(R, Q) \in WP[]$ do
                    if $(curX + curW = Q.X \wedge curY = Q.Y)$ then
                        $WP_{zigzag}[]$.push($seq, Q$)
                        $seq \leftarrow seq + 1$
                    end if
                end for
            end for
        end for
        $dir \leftarrow -1 \times dir$
        $curX \leftarrow curX + n_{wid}$
    end while
    return $WP_{zigzag}[]$
end function
```



Algorithm 5 hTetro Route Generation Strategy

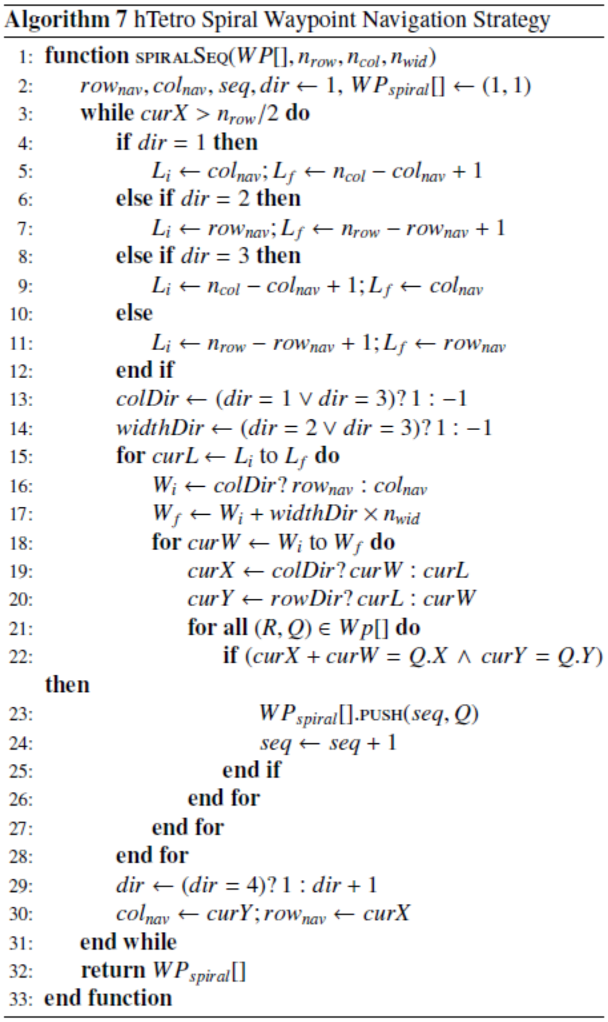
```
function ROUTEGEN($WP_{nav}[], d_{grid}, n_{row}, n_{col}, vel$)
    $seq \leftarrow 1$
    $Route[] \leftarrow []$
    while $seq\,! = $ SIZE($WP_{nav}[]$) do
        for $i \leftarrow 1$ to $size(WP_{nav}[])$ do
            if $i = seq$ then
                $Q_i \leftarrow Q$
            else if $i = seq + 1$ then
                $Q_f \leftarrow Q'$
            end if
        end for
        $Path[] \leftarrow$ PATHGEN($Q_i, Q_f, vel, d_{grid}$)
        $Route[]$.PUSH($Path$)
        $seq = seq + 1$
    end while
    return $Route[]$
end function

function PATHGEN($Q_i, Q_f, vel, d_{grid}$)
    $Path[] \leftarrow []$
    $rbtX \leftarrow Q_i.X; rbtY \leftarrow Q_i.Y$
    for $n \leftarrow 1$ to 4 do
        $rbt.\theta B_n \leftarrow Q_i.\theta B_n$
    end for
    while !($rbtX = Q_f.X \wedge rbtY = Q_f.Y$) do
        $flag \leftarrow false$
        for $n \leftarrow 1$ to 4 do
            if $rbt.\theta B_n! = Q_f.\theta B_n$ then
                $flag \leftarrow true$
            end if
        end for
        if $flag = true \wedge$ CANTRANSFORM() then
            $Path[]$.PUSH(TRANSFORM($\theta B_1, \theta B_2, \theta B_3, \theta B_4$))
            for $n \leftarrow 1$ to 4 do
                $rbt.\theta B_n \leftarrow Q_f.\theta B_n$
            end for
        end if
        $difX = Q_f.X - Q_i.X; difY = Q_f.Y - Q_i.Y$
        if $|difX| > |difY|$ then
            $rbtX \leftarrow rbtX + vel \times d_{grid} \times \frac{difX}{|difX|}$
        else
            $rbtY \leftarrow rbtY + vel \times d_{grid} \times \frac{difY}{|difY|}$
        end if
        $Path[]$.PUSH(MOVE($rbtX, rbtY$))
    end while
    return $Path[]$
end function
```

3.5.2 Evaluation Criteria

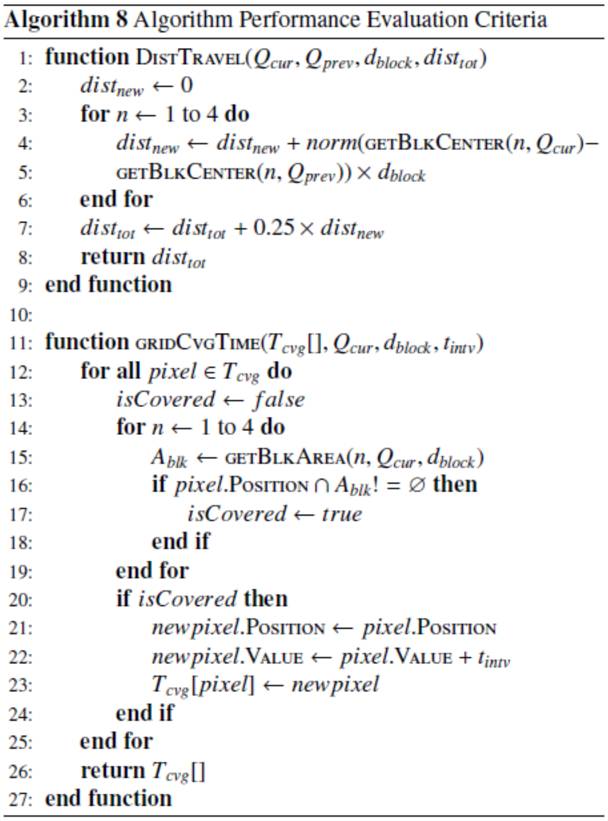
By constructing the proposed planning motion algorithm and six benchmarking navigation algorithms based on waypoint sequencing, the same coverage of the workspace across these algorithm is guaranteed, and further analysis on the performances can be conducted.

We propose two evaluation criteria to find the most efficient algorithm: average distance traveled and average grid cover- age time. Both criteria are presented in Algorithm 8. Average distance traveled is determined based on the trajectories of all four hTetro blocks during the navigation process. In the algorithm, the distance travelled by each hTetro block within every time instance will be calculated and stored in $dist_{tot}$ until the navigation terminates. To calculate the average grid cover- age time of a navigation strategy, we assume that each single grid in the workspace requires minimal 1 second to be covered by an hTetro block. We then decompose each grid in the workspace into 10 × 10 smaller square-shaped pieces referred to as 'pixels'. These pixels will provide a better resolution of total amount of time the robot spends within the workspace. The grid coverage time is stored in an array $T_{cvg}[]$ in Algorithm 8. Since hTetro robot is moving at a constant speed during the simulation, the software will calculate the time hTetro blocks used to cover each individual pixels by sampling the entire workspace every time interval t_{intv}. Once the navigation terminates, the average grid coverage time will be determined accordingly by calculating the mean value of $T_{cvg}[]$ within all accessible pixels in the workspace. The average grid coverage time is an important criterion which determines the efficiency of the proposed path planning algorithm in terms of area re- covered.



Algorithm 7 hTetro Spiral Waypoint Navigation Strategy

```
function SPIRALSEQ($WP[], n_{row}, n_{col}, n_{wid}$)
    $row_{nav}, col_{nav}, seq, dir \leftarrow 1,\ WP_{spiral}[] \leftarrow (1, 1)$
    while $curX > n_{row}/2$ do
        if $dir = 1$ then
            $L_i \leftarrow col_{nav}; L_f \leftarrow n_{col} - col_{nav} + 1$
        else if $dir = 2$ then
            $L_i \leftarrow row_{nav}; L_f \leftarrow n_{row} - row_{nav} + 1$
        else if $dir = 3$ then
            $L_i \leftarrow n_{col} - col_{nav} + 1; L_f \leftarrow col_{nav}$
        else
            $L_i \leftarrow n_{row} - row_{nav} + 1; L_f \leftarrow row_{nav}$
        end if
        $colDir \leftarrow (dir = 1 \vee dir = 3)?\, 1 : -1$
        $widthDir \leftarrow (dir = 2 \vee dir = 3)?\, 1 : -1$
        for $curL \leftarrow L_i$ to $L_f$ do
            $W_i \leftarrow colDir?\, row_{nav} : col_{nav}$
            $W_f \leftarrow W_i + widthDir \times n_{wid}$
            for $curW \leftarrow W_i$ to $W_f$ do
                $curX \leftarrow colDir?\, curW : curL$
                $curY \leftarrow rowDir?\, curL : curW$
                for all $(R, Q) \in Wp[]$ do
                    if $(curX + curW = Q.X \wedge curY = Q.Y)$ then
                        $WP_{spiral}[]$.PUSH($seq, Q$)
                        $seq \leftarrow seq + 1$
                    end if
                end for
            end for
        end for
        $dir \leftarrow (dir = 4)?\, 1 : dir + 1$
        $col_{nav} \leftarrow curY; row_{nav} \leftarrow curX$
    end while
    return $WP_{spiral}[]$
end function
```



Algorithm 8 Algorithm Performance Evaluation Criteria

```
function DistTravel($Q_{cur}, Q_{prev}, d_{block}, dist_{tot}$)
    $dist_{new} \leftarrow 0$
    for $n \leftarrow 1$ to 4 do
        $dist_{new} \leftarrow dist_{new} + norm($getBlkCenter$(n, Q_{cur}) -$
        getBlkCenter$(n, Q_{prev})) \times d_{block}$
    end for
    $dist_{tot} \leftarrow dist_{tot} + 0.25 \times dist_{new}$
    return $dist_{tot}$
end function

function gridCvgTime($T_{cvg}[], Q_{cur}, d_{block}, t_{intv}$)
    for all $pixel \in T_{cvg}$ do
        $isCovered \leftarrow false$
        for $n \leftarrow 1$ to 4 do
            $A_{blk} \leftarrow$ getBlkArea$(n, Q_{cur}, d_{block})$
            if $pixel$.Position $\cap A_{blk}! = \varnothing$ then
                $isCovered \leftarrow true$
            end if
        end for
        if $isCovered$ then
            $newpixel$.Position $\leftarrow pixel$.Position
            $newpixel$.Value $\leftarrow pixel$.Value $+ t_{intv}$
            $T_{cvg}[pixel] \leftarrow newpixel$
        end if
    end for
    return $T_{cvg}[]$
end function
```

3.5.3 Experiments With Proposed Technique:

In this experiment, a total of three scenarios are being tested to analyze the performance of the algorithms. The workspace of these scenarios are being illustrated in Fig 3.15. Scenario-1 illustrates an empty workspace of 10 x 10 grids. The grid width $d_{grid}$is set to 25 cm. Scenario-2 and Scenario-3 are workspaces with scattered obstacles. The workspace size are 11 x 11 and 11 x 14 grids, respectively; while the grid width remains the same. With the grid map created, the simulator runs the global tiling set algorithm as described in Section 4 to tile the grid space. Matrix M^t is being created for each scenario according to the algorithm. Elements in these matrices will hold a negative value of 1 if an obstacle is presented at the position. The fully tiled matrices M^t are being presented in Fig 3.16. It is observed that in Scenario-1, the combination of 'T', 'S', 'Z' tetrominoes are being used to complete the tiling task based on Theorem 4.1. Since the workspace in Scenario-2 does not fulfill the requirements in 4.1, a different combination of 'O', 'L','J' tetrominoes are being utilized instead according to Theorem 4.3; while Scenario-3 is being tiled based on Theorem 4.2. The completion of the tile matrices leads the simulator to generate the reference points for each tile pieces as described in Section 5, with the results shown in Fig 3.17. Once the unsequenced waypoint arrays WP[] are generated, the proposed tiling motion and the benchmarking algorithm will be run to determine the unique sequences of the waypoints to be visited. We define the upper left robot morphology as the starting robot waypoint for all seven algorithms in each scenario to eliminate the influence caused by random starting positions. The result of the generation phase of the proposed tiling motion algorithm is shown in Fig 3.18. The exact waypoint navigation sequence generated by the other six sequences are not presented in this chapter, but the efficiency performances are still presented and analyzed in the following section.

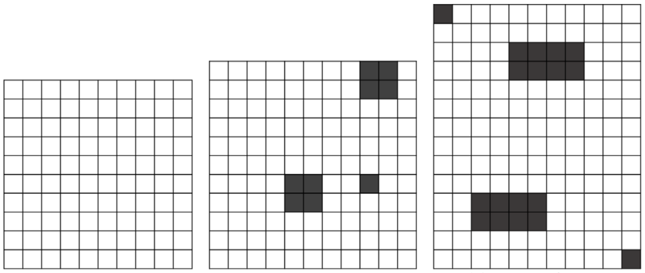
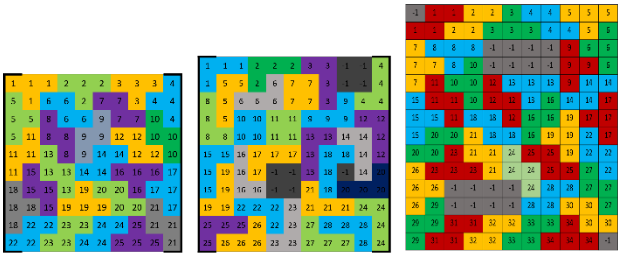

(a) Scenario - 1 (b) Scenario - 2 (c) Scenario - 3

Figure 3.15: Workspace map of the three evaluation scenarios



1	1	1	2	2	2	3	3	3	4
5	1	6	6	2	7	7	3	4	4
5	5	8	6	6	9	7	7	10	4
5	11	8	8	9	9	12	12	10	10
11	11	13	8	9	14	14	12	12	10
11	15	13	13	14	14	16	16	16	17
18	15	15	13	19	20	20	16	17	17
18	18	15	19	19	19	20	20	21	17
18	22	22	23	23	24	24	25	21	21
22	22	23	23	24	24	25	25	25	21

(a) Scenario - 1

1	1	1	2	2	2	3	3	-1	-1	4
1	5	5	2	6	7	7	3	-1	-1	4
8	5	6	6	6	7	7	3	9	4	4
8	5	10	10	11	11	9	9	9	12	12
8	8	10	10	11	11	13	13	14	14	12
15	15	16	17	17	17	13	18	18	14	12
15	19	16	17	-1	-1	13	18	-1	14	20
15	19	16	16	-1	-1	21	18	20	20	20
19	19	22	22	22	23	21	21	21	24	24
25	25	25	26	22	23	27	28	28	28	24
25	26	26	26	23	23	27	27	27	28	24

(b) Scenario - 2

-1	1	1	2	2	3	4	4	5	5	5
1	1	2	2	3	3	3	4	4	5	6
7	8	8	8	-1	-1	-1	-1	9	6	6
7	7	8	10	-1	-1	-1	-1	9	9	6
7	11	10	10	12	13	13	13	9	14	14
15	11	11	10	12	12	13	16	14	14	17
15	15	11	18	18	12	16	16	19	17	17
15	20	20	21	18	18	16	19	19	22	17
20	20	23	21	21	24	25	25	19	22	22
26	23	23	23	21	24	24	25	25	27	22
26	26	-1	-1	-1	-1	24	28	28	27	27
26	29	-1	-1	-1	-1	28	28	30	30	27
29	29	31	31	32	32	33	33	34	30	30
29	31	31	32	32	33	33	34	34	34	-1

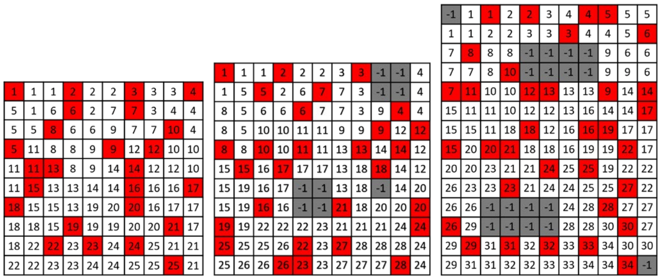
(c) Scenario - 3

Figure 3.16: Tiled matrices M^t in the three scenarios.

3.6 Results And Discussion

3.6.1 Results

During the navigation process, the average distance travelled and the grid coverage time is constantly calculated through Algorithm 8. The results for all three scenarios are being recorded in Table 3.1, Table 3. 2, and Table 3. 3. Based on the values calculated, bar charts are being plot- ted to compare the results between each algorithm, which is as shown in Fig 3.19 and Fig 3.20. In Fig 3.19, the average grid coverage time in all three scenarios are presented. All algorithms demonstrate a better performance in Scenario-1 since it is an obstacle-free environment and areas are less likely to be re-visited. With nrow = 2 chosen, the zigzag and spiral pattern navigation algorithms outperform the algorithms with nrow = 1. The performance of the presented tiling motion algorithm is highly competitive, with rather low average coverage time in all three scenarios compared to other algorithms. The average distance travelled outcome is illustrated in Fig 3.19. The distance travelled in the seven algorithms increases in Scenario- 2 and Scenario-3 compared to Scenario-1 due to an increase of the workspace size and the extra paths taken to avoid obstacles within the environment. The proposed tiling motion algorithm has demonstrated the shortest average distance travelled in all three scenarios compared to the rest of the algorithms. To have a better understanding of the navigation paths taken by each algorithm and how they pose influences in the algorithms performance, we create robot coverage heat maps to better visualize the data. A robot coverage heat map is created once the navigation process terminates and is constructed based on the final coverage values stored in $T_{avg}[]$. In the heat map, areas that are being covered by the hTetro robot are being represented in a color spectrum between yellow to red. The intensity of the red increases when more time are being spent for the robot to cover the area, indicating that the corresponding grid is being visited several times throughout the entire navigation process; while areas that are not being covered will remain black. In this chapter, the coverage heat maps of Scenario-1 and Scenario- 2 are being plotted, which is as shown in Fig 3.21, Fig 3.22 and in Fig 3.23.



1	1	1	2	2	2	3	3	3	4
5	1	6	6	2	7	7	3	4	4
5	5	8	6	6	9	7	7	10	4
5	11	8	8	9	9	12	12	10	10
11	11	13	8	9	14	14	12	12	10
11	15	13	13	14	14	16	16	16	17
18	15	15	13	19	20	20	16	17	17
18	18	15	19	19	19	20	20	21	17
18	22	22	23	23	24	24	25	21	21
22	22	23	23	24	24	25	25	25	21

(a) Scenario - 1

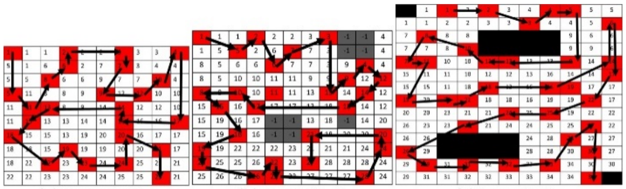
1	1	1	2	2	2	3	3	-1	-1	4
1	5	5	2	6	7	7	3	-1	-1	4
8	5	6	6	6	7	7	3	9	4	4
8	5	10	10	11	11	9	9	9	12	12
8	8	10	10	11	11	13	13	14	14	12
15	15	16	17	17	17	13	18	18	14	12
15	19	16	17	-1	-1	13	18	-1	14	20
15	19	16	16	-1	-1	21	18	20	20	20
19	19	22	22	22	23	21	21	21	24	24
25	25	25	26	22	23	27	28	28	28	24
25	26	26	26	23	23	27	27	27	28	24

(b) Scenario - 2

-1	1	1	2	2	3	4	4	5	5	5
1	1	2	2	3	3	3	4	4	5	6
7	8	8	8	-1	-1	-1	-1	9	6	6
7	7	8	10	-1	-1	-1	-1	9	9	6
7	11	10	10	12	13	13	13	9	14	14
15	11	11	10	12	12	13	16	14	14	17
15	15	11	18	18	12	16	16	19	17	17
15	20	20	21	18	18	16	19	19	22	17
20	20	23	21	21	24	25	25	19	22	22
26	23	23	23	21	24	24	25	25	27	22
26	26	-1	-1	-1	-1	24	28	28	27	27
26	29	-1	-1	-1	-1	28	28	30	30	27
29	29	31	31	32	32	33	33	34	30	30
29	31	31	32	32	33	33	34	34	34	-1

(c) Scenario - 3

Figure 3.17. Generated reference coordinates in the three scenarios.



(a) Scenario-1 (b) Scenario-2 (c)Scenario-3

Figure 3.18. Produced navigation paths in the three scenarios.

Table 3.2. Path Planning Performance Table for Scenario-1

Algorithm Name	Distance travelled	Average coverage time
Random Path	5246	8.942
Zigzag Pattern ($n_{r}ow = 1$)	2572	3.758
Spiral Pattern ($n_{r}ow = 1$)	2686	3.876
Zigzag Pattern ($n_{r}ow = 2$)	2064	3.140
Spiral Pattern ($n_{r}ow = 2$)	1886	2.643
Greedy Path	1770	2.609
Tiling Motion	1566	2.671

Table 3.3 Path Planning Performance Table for Scenario-2

Algorithm Name	Distance travelled	Average coverage time
Random Path	5588	9.705
Zigzag Pattern ($n_{r}ow = 1$)	3124	5.568
Spiral Pattern ($n_{r}ow = 1$)	3050	5.349
Zigzag Pattern ($n_{r}ow = 2$)	2320	4.085
Spiral Pattern ($n_{r}ow = 2$)	1994	3.303
Greedy Path	1964	3.380
Tiling Motion	1912	3.338

Table 3.4. Path Planning Performance Table for Scenario-3

Algorithm Name	Distance travelled	Average coverage time
Random Path	7146	10.404
Zigzag Pattern ($n_{r}ow = 1$)	2688	3.853
Spiral Pattern ($n_{r}ow = 1$)	3434	5.231
Zigzag Pattern ($n_{r}ow = 2$)	2236	3.223
Spiral Pattern ($n_{r}ow = 2$)	2290	3.330

Greedy Path	2088	2.939
Tiling Motion	2040	2.908

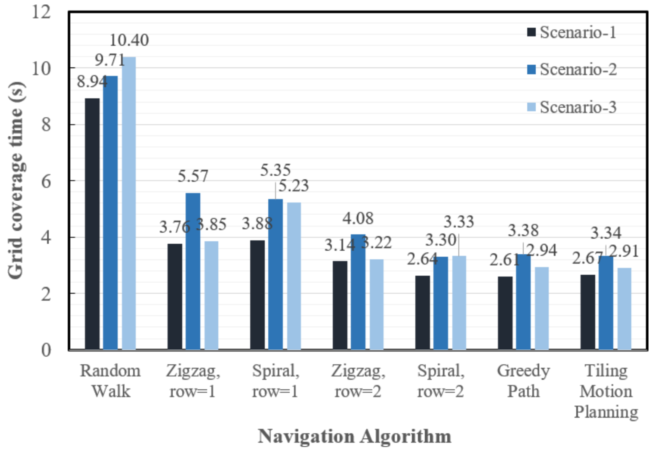


Figure 3.19. The grid coverage time comparison between the 7 algorithms under Scenario 1 to 3

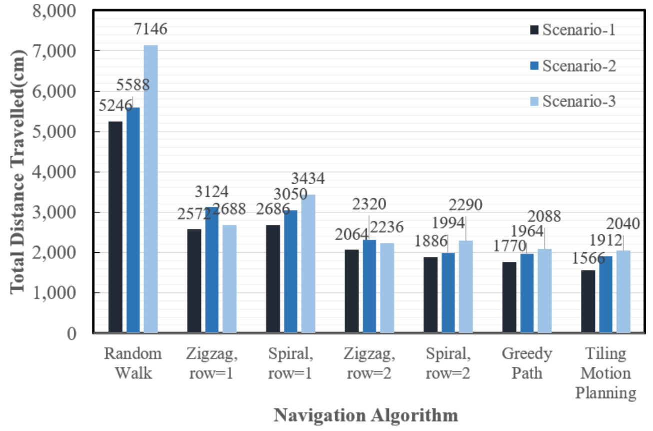


Figure 3.20. The average distance travelled comparison between the 7 algorithms under Scenario 1 to 3

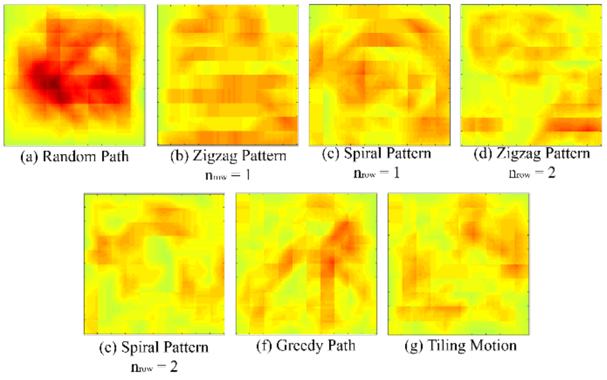
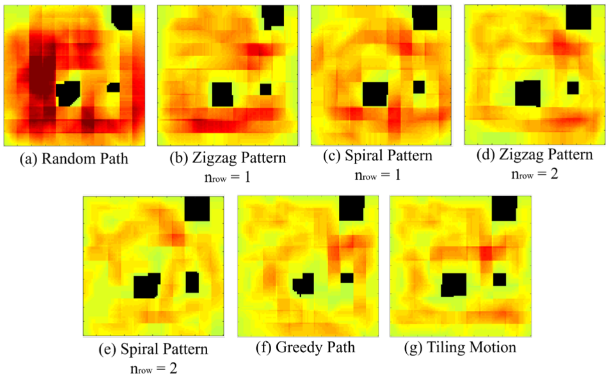
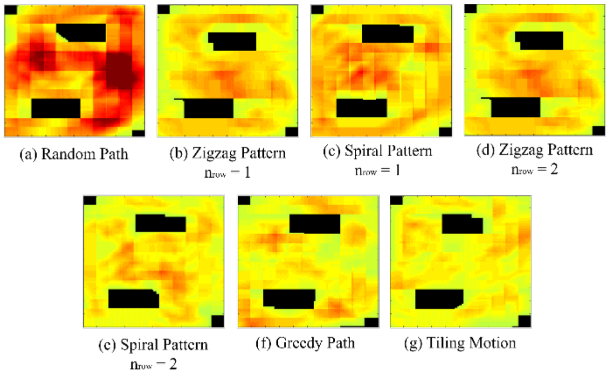

(a) Random Path
(b) Zigzag Pattern $n_{row} = 1$
(c) Spiral Pattern $n_{row} = 1$
(d) Zigzag Pattern $n_{row} = 2$
(e) Spiral Pattern $n_{row} = 2$
(f) Greedy Path
(g) Tiling Motion

Figure 3.21. Coverage heat map in Scenario-1.



(a) Random Path
(b) Zigzag Pattern n_{row} = 1
(c) Spiral Pattern n_{row} = 1
(d) Zigzag Pattern n_{row} = 2
(e) Spiral Pattern n_{row} = 2
(f) Greedy Path
(g) Tiling Motion

Figure 3.22. Coverage heat map in Scenario-2.



(a) Random Path
(b) Zigzag Pattern n_{row} − 1
(c) Spiral Pattern n_{row} = 1
(d) Zigzag Pattern n_{row} = 2
(c) Spiral Pattern n_{row} − 2
(f) Greedy Path
(g) Tiling Motion

Figure 3.23. Coverage heat map in Scenario-3.

3.6.1 Discussion

In this section, we discuss the advantage of our proposed motion planning technique for the hTetro robot by comparing with other path planning algorithms. In both zigzag and spiral pattern algorithms, the robot will follow the designated path to navigate within the environment. The advantage of these algorithms is similar to the conventional zigzag and spiral navigation, which provides a straight- forward method to perform the coverage task within the designated space. These algorithms tend to perform consistent results in larger maps since the performance is less likely to be affected by one or a few unsequenced waypoints within the workspace. The introduction of the revised zigzag and spiral algorithms (based on Algorithm 6 and Algorithm 7) ensures that the entire area coverage can be achieved. Fig 3.21(b-e), Fig 3.22(b- e), and Fig 3.23(b-e) all demonstrate that by increasing the navigation width from n_{row} = 1 to n_{row} = 2, the red-colored areas will significantly decrease. This implies that the navigation efficiency has been improved since fewer grids are being re-visited considering that during the navigation process of hTetro platform the robot tends to sweep two grids at a time instead of just a single grid.

The random path algorithm, though being able to success- fully cover the entire workspace, has shown to be an overall ineffective navigation strategy since the robot is constantly revisitng several grids in the workspace due to the randomly chosen waypoint sequence, which results in the highest average coverage time and the longest distance travelled among all seven algorithms in all scenarios. The greedy search algorithm generally performs well with a small sample size of waypoint sequences. Nevertheless, the greedy search algorithm gets punished by leaving several of the suboptimal waypoints behind during the navigation process. If a few waypoints are being ignored during the sequencing process, the robot has to travel a much longer distance to cover these waypoints at the end of the navigation process. The greedy algorithm also faces some issues when obstacles are being introduced in the map. In Fig 3.23(f), the bottom left corner has been re-visited multiple times compared to the results in Fig 3.23(e and g). This is due to an unwise selection of the navigation route in greedy algorithm that guides the robot to navigate from one side of the bottom-left obstacle in Scenario-3 to the other side (since it is the closest waypoint that can be dis- covered), resulting in undesirable average grid coverage time around obstacles. On the other hand, the proposed tiling motion planning algorithm has demonstrated a great performance with shortest distance travelled and decent time grid coverage time compared to the other algorithms in the three scenarios presented as shown in both Fig 3.19 and Fig 3.20. The grid coverage heat maps shows that the proposed algorithm might face some difficulties navigating through narrow spaces (as shown in the red-colored area in Fig 3.23(g)), but performs well in workspaces with large and empty areas as illustrated in Fig 3.21(g) and in Fig 3.23(g). Overall, based on the reliable and consistent outcome of the proposed tiling motion algorithm in this simulation, we are able to confidently conclude that this is currently the ideal algorithm that we can implement into physical robots for integration and move on to real-world experiments.

3.7 Conclusion

This chapter introduced a motion planning technique to achieve maximum area coverage in a Tetris-inspired shape shifting robotic floor cleaner named hTetro that was based on the polyomino tiling theory. We discussed on autonomous Tile set generation based on the tiling theory concept and introduced the path generation for the hTtero robot to navigate on the generated tiling set with an objective of maximizing the area coverage. We validated the proposed algorithm by benchmarking its performance with traditional coverage path planning techniques. In each experiment, the robot cleared the sequenced waypoints while assuming its morphology at each point. At the end of the performed experiments, the waypoint navigation map and the coverage heat map are being generated. The experimental results establish that the proposed tiling motion technique achieves maximum coverage area with less distance traveled and minimum re-covered area. Future research work is set to focus on the following areas: (1) Optimizing the reference point generation for the created tiling set with respect to energy cost (2) Optimizing the tiling set generated where less number of reconfiguration is required (3) Exploring other optimal techniques for path generation to cover the entire area (4) Studying locomotion control that generates smooth trajectories in realizing the tiling set. (5) Implementing the proposed approach for other polyforms reconfigurable robots.

4. The implementation of proposed motion planning technique on Tetris inspired reconfigurable robot

A wide range of literature has described the application of tiling theorem in the field of gaming applications. For instance, Jho and Lee developed a novel polymorph re-tiling scheme in [100] through which a set of polyomino pieces were alternated with a different combination of polyominoes pieces. The proposed algorithm was used in gaming puzzles that involved numerous models of polyominoes. Through this algorithm, it was possible to create new stages without utilizing more memory. Other examples include the work of [101], which consisted of a novel three dimensional tiling approach for use in a 3 Dimensional gaming puzzles, and the photominoes synthesizer described here, that used digital pictures to construct polyomino puzzle samples for a jigsaw puzzle. Even though numerous studies have focused on the application and development of polyomino tiling theorems, research of this nature is typically limited to the graphics and gaming fields. Furthermore, none of the existing research studies on tiling theorem have specifically focused on how this theory can be enforced to a robotic platforms to solve the floor coverage dispute. As such, there is an enormous space for research and development in this domain.
This chapter describes the utilization of the polyomino tiling theorem for a Tetris-Mimicked self-reconfigurable robotic floor cleaner that demonstrates the functionality to overcome the known area coverage problem. The robot platform can autonomously provoke the required tiling set for the defined area based on the polyominoes tiling theory. In this chapter, we reviewed the fundamental challenges of the proposed technique such as realizing the reconfigurable ability, smooth navigation, and translating the tiling theoretical approach through the analytical process into demonstrable systems. The chapter concludes by presenting an overview of the experimental results that were achieved using the developed hTetro platform that endorses the introduced technique. The application of the polyomino tiling theory to the floor coverage dispute associated with robotic floor cleaners represents a fundamental achievement in the process by which a reconfigurable system that can able to automatically provokes a global polyominoes tile set for any defined space is designed and developed.

Polyominoes are a simple geometrical design created by endwise coupling of equal squares [107]. Polyominoes can be classified into one of three categories—one-sided, fixed, and free polyominoes—based on their spatial orientation, geometrical transformation, and chirality. For instance, the standard domino, which is formed of two equilateral squares, can form single one-sided, one free and double-fixed dominoes as its subsets. Accordingly, triominoes (3-omino) can generate two free triominoes, double one-sided triominoess, and six fixed triominoes [107]. Tetrominoes contain four equivalent squares and can generate seven one-sided tetrominoes, nineteen fixed tetrominoes, and five free tetrominoes. The research described in this article employed a tetromino-mimicked self-reconfigurable system, hTetro that can able to transform between any of the one-sided tetrominoes morphologies.

4.1 Polyominoes Tiling Theory:
The polyominoes tiling theory is a branch of mathematics which conduct studies to tile a given 2d space with the set of polyominoes pieces. As previously described, numerous studies have examined the tiling theorems. The source of inspiration for the hTetro robot described in this chapter was the game of Tetris. This chapter describes our initial attempt to apply the tetromino based tiling theory to the area coverage problem of a reconfigurable robotic floor cleaner. Specifically, we evaluated three theorems that were previously proposed in [102] and [103] for tiling regular and modified rectangular region using the ‘T’, ‘L’, and ‘X’ tetrominoes. In our T tileset, shown in Fig 4.3. the tiles τ1 through τ4 are called skew tetrominoes and the tiles τ5 through τ8 are called T-tetrominoes. In the experiments that were performed as part of this study, we used a fixed testbed area that consisted of regular rectangle, or an altered rectangle with respect to the validating theorem such that hTetro could only achieve complete coverage using only sets of ‘T’, ‘L’, and ‘X’ configurations. The three theorems and lemmas that support the validity of the theorems are discussed in detail in the rest of this section.

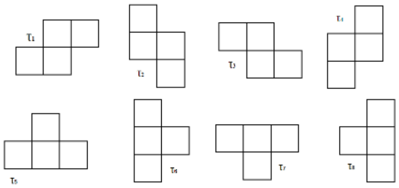


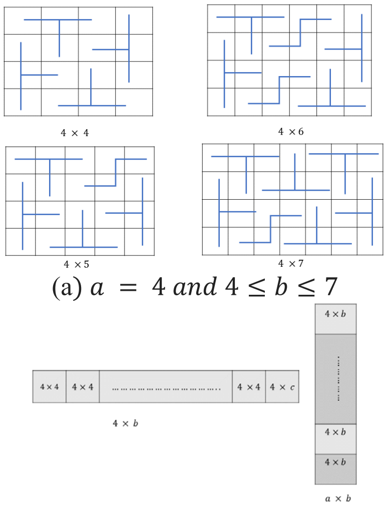
Figure 4.1. ‘T’ set of Tetromino pieces

Theorem 1: *An $a \times b$ rectangle can be tiled by the set T if, and only if, $a, b \geq 4$ and either*
1. One side is divisible by 4, or

2. $a, b \equiv 2\ (mod\ 4)\ And\ a + b > 16$.

'T' is a set of four skew tetrominoes and T tetrominoes. If a rectangular grid area is divisible by 4, then the grid space can be tiled with any of the tetromino pieces due to each Tetris shape is consist of four equivalent squares. This representation computes the first condition of Theorem 1. Assume a and b are the grid ranges of the rectangular space that need to be covered using tetris tile set. The above notation can be formulated to $(a \times b) = 4n$, where $n \in \{1,2,3 \ldots \ldots\}$ i.e. each of two a or b should be a multiple of 4. For instance, if the area is having a dimension of $4 \times 5, 8 \times 23, 12 \times 6 \ldots \ldots$ can be covered using 'T' tiling set. When it comes to the second condition in theorem 1, there are some rectangular space that cannot be covered with the 'T' set tetris pieces. For instance, the rectangle with the range of 6×6, 10×6, 6×10 cannot be tiled with any combination of 'T' set tetris Pieces.
The Lemma 1 and Lemma 2 described below endorse Theory 1.

Lemma 1: Let $a = 4$ and $4 \leq b \leq 7$, then a $4 \times b$ rectangle could be covered with the combination of skew and T Tetrominoes illustrated in Fig 4. 4(a). Let $b > 7$ and $c \in \{4, 5, 6, 7\}$, then $b = 4n + c$, where n can hold a positive integer value. This arrangement aids to decompose the $(4 \times b)$ space to $n\ (4 \times 4)$ r and single (4 × c) rectangular space, as illustrated in Fig 4. 4(b). The decomposition suggest if b ≥ 4, a rectangular space (4 × b) can be covered with the set of T and skew tetrominoes. Similarly, in the case of $a = 4m$, where m is a positive integer, then a rectangle (a × b) can be decomposed into m $(4 \times b)$ rectangles as shown in Fig 4. 4(b). Hence, a $(a \times b)$ rectangle can be tiled using T and skew tetrominoes.

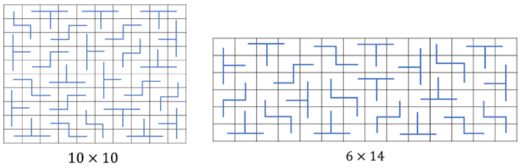


(a) $a = 4\ and\ 4 \leq b \leq 7$

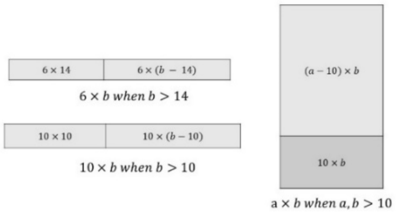
(b) The division of a x b rectangles into smaller rectangles

Figure 4.2. Image argument for lemma 1

Lemma 2: The smallest rectangles that satisfy Condition 2 of Theorem 1 are (10×10) and (6×14). These rectangles can be tilled using the arrangement shown in Fig 4. 5(a). In the case of b > 14, it is possible to decompose any (6 × b) rectangle into a (6 × 14) and a 6 × (b−14) rectangle, as shown in Fig 4. 5(b). By Lemma 1, since (b – 14) is divisible by 4 (as b and 14 are both congruent to 2 modulo 4), then the 6 × (b−14) and hence a 6 × b rectangle can be tiled with skew and T tetrominoes. Similarly, if b>10, then a (10 × b) can be divided into two smaller rectangles that can be tiled with the T set. If a, b >10, then an (a × b) can also be decimated into smaller rectangles (as shown in Fig 4. 5[b]) that are tileable based on Lemma 1. Hence, when $a \leq b$, then $(a \times b)$ rectangle can be tiled using skew and 'T' tetrominoes.



(a) 10x10 and 16x14 tiled rectangle



(b) Division of (a x b) rectangle

Figure 4.3. Image argument for lemma 2

Theorem 2: *Let a and b be integers that are strictly greater than 1. As such, the set 'L' of tetrominoes can tile a modified rectangle of side a and b, if, and only if, either:*
1. a≡2 (mod4)and b is odd, or
2. b=2 and a≡1 (mod 4).

Fig 4. 6, shows the set L, which is formed of a pair of T tetrominoes and skew tetrominoes. The term *modified rectangle* in the theorem describes a rectangle of sides a and b with a single square truncated from both corners. As per Theorem 1, the area of the modified rectangle should be divisible by four for the area to be tiled entirely using tetrominoes. Hence, it is clear that $(ab - 2)$ must be divisible by four. The above notion regarding tiling the modified rectangle points to Condition 1 and Condition 2 of Theorem 2. Lemma 3 and 4 validate Theorem 2 by proving Conditions 1 and 2.

Lemma 3: Let M (a, b) be the modified rectangle, and the smallest modified box is the first member of set L itself. If Condition 1 is valid for M(2, b-2) and b > 3, then M(2,b) can be decomposed into M(2,b-2) and M(2,3), as shown in Fig 4. 7(a). Hence, M (2, b) can be tiled using the Tetrominoes in set L when every odd value of b is greater than 1. If Condition 1 is valid for M (a-4, b) and a > 2, M (a, b) can be decomposed to M (a-4, b), S (b), and one 'T' tetromino, as depicted in Fig 4. 7(b). This proves that, if b > 1 is odd, then S (b) can be tiled using 'L'. If b = 3, we can see that L tiles S (3) as in Fig 4. 7(c). Moreover, if Condition 1 is valid for S (b-2) and b > 3, then S (b) can be decomposed to S (b-2), Tetromino τ1, and Tetromino τ3.

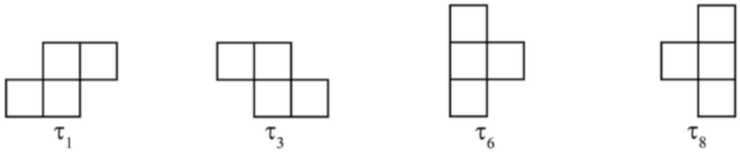
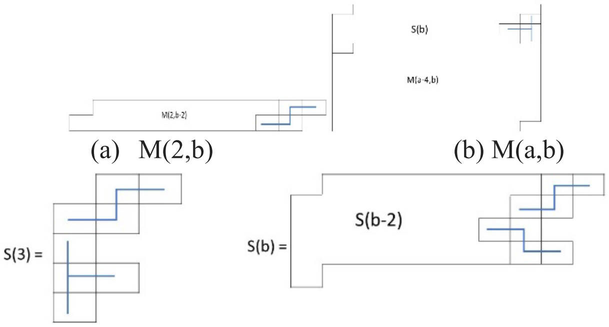


Figure 4.4. The set 'L' of tetrominoes.

Lemma 4: Lemma 4 supports Condition 2 of Theorem 2. The smallest such modified rectangle that comes under condition 2 is $M(5,2)$, can be tiled using τ_6 and τ_8, as in Fig 4. 8 (left). If Condition 2 is valid for $M(a-4,2)$ for a > 5, then $M(a,2)$ can be divided into two sections $M(a-4,2)$ and $M(5,2)$ and tileable, as shown in Fig 4. 8 (right).



(a) M(2,b)

(b) M(a,b)

(c) S(b)

Figure 4.5. Image argument for lemma 3

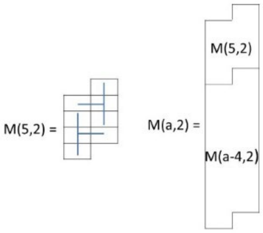


Figure 4.6. Image argument for lemma 4

Theorem 3: *the set 'X' of tetrominoes can tile a deficient rectangle of side a and b, only if*
1. The a and b is odd (2n+1)X(2n+1), n≥1, and
2. The missing cell has to be on the odd position if the rectangle is (4m + 1) × (4m + 1) m ≥ 1and in an even position if the rectangle is (4m + 3) × (4m + 3) m ≥ 1.

Fig 4. 9 shows a set of 'X' tetrominoes including different versions of L tetrominoes (τ9 to τ12) and an O tetromino (τ13) pieces. In this theorem, a square of side a and b with a single cell removed is defined as a deficient rectangle. We can observe that a pair of L-tetromino forms a 2*4 rectangle, and the combination of the 2*4 rectangle (a pair of L tetromonoes) and an O-tetromino can tile a 2*6 rectangle. Hence it is clear that aside with odd squares should eliminate a cell in order to tile rectangle with a set of 'X' tetromino. Below mentioned Lemma 5 and 6 supports the conditions of theorem3.

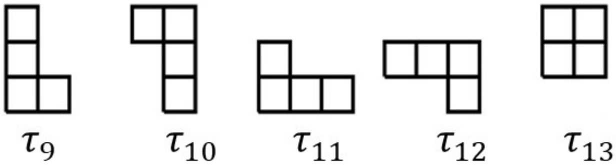
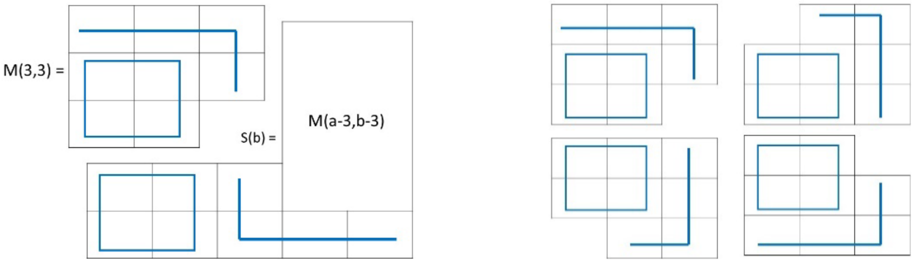


Figure 4.7. The set of 'X' Tetrominoes

Lemma 5: The smallest possible modified rectangle according to the condition 1 of Theorem 3 is M (3, 3). As per condition 1, M (a, b) can be decomposed into M (3, 3) and M (a-3, b-3), provided n> = 2. M (a-3, b-3) is mentioned as S (b) in Fig 4. 10(a). The 3*3 square can be tiled using an L-tetromino and an O-tetromino as shown in Fig 4. 10(b). If condition 1 is valid, then the rectangle will be having odd no of squares along its sides. If M (3, 3) is removed from the rectangle, even squares will remain untiled. This untiled space can be tiled using the set of 'X' tetrominoes.



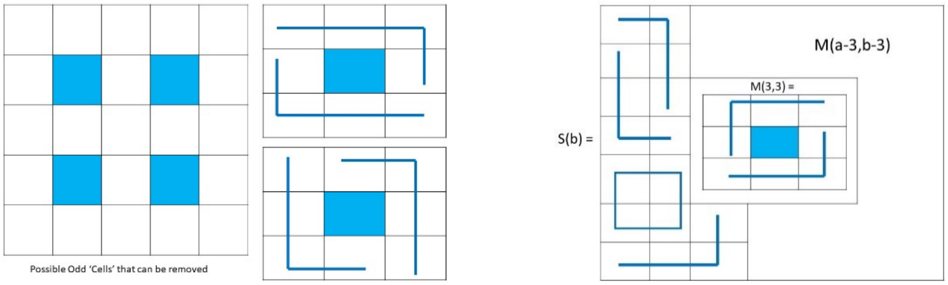
(a) M(a, b)

(b) Possible 3x3 squares tiling when odd squares removed with 'X' tetrominoes

Figure 4.8. Image argument for Lemma 5

Lemma 6: which is similar to Lemma 5 gives necessary support for validating the condition 2 of Theorem 3. As per the condition 2, an odd cell is removed when the sides are (4m+1)*(4m+1) and m >= 1. If an odd cell is removed, the rectangle can be tiled by decomposing into M (3*3) and S (b) as per lemma 5. In the case of rectangles having sides (4m+3) and (4m+3) m>=1, the even cells are removed as shown in Fig 4. 11(a left). In this case, we can decompose the rectangle into M (3, 3) and S(b) as in Lemma 5. The tiling strategy with M (3, 3) is illustrated in Fig 4. 11(a right).

Similar to the lemma 5, the remaining space of M (a,b) can be tiled with set of 'X' tetrominoes as shown in Fig 4. 11(b).



(a) Possible 3x3 squares tiling

(b) M (a, b)
When an even square
removed With 'X' Tetrominoes

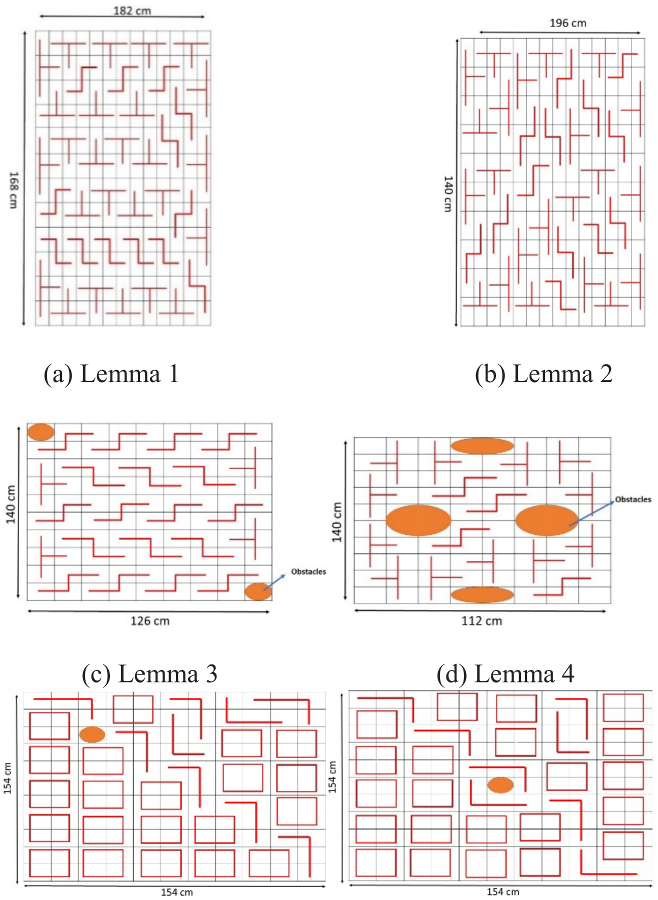
Figure 4.9. Image argument for Lemma 6

4.2 Experiments and Results

As part of this research, the polyomino tiling theory was applied to a novel Tetris-inspired reconfigurable floor-cleaning robot in order to solve the area coverage problem. Specifically, this experimental study aimed to validate three specific theorems and six lemmas, the details of which were presented in Section II. The first set of experiments tested Theorem 1, wherein the test area consisted of a rectangular plot of 168 cm x 182 cm which was segmented into 12 x 13 square grids, as per the conditions described in Lemma 1. Fig 4. 12(a) presents the global tiling set automatically generated based on Lemma 1 to tile the test area using only 'T' set of tetrominoes. To validate Lemma 2, a rectangular plot of 196 cm x 140 cm was employed as the test area, and this was segmented into 14 x 10 squares. Fig 4. 12(b). Presents the global tiling set based on Lemma 2 through which the second test area was tiled using only 'T' set of tetrominoes.

In the third set of experiments, we validated the application of Theorem 2 in our hTetro robot. We used a rectangular plot of dimensions 140 cm x 126 cm as the test area and segmented it into 10 x 9 square grids. Two obstacles with a size of the single block were placed at the opposite corner cells (1, 1) and (10, 9)of each segmented area. The modified area can be tilled using an 'L' set of tetromino pieces according to the argument supported by Lemma 3. Fig 4. 12(c) presents the global tiling set that was automatically generated based on Lemma 3 for the considered test area. The fourth set of experiments involved validation of Lemma 4. Since the grid size involved in Lemma 4 is small and highly constrained, we merged the smaller grids and developed a 10 x 8 square grid within a zone of 140 cm x 112 cm. To realize a modified rectangle that met the specifications of Lemma 4, obstacles were placed at the center of the testing field which covers the cell number (5, 2) (5, 3) & (6, 2) (6, 3). Also, two more obstacles were placed, one covers the cell number (1, 4) (1, 5), and another covers cells (10, 4) & (10, 5). Fig 4. 12(d). Presents the global tiling set based on Lemma 4. The objective was to tile the concerned test area using only 'L' set of tetrominoes.

Similarly, with Theorem 3, we used a square plot as a test arena with a dimension of 154 cm x 154 cm and segmented it into 11 x 11 square grids for both the Lemmas (5&6). We modified the defined area by placing obstacle inside the test arena. However, in the experiment conducted with Lemma 5, the obstacle was placed in an odd cell (3, 3) and was placed in an even cell (6, 6) when it comes to Lemma 6. In both the experiments the obstacles used are same that covers an area of 140 mm x 140 mm. Fig 4. 12(e & f) presents the global tiling set using 'X' tetrominoes that were automatically generated based on Lemma 5 and 6 arguments for the defined test area.



(a) Lemma 1

(b) Lemma 2

(c) Lemma 3

(d) Lemma 4

(e) Lemma 5

(f) Lemma 6

Figure 4.10. Tiled Testing Area with 'T', 'L', and 'X' tetrominoes.

4.2.1 Experimental Testbed

The experimental setup consisted of a predefined floor area that was segmented into square blocks that were congruent with individual blocks of the hTetro robot and overhead support frame mechanism to house an image-acquisition device. The raw image data from the vision sensor was post-processed to calculate the percentage area covered by the hTetro robot during the experimental trials. The total area defined for our experiments was 196 cm x 196 cm. The boundaries of the defined area were adjusted according to the arguments in each theorem using extendable metal frames. The defined area was segmented into a 14 x 14 square grid using white tape, and each square or cell size of the segmented grid is equivalent to the dimension of a single block in the hTtero robot (Fig 4. 13). A vibration-immune parallelepiped structure was constructed using an aluminum extrusion profile, and a camera was subsequently mounted on the midpoint of the upper surface of the constructed structure. Moreover, we ensured that the image plane lay horizontal to the ground in order to avoid the complexity of perspective projection in area calculation. Hence, when mounting the camera, we used a spirit level to ensure the camera was parallel to the floor. The camera arrangement and test area are depicted in Fig 4. 14. The auto-focus feature of the camera was disabled, and a fixed-focal-length was used to record the robot's tiling process. The resulting raw video files were post-processed to analyze the performance of the robot.

An image-processing algorithm was applied that employed three major steps to generate a map that tracked the movement of the hTetro robot. The first step in the process involved saving the reference image in which the track map would be generated. During the second step, the position and configuration of the robot was detected in every frame. The algorithm identified the hTetro robot as a triplet of blobs after multichannel color thresholding. The centroid of each of the blobs represents the centroid of each red color. The final step generated a track map by plotting the green-colored squares with reference to the detected blobs on the reference image. Once the track map had been generated, the percentage of the total area that had been covered was calculated using the following formula:

$$\%Coverage\ Area = \frac{\text{Pixels area of the robot}}{\text{Total pixels area of the testing field}}\ X\ 100 \tag{4.1}$$

$$\%\ Coverage\ Area = \frac{\text{Pixel area of the robot}}{\text{Total pixel area of the testing field} - \text{Total pixel area of the obstacles}}\ X100 \tag{4.2}$$

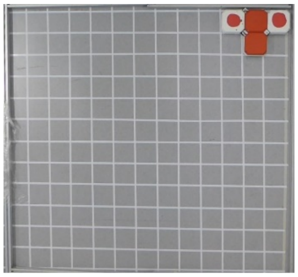


Figure 4.11. Defined test area with segmented square grids

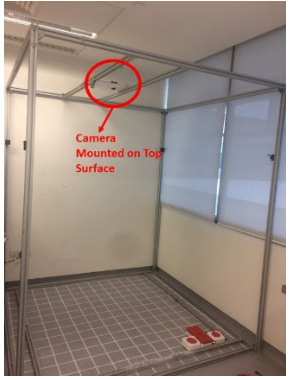


Figure 4.12. Test area with parallelepiped structure with provision for camera mount

4.2.2 Results & Analysis

Before each experimental run, the hTetro robot was placed in a pre-defined starting point within the test area. The area coverage process while conducting the experiment with the hTetro robot was recorded. After completing the experiments, the image-processing algorithm detailed in Section IV was used to process the recorded videos to produce the robot tracked map. Fig 4. 15 illustrates the robot tracked map with hTetro which was generated from the video recorded from the experiment validating first lemma. In the image, the green shaded pixels represents the area tiled by the hTetro. On top of the green shaded image the total area covered at that particular time point computed using (1) is displayed. Fig 4. 15 contains 4 set of images, 2 in each set that represents the coverage process at distinct time point of the experiment validated Lemma 1. In the each set of images, the odd pictures represents the hTetro robot's position at particular time point, and the even picture indicates hTetro tracked map with a overlaid tiling set at the same instance. The first set of images presents the tiling area while the area coverage was 30%, wherein the first image displays the hTetro robot while it was in Z-shape. Similarly the subsequent image sets displays 50, 60, and 97% 0f the area coverage while tiling T shaped Tetris pieces. The robot's locomotion and transformation were controlled using the hTetro user interface. The robot's trajectory was generated with respect to the produced global tiling set as illustrated in first Lemma. The results highlight that the hTetro robot could achieve a coverage area of more than 95% in the defined test space that validates the application of Lemma 1. In Experiment 2, which sought to validate Lemma 2, the hTetro robot covered an area more than 97%. In Fig 4. 16, the first set of images presents the tiling area while the area coverage was 34%, wherein the first image displays the hTetro robot while it was an S-shape. Similarly, the second and third set of images displays 57% and 69% of the area coverage which is also while tiling with S-shaped Tetris pieces. The final set of images offers area coverage of 97 when tiling in a T-shaped Tetris piece. The increase in area coverage performance in the second set of experiments was attributed to the excessive area covered during the transformation cycles of the hTetro robot due to uncertainty in transformation gait synthesis.

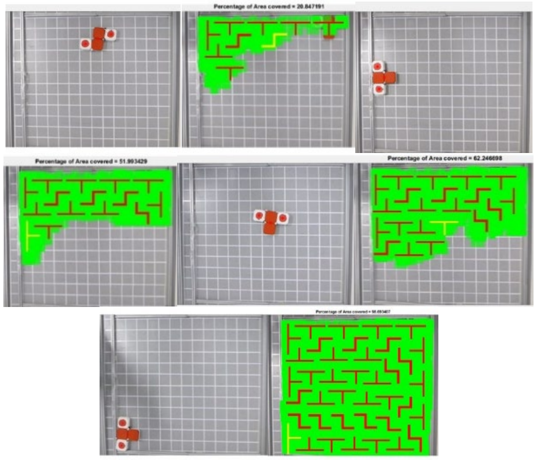

Percentage of Area covered = 20.847191
Percentage of Area covered = 51.993429
Percentage of Area covered = 62.246698

Figure 4.13. Experiments validating application of Lemma 1 in hTetro robot

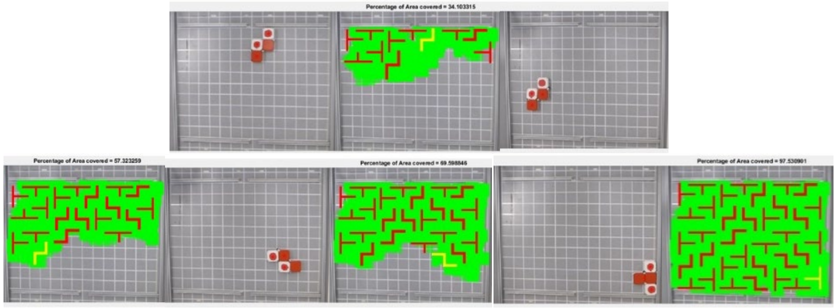

Percentage of Area covered = 34.103315
Percentage of Area covered = 57.323259
Percentage of Area covered = 69.598846
Percentage of Area covered = 97.530901

Figure 4.14. Experiments validating application of Lemma 2 in hTetro robot

Since Lemma 3 and 4 deals with modified test area, obstacles were placed within the experimenting field. The obstacles were placed two opposite corners of the testing field as shown in Fig 4 17. The test space dimensions were realigned with respect to the third and fourth lemma's argument. Since the area of the rectangles were modified, the total accessible area was calculated by neglecting the pixels correlated to the obstacles. Fig 4. 17 illustrates the tiling process of the hTetro robot that proofing the application of third Lemma. The first set of images shows a coverage area of 32% while tiling a T-shaped Tetris piece. The subsequent image sets displays a coverage area of 40, 65, and 99% while tiling the Z-, T-, and S-shaped Tetris pieces respectively. The total area covered was computed using (2). In the third set of experiments, the results clearly reveal that the hTetro system can cover more than 99 % of the total experimental space, by that validates the utilization of third lemma.

In the fourth experiment associating fourth Lemma, the hTetro robot achieves a coverage area of more than 95% of the defined experimental space. Fig 4. 18 illustrates the coverage area all along 17%, 43%, 75%, and 95% coverage process of our fourth set of experiments. The odd images of Fig 4 17 shows the different stage of the experiment 4 with the subsequent tiling piece that is tiled during the process.

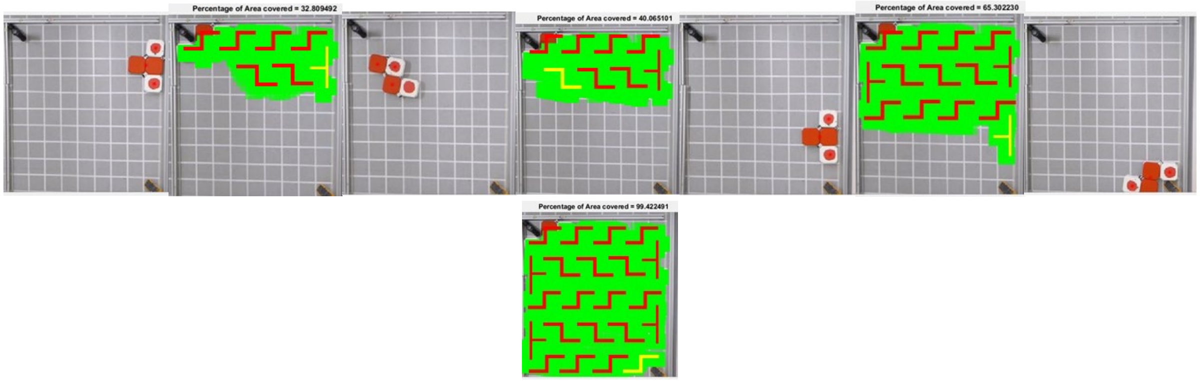

Percentage of Area covered = 32.809492
Percentage of Area covered = 40.065101
Percentage of Area covered = 65.302230
Percentage of Area covered = 99.422491

Figure 4.15. Tiling process with lemma 3 tiling set

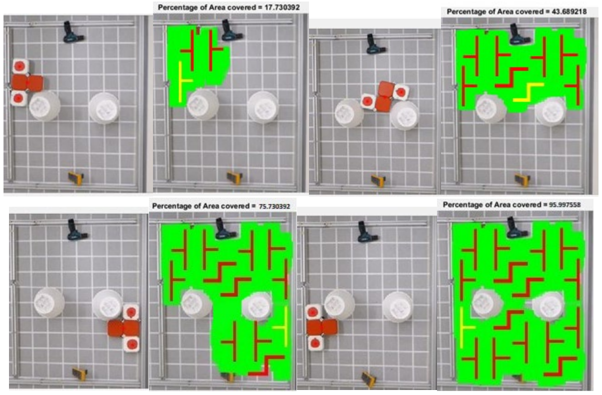

Percentage of Area covered = 17.730392
Percentage of Area covered = 43.689218
Percentage of Area covered = 75.730392
Percentage of Area covered = 95.997558

Figure 4.16. Tiling process with lemma 4 tiling set

Furthermore, with a fifth and sixth set of experiments, we changed the test area to 154 cm x 154 cm by adjusting the metal frames. We modified the testing area by placing an obstacle in the corresponding cell. The whole tiling process for both the experiments was recorded and post-processed to compute the total area covered using (2). The area coverage process during the validation of lemma 5 is shown in Fig 4. 19. In particular, we displayed the tiling process of the hTetro robot when it is covering 17%, 59%, 87%, and 99% of the total defined area. The first image of each set in Fig 4. 19, shows the O-, L- and J- configuration of the hTetro robot in the process of covering subsequent Tetris pieces. The result with the fifth experiment shows that the hTetro robot can cover more than 99% of the area while leveraging on Lemma 5. Similarly, in the last set of an experiment involving lemma 6, the hTetro achieved more than 99% of area coverage. Fig 4. 20 presents the area coverage process of the hTetro robot while it is covering 17, 59, 87, and 98% of total defined space. Also in Fig 4. 20 the odd images shows O-, L-, J- configurations of the hTetro robot when tiling the subsequent tiling sets. The experimental results indicate that there are significant untapped research and development opportunity related to the application of the polyomino tiling theory within the area coverage problem.

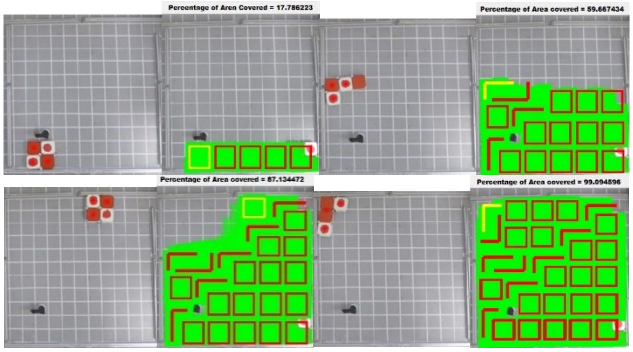

Percentage of Area Covered = 17.786223
Percentage of Area covered = 59.667434
Percentage of Area covered = 87.134472
Percentage of Area covered = 99.094896

Figure 4.17. Tiling process with lemma 5 tiling set

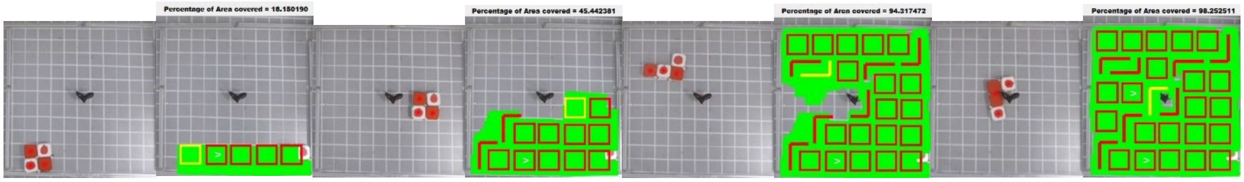

Percentage of Area covered = 18.150190
Percentage of Area covered = 45.442381
Percentage of Area covered = 94.317472
Percentage of Area covered = 98.252511

Figure 4.18. Tiling process with lemma 6 tiling set

4.3 Conclusion

This chapter introduced a unique area coverage technique for a Tetris-inspired shapeshifting robotic floor cleaner named hTetro that was based on the polyomino tiling theory. Specifically, the hTetro robot was employed to validate the application of three Tetris-tiling theorems. The results of the experiments clearly established the efficiency of the proposed technique, which successfully achieved superior floor coverage across all the experimental scenarios. By automating the process of creating a global polyomino tiling set for dealing with area coverage in our hTetro robot, we anticipate to greatly simplify the coverage path planning problem. This chapter also described the system architecture that underpinned the hTetro robot, together with details of the experimental scenario and design. Future research will focus on: (1) Integrating ultrasonic, bump sensor, and infrared for autonomous navigation functions; (2) Addressing computational complexity in the proposed approach in relation with energy and implementing optimization techniques to overcome those dispute; (3) Implementing dissipativity-based fuzzy logic integral sliding mode control system and SMC stabilization of nonlinear markovian jump singular systems in order to tackle the uncertainties on our physical hTetro system (4) the development of additional features by which power management issues and auto-docking modes can be addressed (5) the development of software algorithms that perform both autonomous global tile set generation and robot navigation and (6) the development of algorithms that can handle autonomous reconfiguration in relation to the perceived environment and selected tiling theory that targets the optimal coverage area.

5. The modularity concept and implementing motion planning technique on other class of polyomino robots

In this chapter, we present a novel application of Tromino tiling theory, a class of Polyomino with three cells in the context of a reconfigurable floor cleaning robot, hTromo. The developed robot platform is able to automatically generate a global tiling set required to cover a defined space while leveraging on the Tromino tiling theory. The main challenges in the proposed approach include the design of the reconfiguring mechanism, the inclusion of cleaning features and the non-trivial process of implementing theoretical Tromino tiling designs generated analytically into physical mechanisms. All these aspects are detailed in this chapter, concluding with experimental results using the prototype hTromo robot that validates the proposed approach. The application of Tromino tiling theory herein presented is a critical effort towards designing a self-reconfigurable robot that is capable of autonomously generating global tiling set for any given space, identifying associated local and global optimal trajectories and generating appropriate motor primitives based on inverse kinematics and dynamics model.

5.1 Polyomino Tiling Theory

Polyominoes are plain geometrical structures formed by endwise coupling of congruent squares. Based on spatial orientation, geometrical transformation and chirality, each polyomino can be categorized into free polyominoes, one-sided Polyominoes, and fixed Polyominoes. For instance, the set domino, formed by the combination of two congruent square, can have single one-sided, single free and two fixed dominoes as its subsets. Correspondingly, triominoes (3-omino) can exist as two free, two one-sided, and six fixed triominoes. In the case of tetromino that contains four constituent squares can form five free, seven one-sided, and nineteen fixed tetrominoes. In this chapter, we used a Tromino inspired reconfigurable robot, hTromo capable of switching between one of the three forms.

5.1.1 Polyominoes Tiling Theory:

The Polyominoes tiling theory deals with the problem of partitioning or filling of a geometrical region using same or multiple sub-regions. Literature offers numerous work that discusses tiling theorems with proof for distinct polyomino set. With Tromino forming the inspiration for our hTromo robot, this chapter presents our first attempt at applying Tromino tiling theory to coverage problem for a floor cleaning robot. Specifically, we apply five theorems proposed in [107], [108], and [109].

Theorem 1: *An a×b rectangle can be tiled with L- and I- trominoes if and only if the area of that rectangle is divisible by 3.*

Fig 5. 1 shows tiles τ2 and τ4 a right-oriented L- tromino piece, and tiles τ1 to τ3, a left-oriented L- tromino piece. In the experiments performed, we either used pre-defined testbed space that was either a complete rectangle or modified rectangle based on the theorem considered where the concerned robot was able to achieve complete coverage. Fig 5. 2 presents a sample case for Theorem 1 involving tiling using I- and L-tromino configurations.

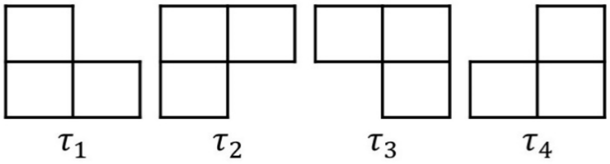


Figure 5.1. ‘T’ set of Tromino pieces

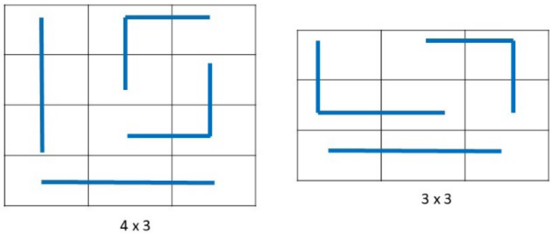


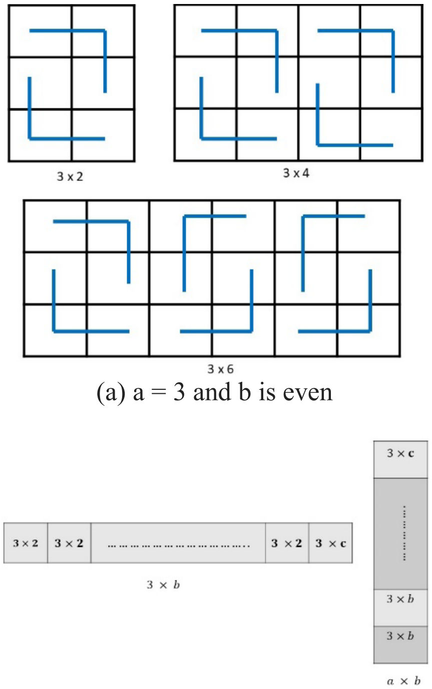
Figure 5.2. Tiling a ‘a x b’ rectangle with I- & L- Trominoes.

Theorem 2: *Let a, b be the integers such that 2 ≤ a ≤ b. An a×b rectangle can be tiled with set ‘T’ trominoes if and only if one of the following conditions holds:*

1. *a = 3 and b is even;*
2. *a ≠ 3 and ab is divisible by 3.*

‘T’ is a set of four L- trominoes. Since the tromino has three constituent squares with unit area, any rectangle with an area that is a multiple of 3 can be covered using any of the trominoes pieces. This notion leads to the formulation of Condition 2 of Theorem 2. Let a and b be the dimensions of the rectangle to be tiled. Since the area of the rectangle has to be a multiple of 3, $(a \times b)=3n$, where $n \in \{1,2,3\ldots\ldots\}$. This implies that either a or b must be divisible by 3. However according to condition 2, a cannot be 3. For example, the rectangles that have a dimension of 6×5, 4×9, 12×6 can be tiled using the set of ‘T’ trominoes. Condition 1 of Theorem 2 excludes certain rectangles that cannot be tiled using ‘T’ trominoes. Specifically, according to this theorem, it is impossible to tile rectangles that have a dimension of 3×5, 3×7, 3×9 using any arrangements of ‘T’ trominoes. The Lemma 1 and Lemma 2 detailed below validates Theorem 2.

Lemma 1: Let a=3 and b ∈ {2,4,6} then a 3×b rectangle can be tiled using the arrangement of ‘T’ trominoes shown in Fig 5. 3(a). Hence it is proved that the smallest rectangle that satisfies condition 1 theorem 2 is (3x2). Let b > 6, even numbers, and $c \in \{2,4,6\}$, then $b = 3n+c$, where n can have a positive even integer value. As such, it allows for splitting of a $(3 \times b)$ rectangle into n (3×2) rectangles and one $(3 \times c)$ rectangle, as in Fig 5. 3(b). This implies, if b ≥ 2, even numbers, then a $(4 \times b)$ rectangle can be tiled using a set of ‘T’ trominoes.

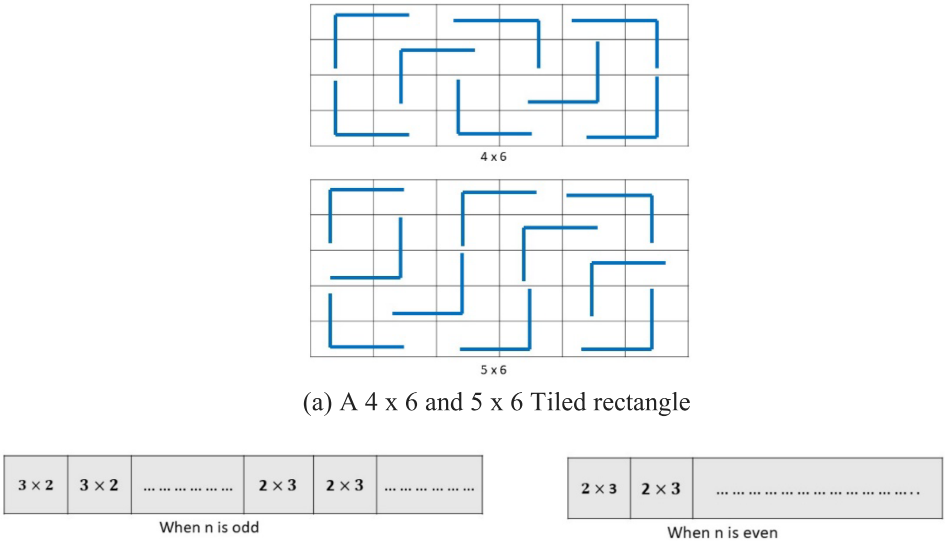


(a) a = 3 and b is even

(b) Decomposing a x b rectangle into sub-rectangles

Figure 5.3. Image Argument for Lemma 1.

Lemma 2: According to condition 2 of Theorem 2, the smallest rectangle that can be tiled using 'T' trominoes is (2 × 3). Let a = 4; then according to condition 2, the possible b values must be divisible by 3. Let's assume the dimension of the rectangle is 4 × 6, Fig 5. 4(a); then it is possible to decompose it to a 4 (2 × 3) rectangles. Hence, a rectangle with a dimension was a > 3, and b is a multiple of 3, then the rectangle could be decomposed to n(2 × 3) rectangles as in Fig 5. 4(b), which can be tillable with 'T' trominoes. However, the rectangle with dimensions of 5 × 3, 7 × 3, and 9 × 3 cannot be tiled using any arrangement of 'T' trominoes.



(a) A 4 x 6 and 5 x 6 Tiled rectangle

(b) Decomposition on a x b rectangle according to Lemma 2

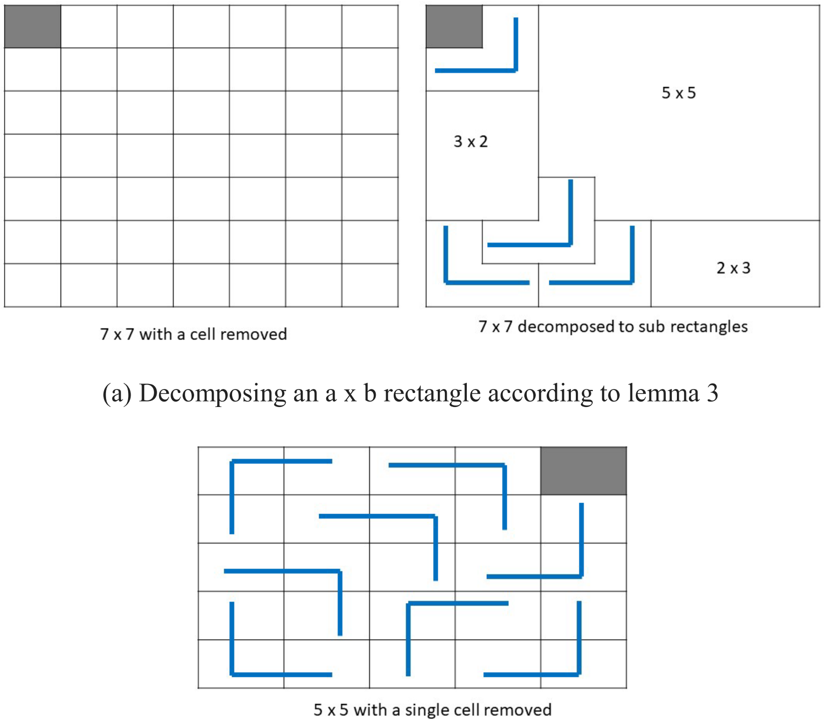
Figure 5.4. Image Argument for Lemma 2

Theorem 3: *A deficient n X n rectangle board can be tiled with the set of 'T' trominoes, if, and only if, either :*

1. *n is odd, and >5, and* $n^2 - 1$ *is divisible by 3.*
2. *n is even, and >1, and* $n^2 - 1$ *is divisible by 3*

Also with Theorem 3, we are utilizing set 'T' tromino pieces in order to tile the given area. The term deficient in the theorem describes a square grid of side a and b with a single cell truncated. According to Theorem 3, if the sides of the squares are odd, it should be greater than 5, and to the square of that sides subtracted by 1 must be divisible by 3 to tile the space using 'T' trominoes. In order to tile an even-sided deficient square with 'T' trominoes, the value must be greater than 1 and, to the square of that value subtracted by 1 must be divisible by 3. The above notion regarding tiling the deficient square points to Condition 1 and Condition 2 of Theorem 3. Lemma 3 and 4 validate Theorem 3 by proving Conditions 1 and 2.

Lemma 3: Let M (a, b) be the modified rectangle and segmented into an (i, j) grids. Let's consider an M(7x7) rectangle grid with a single (1, 1) square is removed Fig 5. 7 (a Left). As shown in Fig 5. 5(a Right), the considered rectangle can be decomposed to a 2x3, 3x2, 5x5, and three separate L-trominoes. According to Lemma 1, a 2x3, and 3x2, a rectangle can be easily tillable with 'T' tromino set. When it comes to a 5 x 5 square, it is tileable with 'T' trominoes only if the corner cells are removed as shown is Fig 5. 7 (b). Hence, the results show that an M(7x7) rectangle with single cell removed is tillable with 'T' trominoes. It is also possible to tile an M(7x7) rectangle when we delete (1, 4), (2, 3), (2, 4), or (4, 4) cells.

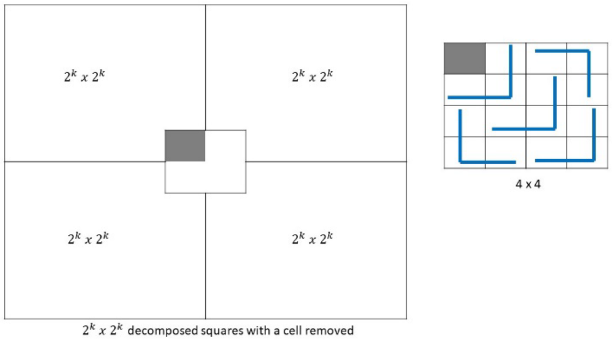

5 x 5
3 x 2
2 x 3
7 x 7 with a cell removed
7 x 7 decomposed to sub rectangles
5 x 5 with a single cell removed

(a) Decomposing an a x b rectangle according to lemma 3

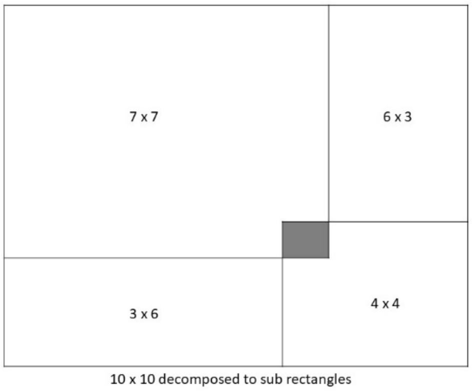
(b) A 5 x 5 modified tiled rectangle

Figure 5.5. Image Argument for Lemma 3

Lemma 4: Similarly, with condition 2 theorem 3, we are considering an M(10x10) rectangle. The considered rectangle can be split into a 7x7, 6x3, 3x6, and 4x4 sub-rectangles shown in Fig 5. 6(b). According to lemma 3 a 7x7 rectangle with one square removed can be tiled with 'T' tromioes. Similarly, with the help of lemma 1 & 2, we know that 6x3, and 3x6 can also be tiled with 'T' trominoes. For a 4x4 square, it is proven that a $2^k x 2^k$ when k≥ 1 can be tiled with 'T' tominoes as shown in Fig 5. 6(a). Hence, it is proved that a rectangle with even sided can be tiled using set of 'T' tromino pieces.



(a) Decomposing a Modified rectangle with a 4 x 4 tiled modified rectangle



(b) A 10 x 10 modified decomposed rectangle

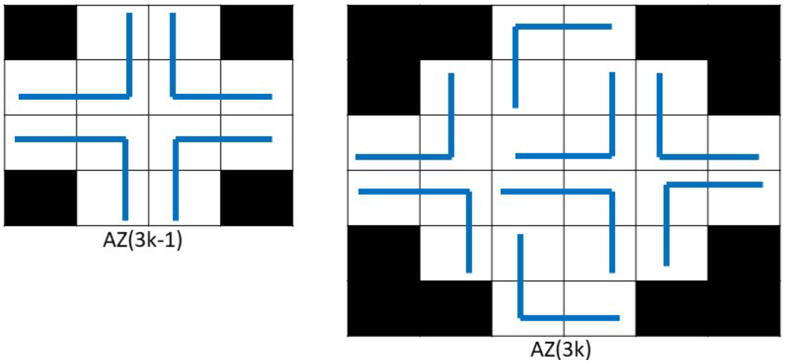
Figure 5.6. Image Argument for Lemma 4

Theorem 4: *An Aztec Diamond, AZ(n) can be tiled using 'T' set of tromino tiling pieces if, and only if n(n+1) ≡ 0 mod 3, where n can be a positive integer.*

Theorem 5: *A deficient Aztec Diamond, AZ(k) can be tiled using 'T' set of tromino tiling pieces if, and only if k=(3n-2), where n can be a positive integer.*

An Aztec diamond of order n is the region obtained from staircase shapes of height n by gluing them together along the straight edges. According to Theorem 4, a non-deficient Aztec diamond can be tiled using the 'T' set of tromino pieces if and only if n (n+1) ≡ 0 mod 3. Hence, it is clear that the tabbed n^{th} order must be divisible by 3 after applying it to n(n+1), were n can be any integer. Similarly with theorem 5, it is clear that, we can tile a defected Aztec diamond (with one square removed) using 'T' tromino set only if the n^{th} order is equivalent to 3n-2, where n can be any integer. Below mentioned Lemma 5 and 6 supports the conditions of theorem 4 and theorem 5.

Lemma 5: Note that the only values for which n(n + 1) ≡ 0 (mod 3) holds are n = 3k or n = 3k − 1 for some unique positive integer k. Thus, the statement is equivalent to say that for all positive integers k there is a tiling for AZ(3k) and AZ(3k − 1) as shown in Fig 5. 7 but there is no tiling for AZ(3k − 2). The tiling of Aztech diamond can be achieved by tiling the edges of considered space. We used stairs concept in order to tile the edges of the Aztec diamond. A stair is a polyomino made-up of tromino pieces wherein their 180◦ rotations are connected as steps shown in Fig 5. 8. The height of the stair was computed under the formulation of 3k + 2, for any positive integer k. The height of the stair is equal to the order of Aztec diamond. If the order of Aztec diamond is n ≤ 4, then AZ (n) can be tiled using 'T' trominoes through tiling partial or half stairs as shown in Fig 5. 7 (Top Right & Left). If the order n ≥ 5, then the AZ (n) can be tiled using K-stair of 'T' tromino pieces. The image argument for order n = 5 is shown in Fig 5. 7 (bottom), green colored tromino pieces tiled the edges using 1-stair.



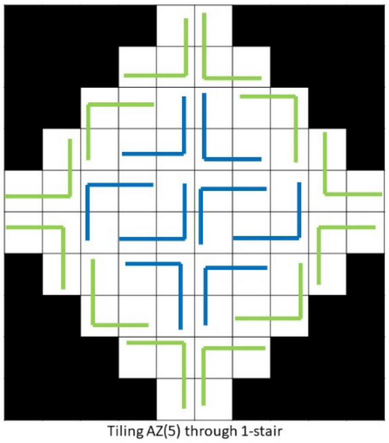


Figure 5.7. Image Argument for Lemma 5

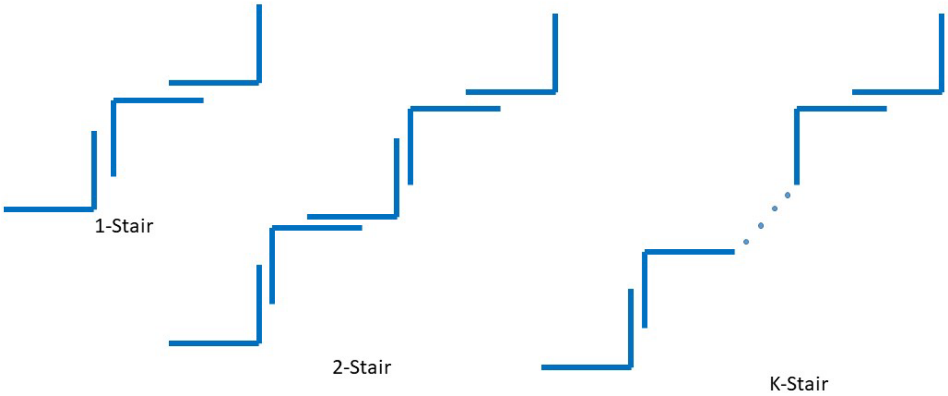


Figure 5.8. Stair Edge Tiling

Lemma 6: To tile AZ(3k − 2) with one defect a fringe appearing has been used as shown in Fig 5. 9(left). It is easy to check that if a fringe has exactly one defect, then it can be covered with 'T' trominoes. In particular, a possible rectangle with a fringe that can be tiled by 'T' tromino is 2 x 2 square. Similarly, as lemma 5, the stair patterns were used to tile the edges of the concern defected Aztec diamond. The space with fringe placed is considered as a 2 x 2 square plot and can be tiled using 'T' trominoes Fig 5. 9(left). The other space of defected Aztec diamond were tiled similarly to Lemma 5. In another approach, tiling pattern of Fig 5. 8 was used wherein the k-stairs laid above and below the fringe in order to tile the Aztec diamond with 'T' trominoes, as shown in Fig 5. 9 (Right).

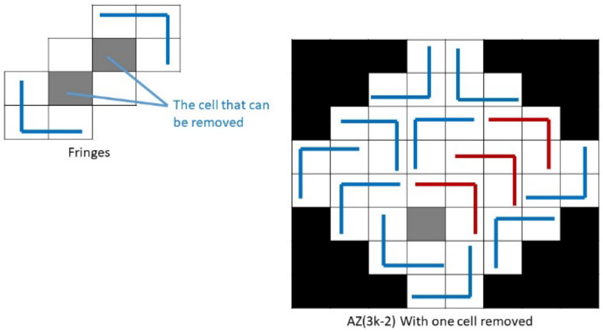
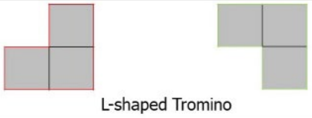
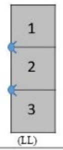
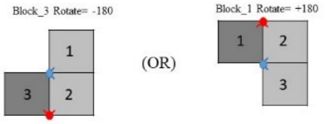
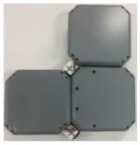
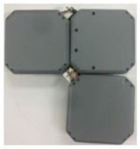


Figure 5.9. Image Argument for Lemma 6

5.2 hTromo: Robot Architecture

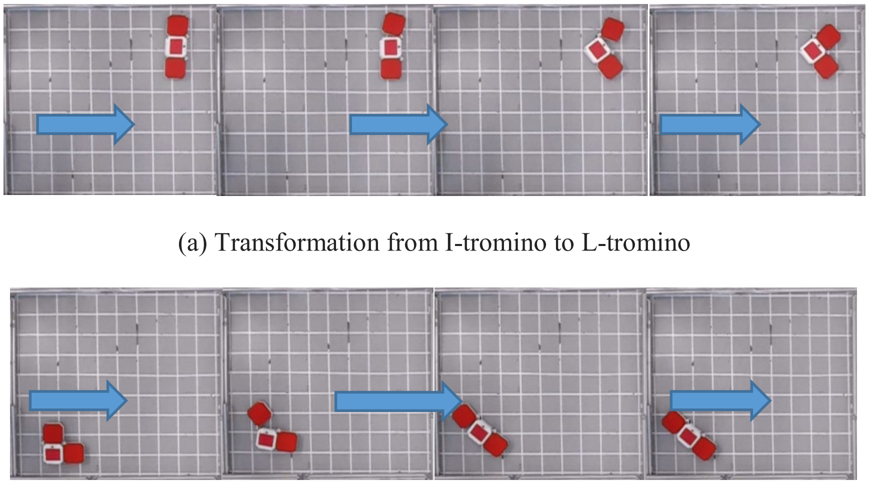
The experiments presented in this chapter involves a novel area coverage technique with Tromino tiling theory as a basis. The proposed approach was validated on the hTromo; a Tromino inspired reconfigurable floor cleaning robot that was developed based on the theory of “hinged dissection of polyominoes”. Hinged dissection is a geometric analysis wherein a planar structure dissected into finite pieces connected by “hinged” points, such that the formation of one structure to another can be accomplished by continuously swinging the hinged points without breaking the chain. Several studies targeting the concept of hinged dissection have been reported. Pertinent efforts in this field have included the remodeling of an equilateral triangle into a polygon, combining several rigid duplicates of the same polyhedron, the creation of unique patterns through rearrangement of shapes from one to another, and 3D hinged dissection being used for the formation of 3D polyhedra. In robotics, studies the hinged dissection principle, with a view to creating a nested re-configurable robot module named ‘hinged-tetro’, and to demonstrating that LLR or LLL hinged dissection applies to the creation of all one-sided tetromino forms. The LLR hinged dissection method was utilized in the creation of the hTetro cleaning device, in order to achieve the transformational capability. Since hTromo robot only consists of three blocks and requires only two hinged points; we utilized LL hinged dissection configuration to achieve the shape-shifting ability outlined in Table 4. 1.

Table 1. LL hinged points of hTromo robot.

HINGED DISSECTION	 L-shaped Tromino
	 Block_3 Rotate= -180 1 3 2 (OR) Block_1 Rotate= +180 1 2 3
hTromo_I_config	 Right Oriented hTromo_L_config  Left Oriented hTromo_L_config

5.2.1 Mechanism Design

The hTromo robot comprises three squares, and two (LL) hinged points, to facilitate transformation. Fig 5. 10 shows the intermediate form of the hTromo robot while transforming from one configuration to another. Fig 5. 11 shows the detailed component list for the device. Block 1 contains the requisite components for movement, block 2 the electronic peripherals, and blocks 3 the cleaning components and functionality. A single hTromo block measures 140 x 140 x 75mm, with 4mm thick honeycomb walls, formed from PLA for minimum tensile strength. The device houses six DC motors, with four mounted on block 1 and controlled fundamental mobility. Two further DC motors sit on block 3, optimizing smooth mobility for the robot. For ease of navigation within an area, the robot has omnidirectional movement capabilities. Furthermore, caster wheels were attached to blocks 2 and 3 to maintain the position of the blocks. Blocks 3 also house a tailor-made vacuum module, for cleaning functionality, and the collection of dirt. The suction chamber and duct were specifically created to minimize loss of suction and dust spillage during operation. For the creation of the three available one-sided Tromino shapes, the robot houses two smart servos that sit on hinged points. Both the servos attached to, and anchoring, block 2 and driving blocks 1 and 3. Every motor fixes its position with its stall torque, allowing it to continually support robot morphology for the duration of its operation.



(a) Transformation from I-tromino to L-tromino

(b) Transformation from L-tromino to I-tromino

Figure 5.10. hTromo robot’s transformation from one configuration to another.

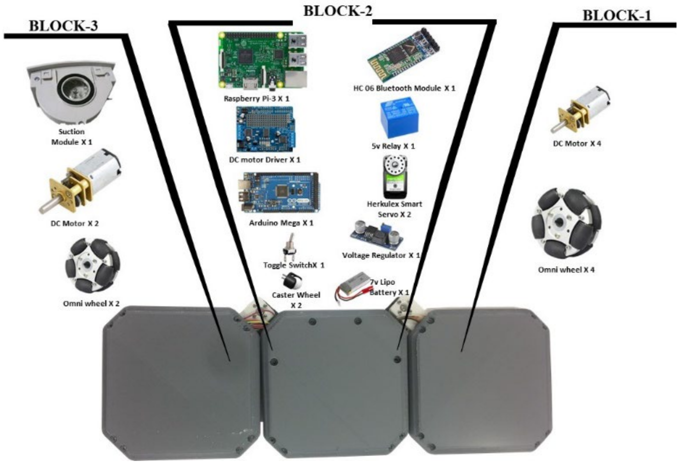


Figure 5.11. Components list of hTromo robot

For a low-level controller, block 2 houses an Arduino Mega 16-bit microcontroller to manage movement and transformation gait performance. Action commands are sent by Raspberry PI which is also mounted on block 2, with the microcontroller executing these commands, sending PWM motor primitives to the driver unit. Integrated electronics are powered by a LiPo battery, providing 7.4v through a toggle switch. In block 2, a DC step-down controller maintains a 5v input voltage for the Raspberry PI. The vacuum module is operated through activation of a relay switch. For user commands to reach the Raspberry PI, a wireless Bluetooth interface is used. Fig 5. 12 shows an intuitive Android App that was developed for the control of the hTromo robot. To maneuver and transform the robot, and operate its cleaning functions, users select commands from a list of pre-set options. The app provides arrow buttons for directional commands, seven Tetris buttons for shape reconfiguring, pause and play buttons to freeze or commence actions, and on and off buttons for vacuuming operations.

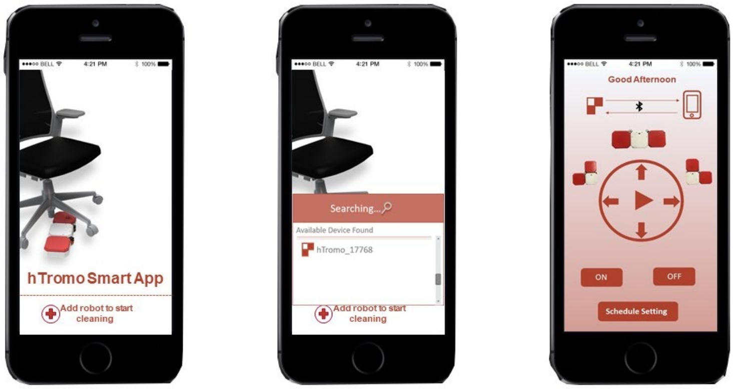


Figure 5.12. hTromo User Interface.

5.3 Experiments and Results

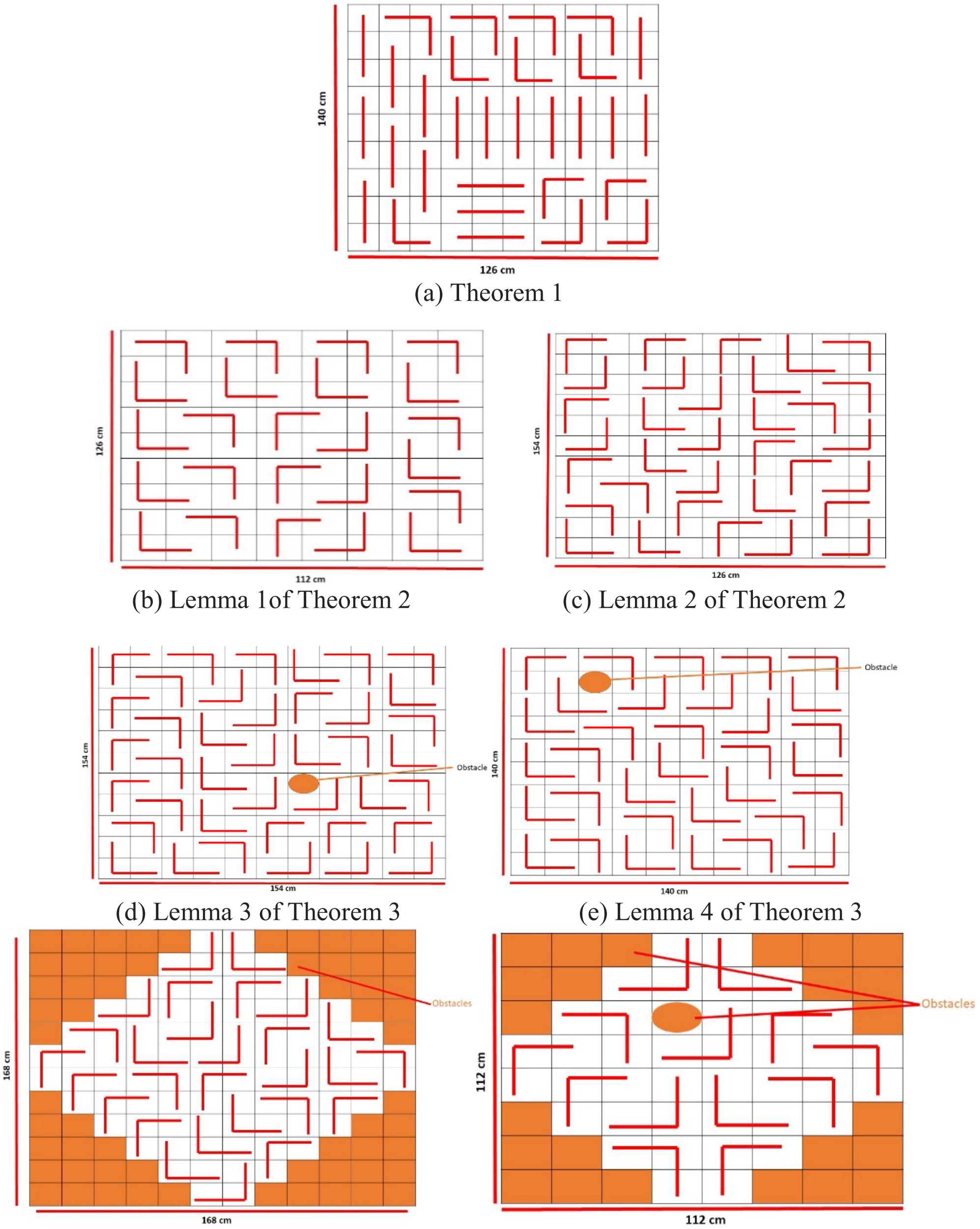
In this chapter, we present the application of Tromino tiling theory, a class of Polyomino with three cells in the context of a reconfigurable floor cleaning robot, hTromo. The developed robot platform is able to automatically generate a global tiling set required to cover a defined space while leveraging on the Tromino tiling theory. There are five set of theorems that this work sought to validate, as denoted in Section II. The first set of experiments validated the application of Theorem 1 using a rectangular surface of 140 x 126 cm, split into 10 x 9 squares. Fig 5. 13(a) presents the universal tiling set that was auto-generated by our path planning algorithm to cover the given area using only L- and T-trominoes. In order to validate the application of Theorem 2, Lemma 1, a rectangular area measuring 126cm x 112cm was utilized and further sub-divided into 9 x 8 square grids. Fig 5. 13(b) illustrates the corresponding universal tiling set auto-generated by our path planning algorithm using which tiles the second test area with only T-set trominoes. The third set of experiments validates Theorem 2, Lemma 2. This was done in a rectangular area of 154cm x 126cm, split into 11 x 9 square grids.

Fig 5. 13(c), shows the corresponding tiling set generated based on Theorem 2, Lemma 2. In the fourth set of experiments, obstacles were inserted within the test area in order to modify the area based on Theorem 3, Lemma 3. We utilized a square area of 154cm x 154cm, subdivided into 11 x 11 square grids. According to the assertion of Lemma 3, this modified area can be tiled using a 'T'-set of tetromino pieces. Fig 5. 13(d) illustrates the universal tiling set that was auto-generated based on Theorem 3, Lemma 3. The fifth set of experiments focused on validation of Theorem 3, Lemma 4. The test was performed within a square area of 140cm x 140cm, further subdivided into 10 x 10 square grids. To meet the requirements of Lemma 4 towards realizing a modified rectangle, obstacles were inserted into the middle of the test area. The global tiling set that was auto-generated based on Lemma 4 can be seen in Fig 5. 13(e).

With lemma 5 of Theorem 4, we used a square plot as a test arena with a dimension of 168 cm x 168 cm and segmented it into 12 x 12 square grids. Since the concern theorem deals with Aztec diamond space, we modified the defined area into a 6th order diamond space by placing obstacles. Fig 5. 13(f) shows the tiling set generated with 'T' trominoes according to the lemma 5 and the orange shaded areas are filled with obstacles. Similarly, for Lemma 6, we used a 112 cm x 112 cm square plot which was segmented to an 8 x 8 square grids. The considered area was filled with obstacles in order to convert the square area to a 4th order Aztec diamond. Also, we placed a separate obstacle in the 4, 4 cell to modify area as a deficient diamond. Fig 5. 13(g) shows the tiling set generated with 'T' trominoes according to the lemma 6.

5.3.1 Experimental testbed:

When establishing the test environment, a pre-determined floor area was split into squares, congruent with the hTromo robot blocks, and an overhead support frame mechanism erected, to accommodate a camera. Image data captured by this camera was post-processed, to evaluate the percentage of the test area covered by hTromo during each experiment. The complete area used for all experiments measured 196 x 196cm. The limits of the specified test area were adapted using an extendable metal framework, according to the assertions of each theory. The test area was further split, using white tape, into a 14 x 14 square grid. As shown in Fig 5. 14, every square within this grid mirrored the measurements of a single hTromo robot block.



(a) Theorem 1

(b) Lemma 1of Theorem 2

(c) Lemma 2 of Theorem 2

(d) Lemma 3 of Theorem 3

(e) Lemma 4 of Theorem 3

(f) Lemma 5 of Theorem 4

(g) Lemma 6 of Theorem 5

Figure 5.13. Tiled Testing Area with I- and L- trominoes

A shake-resistant parallelepiped was created using an aluminum extrusion profile, with a camera then attached in the center, at the top of the structure. Furthermore, the image plane was verified as being horizontal to the floor, to avoid perspective projection complications for area calculation. As such, when the camera was attached to the structure, a spirit level was used to ensure it sat parallel to the ground. The camera set-up and corresponding test area can be seen in Fig 5. 15. Auto-focus was turned off, and a set focal length employed when recording robot tiling. The raw video recordings were subsequently post-processed, in order to assess robot performance.

To generate a tracking map for the hTromo movement, an image-processing calculation was employed, which comprised three key stages. The first of these stages concerned the storage of a reference image, from which a track map would be created. Secondly, the location and form of the robot were identified in each frame. After multichannel color thresholding, the algorithm recognized the robot as three spots. The center of these spots corresponds to the center of each red color. Finally, a track map was created, by marking the green squares in accordance with spots detected on the reference image. When a track map was ready, a percentage calculation was conducted using the following formula, to assess the total area coverage achieved.

$$\%Coverage\ Area = \frac{\text{Pixels area of the robot}}{\text{Total pixels area of the testing field}}\ X\ 100 \quad (5.1)$$

$$\%\ Coverage\ Area = \frac{\text{Pixel area of the robot}}{\text{Total pixel area of the testing field} - \text{Total pixel area of the obtacles}} X100 \quad (5.2)$$

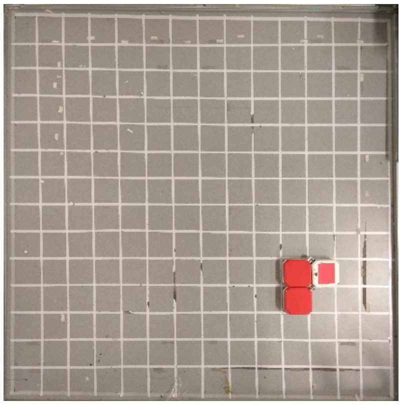


Figure 5.14. Defined test area with segmented square grids

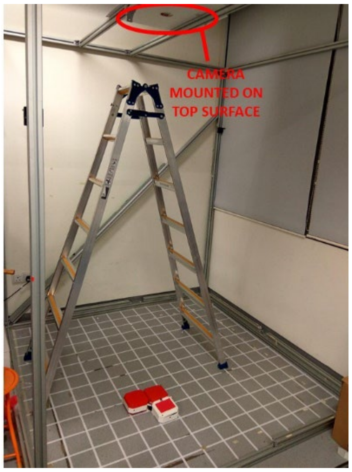


Figure 5.15. Test area with parallelepiped structure with provision for camera mount

5.3.2 Results and Analysis:

Each experiment was started after placing the hTromo robot in a predefined position inside the area. We recorded the robot in action from the beginning to the end of the experiments. Once the tests were completed, the recorded videos were post-processed using the image processing algorithm detailed in section IV in order to generate the track map. Fig 5. 16 presents the track map images of the hTromo robot that were generated following the first set of experiments that sought to validate the application of Theorem 1. The green colored shading represents the area covered by our hTromo robot. The percentage of area covered was computed using equation (1) and is displayed on top of the tracked images. Fig 5. 17 presents the tiled area during different stages of our first set of experiment. The figure indicates the actual position of the robot at a specific time point, and the associated track map at that instance overlaid with the completed tiling set. The robot path was executed according to the global tiling set specified in basic tromino theory. The results clearly show that our hTromo robot covered more than 95% of the test area in the first set of experiment validating the underline theory of tromino tiling. In the second set of an experiment involving Lemma 1 of Theorem 2, the area covered by our hTromo robot was found to be over 95%. Fig 5. 16 presents the tiled area during different stages of our second set of experiment. We extended the testing area by adjusting the metal frames as mentioned in section IV (a), in order to validate Lemma 2 of Theorem 2. The same process has been followed to generate the track map in order to compute total area covered. The results show that the htromo covered more than 91 % of the defined testing area thereby validating the application of Lemma 2 shown in Fig 5. 18.

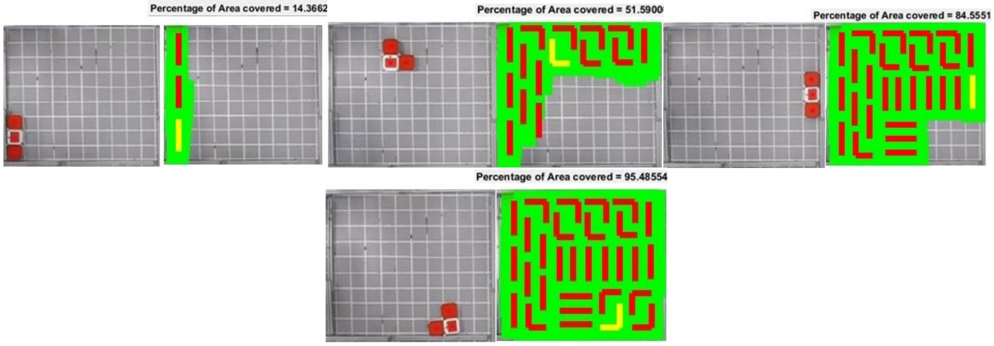

Percentage of Area covered = 14.3662
Percentage of Area covered = 51.5900
Percentage of Area covered = 84.5551
Percentage of Area covered = 95.48554

Figure 5.16. Experiments validating application of Theorem 1

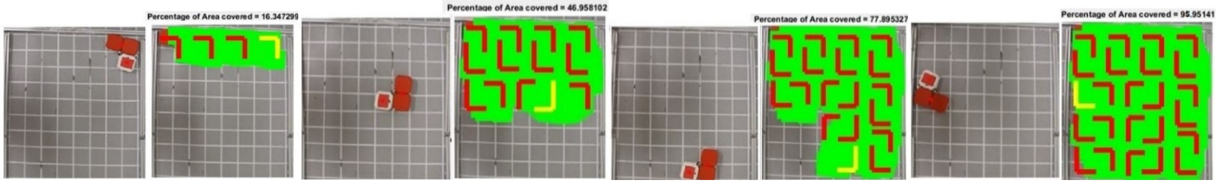

Percentage of Area covered = 16.347299
Percentage of Area covered = 46.958102
Percentage of Area covered = 77.895327
Percentage of Area covered = 95.95141

Figure 5.17. Experiments validating application of Theorem 2, Lemma 1

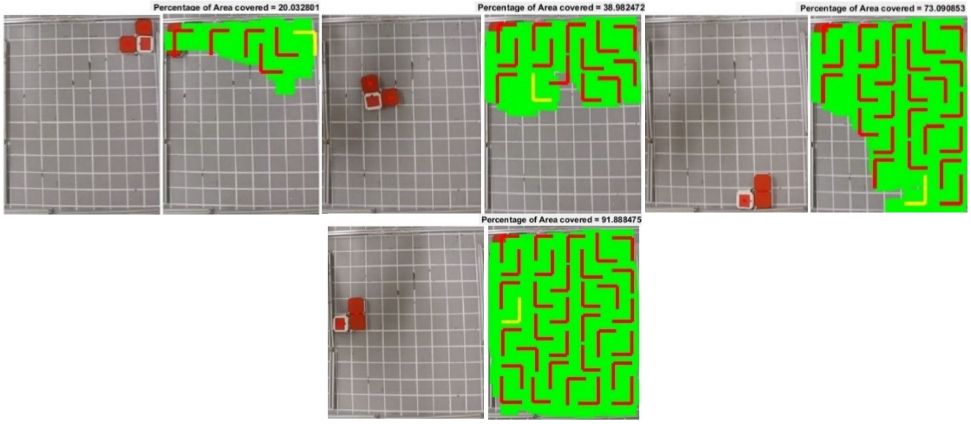
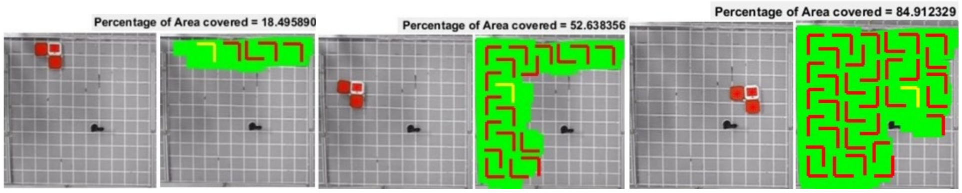

Percentage of Area covered = 20.032801
Percentage of Area covered = 38.982472
Percentage of Area covered = 73.090853
Percentage of Area covered = 91.888475

Figure 5.18. Experiments validating application of Theorem 2, Lemma 2

In the fourth and fifth set of experiments, obstacles were placed inside the testing area as a means of modifying the test space as required by the Lemma. The boundaries of the testing area were adjusted according to the arguments in Lemma 3 and 4 of Theorem 3. We computed the total area covered by excluding the pixels associated with the obstacles by utilizing equation (2). Fig 5. 19 depicts the area coverage process while validating the application of Lemma 3. Results clearly show that hTromo robot covered an area in excess of 97% of the total test area, thereby validating the application of the Lemma 3 of Theorem 3. Similarly, with experiments involving Lemma 4, the hTromo robot covered an area of over 95% of the total test area. Fig 5. 20, presents the tiled area during the different stages of a fifth and final set of experiments.


Percentage of Area covered = 18.495890
Percentage of Area covered = 52.638356
Percentage of Area covered = 84.912329

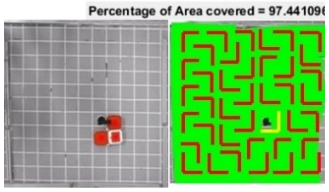


Figure 5.19. Tiling Process with Lemma 3 tile set

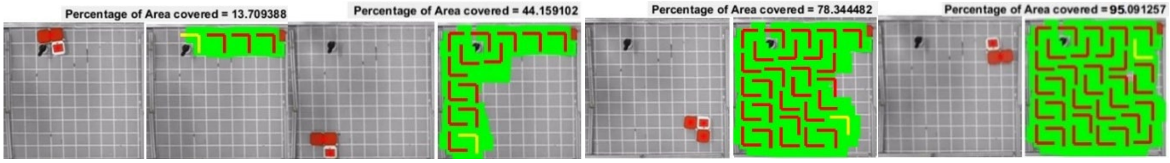

Percentage of Area covered = 13.709388
Percentage of Area covered = 44.159102
Percentage of Area covered = 78.344482
Percentage of Area covered = 95.091257

Figure 5.20. Tiling Process with Lemma 4 tile set

Furthermore, with the sixth set of experiments, we changed the test bed size to 168 x 168 cm by adjusting the metal frames. We placed obstacles inside the test area in order to make the square space into an Aztec diamond. We again followed the same experimental procedure to generate the tack maps to compute the total area covered. Since we placed obstacles inside the test arena; we used equation (2) to compute the total area covered. Fig 5. 21 shows the area coverage process during the validation of lemma 5. The results show that the hTromo robot covers more than 97% of the defined area, thereby validating the application of the Lemma 5 of Theorem 4. The experiments that involve lemma 6 of Theorem 5 was tested in a 112 cm x 112 cm area. The area coverage of hTromo robot under this experiment was computed using the equation (2). Fig 5. 22 shows the coverage process of the hTromo robot during the validation lemma 6. The results show that hTromo robot can achieve a coverage area of more than 92 %, through validating the application of Lemma 6 of Theorem 5. The experimental results clearly indicate that there are significant untapped research and development opportunity related to the application of the polyomino tiling theory within the area coverage problem.

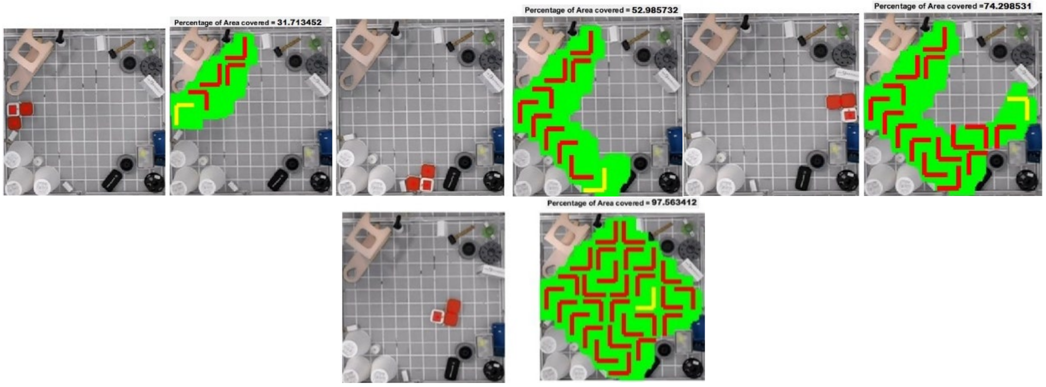

Percentage of Area covered = 31.713452
Percentage of Area covered = 52.985732
Percentage of Area covered =74.298531
Percentage of Area covered = 97.563412

Figure 5.21**.** Tiling Process with Lemma 5 tile set

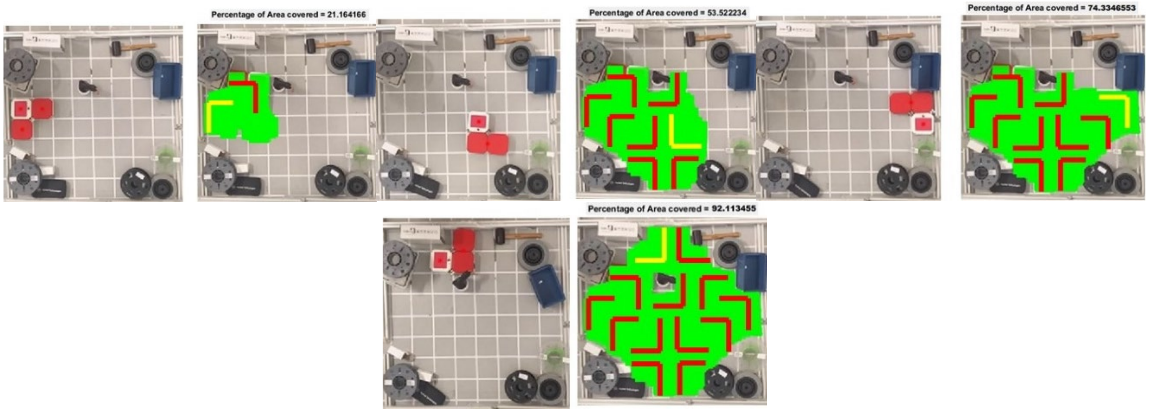

Percentage of Area covered = 21.164166
Percentage of Area covered = 53.522234
Percentage of Area covered = 74.3346553
Percentage of Area covered = 92.113455

Figure 5.22. Tiling Process with Lemma 6 tile set.

5.4 Conclusion:

In this chapter, we proposed a novel area coverage approach for a reconfigurable floor cleaning robot, hTromo using Tromino tiling theory. Specifically, we validated the application of five tromino tiling theorems with our hTromo robot. Experiments performed clearly demonstrate the efficacy of the proposed approach resulting in very high levels of area coverage performance in all considered experimental cases. By automating the process of generating a global tiling set for tackling area coverage in our hTromo robot, we hope to simplify the path planning problem greatly. This chapter also introduced the system architecture of our hTromo robot and details of experimental design and testbed. Future research will focus on: (1) integration of infrared, ultrasonic, and bump sensors for obstacle avoidance functions; (2) Optimising robot's path planning by implementing the Decision Making Framework for Human-Robot Collaborative Workplace Generation framework (3) Experimenting the proposed approach in a larger space by increasing the floor area and computing the total area covered within a given time period and (4) exploring global and local path planning by utilising the images captured from the overhead camera.

6. Conclusion:

This chapter summarizes a series of conclusions from the results obtained in the previous chapters. Also, this chapter gives an overview of this thesis that reiterates the developed hTetro robot which implements the hinged dissection of polyominoes. Also this chapter gives the overview of the mechanical design, electrical system and HMI modules of the hTetro robot and its ability to transform between any of seven one-sided tetromino morphologies to maximize floor coverage. This chapter not just discuss on physical design of hTetro but also gives an idea about the implementation of unique area coverage technique for the same that was based on the polyomino tiling theory. At the end, this chapter explains the experimental validation of the proposed techniques and its advantages. This chapter finally conclude that in all considered experimental cases, the hTetro robot exhibited a significant performance advantage in terms of floor coverage area due to its ability to assume optimal morphologies while navigating its environment

6.1 hTetro- A Tetris Inspired Shape shifting floor cleaning robot

we have presented a novel reconfigurable Tetris-inspired floor cleaning robot called hTetro that implements the hinged dissection of polyominoes. We introduced the mechanical design, electrical system and HMI modules of the hTetro robot and successfully validated its ability to transform between any of seven one-sided tetromino morphologies to maximize floor coverage. Experiments were performed in two different settings to systematically compare the coverage area performance of the developed hTetro robot with two commercially available circular and 'D'-shaped fixed morphology robot platforms. We tracked the test robots using an overhead camera on top of a constructed testbed to generate the robots' track MAPS. The track maps generated from each set of experiments were post-processed to measure the area covered by the robots. In both sets of experiments, the hTetro robot exhibited a significant performance advantage in terms of floor coverage area due to its ability to assume optimal morphologies while navigating its environment.

Future research work is set to mainly focus on the following areas: the first is to integrate range and bump sensors to enable autonomous navigation in the hTetro robot. The second is to study the application of Polyomino tiling theory towards facilitating global path planning through automatic generation of global tiling set required to cover a defined space. Moreover, we intend to implement highly adaptive locomotion control that allows for smooth trajectories at low computational costs across all seven configurations.

6.2 hTetro- Novel Motion planning techniques for a Tetris inspired reconfigurable robot

This chapter introduced a motion planning technique to achieve maximum area coverage in a Tetris-inspired shape shifting robotic floor cleaner named hTetro that was based on the polyomino tiling theory. We discussed on autonomous Tile set genera- tion based on the tiling theory concept and introduced the path generation for the hTtero robot to navigate on the generated tiling set with an objective of maximizing the area coverage. We validated the proposed algorithm by benchmarking its per- formance with traditional coverage path planning techniques. In each experiment, the robot cleared the sequenced waypoints while assuming its morphology at each point. At the end of the performed experiments, the waypoint navigation map and the coverage heat map are being generated. The experimen- tal results establish that the proposed tiling motion technique achieves maximum coverage area with less distance traveled and minimum re-covered area. Future research work is set to focus on the following areas: (1) Optimizing the reference point generation for the created tiling set with respect to energy cost
(2) Optimizing the tiling set generated where less number of reconfiguration is required (3) Exploring other optimal tech- niques for path generation to cover the entire area (4) Studying locomotion control that generates smooth trajectories in realiz- ing the tiling set. (5) Implementing the proposed approach for other polyforms reconfigurable robots.

6.3 The implementation of proposed motion planning technique on Tetris inspired reconfigurable robot

This chapter introduced a unique area coverage technique for a Tetris-inspired shapeshifting robotic floor cleaner named hTetro that was based on the polyomino tiling theory. Specifically, the hTetro robot was employed to validate the application of three Tetris-tiling theorems. The results of the experiments clearly established the efficiency of the proposed technique, which successfully achieved superior floor coverage across all the experimental scenarios. By automating the process of creating a global polyomino tiling set for dealing with area coverage in our hTetro robot, we anticipate to greatly simplify the coverage path planning problem. This chapter also described the system architecture that underpinned the hTetro robot, together with details of the experimental scenario and design. Future research will focus on: (1) Integrating ultrasonic, bump sensor, and infrared for autonomous navigation functions; (2) Addressing computational complexity in the proposed approach in relation with energy and implementing optimization techniques to overcome those dispute; (3) Implementing dissipativity-based fuzzy logic integral sliding mode control system, and SMC stabilization of nonlinear markovian jump singular systems in order to tackle the uncertainties on our physical hTetro system (4) the development of additional features by which power management issues and auto-docking modes can be addressed (5) the development of software algorithms that perform both autonomous global tile set generation and robot navigation and (6) the development of algorithms that can handle autonomous reconfiguration in relation to the perceived environment and selected tiling theory that targets the optimal coverage area.

6.4 The modularity concept and implementing motion planning technique on other class of polyomino robots

This chapter summarizes a series of conclusions from the results obtained in the previous chapters. Also, this chapter gives an overview of this thesis that reiterates the developed hTetro robot which implements the hinged dissection of polyominoes. Also this chapter gives the overview of the mechanical design, electrical system and HMI modules of the hTetro robot and its ability to transform between any of seven one-sided tetromino morphologies to maximize floor coverage. This chapter not just discuss on physical design of hTetro but also gives an idea about the implementation of unique area coverage technique for the same that was based on the polyomino tiling theory. At the end, this chapter explains the experimental validation of the proposed techniques and its advantages. This chapter finally conclude that in all considered experimental cases, the hTetro robot exhibited a significant performance advantage in terms of floor coverage area due to its ability to assume optimal morphologies while navigating its environment

References

[1] Angie Parkinson (2017, 5th April). Moneual RYDIS H68 Pro Review. *Top Ten Reviews.* Retrieved April 22nd 2017 from: http://www.toptenreviews.com/home/vacuum-carpet-cleaning/best-robot-vacuums/moneual-review/

[2] Anonymous (2017). Neato Botvac Series. *Neato Robotics.* Retrieved April 22nd 2017 from: https://www.neatorobotics.com/robot-vacuum/botvac/

[3] Anonymous (2017). Robotic Vacuum Cleaner – Tech Talk. *SHARP Corporation.* Retrieved April 22nd 2017 from: http://www.sharp-world.com/products/robotic_appliance/tech/index.html

[4] Anonymous (2017). SR8900 ROBOT VC with Auto-emptying Dust Bin, 40 Watt, Black. *SAMSUNG.* Retrieved April 22nd 2017 from: http://www.samsung.com/nz/consumer/home-appliances/vacuum-cleaners/robot/VCR8980L4K/XSA/

[5] Anonymous (2017). Powerful & Smart Corner Cleaning – VR65710LVMP. *LG Electronics.* Retrieved April 22nd 2017 from: http://www.lg.com/sg/vacuum-cleaners/lg-VR65710LVMP

[6] Anonymous (1997, December). Electrolux unveils prototype for robot vacuum cleaner. *Electrolux Group.* Retrieved April 22nd 2017 from: http://www.electroluxgroup.com/en/electrolux-unveils-prototype-for-robot-vacuum-cleaner-4359/

[7] Anonymous (2007). Electrolux Trilobite. *Robot Buying Guide.* Retrieved April 22nd 2017 from: http://robotbg.com/robots/floor_cleaners/electrolux/trilobite

[8] Anonymous (2017). iRobot Roomba. *iRobot Corporation.* Retrieved April 22nd 2017 from: https://www.irobot.com/For-the-Home/Vacuuming/Roomba.aspx

[9] Anonymous (2017). IMap Water. *Milagrow Business & Knowledge solutions.* Retrieved April 22nd 2017 from: http://milagrowhumantech.com/floor-robots/805-imap-water.html

[10] Anonymous (2012). Cleanmate Q3 LCD. *METAPO Inc.* Retrieved April 22nd 2017 from: http://www.metapo.com/cleanmate3.html

[11] Ashlee Clark Thompson (2016, 17th August). Robomow RS612 Review. *CBS Interactive Inc.* Retrieved April 22nd 2017 from: https://www.cnet.com/products/robomow-rs612/

[12] Anonymous (2016). Rapid XLS. *Aquabot.* Retrieved April 22nd 2017 from: http://www.aquabot.com/residential/aquabot-rapids-xls/

[13] Anonymous (2016). Viu Technology. *Viu Global.* Retrieved April 22nd 2017 from: http://viuglobal.com/how-it-works/

[14] Anonymous (n.d). Skypro Window Cleaning System. *J.Racentein Inc.* Retrieved April 22nd 2017 from: http://www.jracenstein.com/category/000GSP/skypro-high-rise-cleaning-system/

[15] Anonymous (n.d). Seaswarm is a fleet of low-cost oil absorbing robots. *Seaswarm.* Retrieved April 22nd 2017 from: http://senseable.mit.edu/seaswarm/

[16] Toh Ee Ming (2016, 17th October). Govt exploring use of self-driving vehicles for street cleaning. *Mediacorp Press Ltd.* Retrieved April 22nd 2017 from: http://www.todayonline.com/singapore/driverless-vehicles-may-be-used-clean-singapores-streets

[17] Kimberly Amadeo (2016, 9th November). The Great Recession of 2008: Explanation with Dates. *The Balance.* Retrieved April 26th 2017 from: https://www.thebalance.com/the-great-recession-of-2008-explanation-with-dates-4056832

[18] Michael Chui, James Manyika, and Mehdi Miremadi (2016, July). Where machines could replace humans-and where they can't (yet). *Mckinsey & Company- Mckinsey Insights.* Retrieved April 23rd 2017 from: http://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/where-machines-could-replace-humans-and-where-they-cant-yet

[19] Christos Kouroupetroglou (2014). Enhancing the human experience through assistive technologies and E-Accesibility. *Caretta-Net Technologies* . Retrieved April 22nd 2017 from: https://books.google.com.sg/books?id=jReXBQAAQBAJ&pg=PA35&lpg=PA35&dq=cleaning+robot+vs+human&source=bl&ots=jqK27-

Qcrd&sig=JuVtHaiua375N9I5ESj_6Wjokfk&hl=en&sa=X&ved=0ahUKEwjP3p78xbXTAhUaSo8KHTMlAtM4HhDoAQg2MAY#v=onepage&q=clean&f=false

[20] Evan Ackerman (2013, 12th November). iRobot's New Roomba 800 Series Has Better Vacuuming With Less Maintenance. *IEEE Spectrum.* Retrieved April 22nd 2017 from: http://spectrum.ieee.org/automaton/robotics/home-robots/irobot-roomba-800-series-combines-better-vacuuming-with-less-maintenance

[21] Evan Ackerman (2014, 30th April). Avidbots wants to Automate Commercial cleaning with robots. *IEEE Spectrum.* Retrieved April 22nd 2017 from: http://spectrum.ieee.org/automaton/robotics/industrial-robots/avidbots-want-to-industrialize-robot-cleaning

[22] Stephen Nessen (2014, 13th November). A modern John Henry Case: Window Cleaning Robots vs Humans. *WNYC News.* Retrieved April 22nd 2017 from: http://www.wnyc.org/story/modern-john-henry-window-cleaning-robots-vs-humans/

[23] Chris Woodford (2016, 29th November). Roomba Robot Vacuum Cleaners. *Explain That Stuff.* Retrieved April 22nd 2017 from: http://www.explainthatstuff.com/how-roomba-works.html

[24] Becky Mollenkamp (2016, 25th February). Robotic Cleaners give BSC A Leg Up. *Clean link.* Retrieved April 22nd 2017 from: http://www.cleanlink.com/cp/article/Robotic-Cleaners-Give-BSCs-A-Leg-Up--19313

[25] Becky Mollenkamp (2016, 25th February). The advantages of robotic cleaning. *Clean link.* Retrieved April 22nd 2017 from: http://www.cleanlink.com/cp/article/The-Advantages-of-Robotic-Cleaning--19314

[26] Becky Mollenkamp (2016, 25th February). Managing Fears, Expectations, Costs of cleaning robots. *Clean link.* Retrieved April 22nd 2017 from: http://www.cleanlink.com/cp/article/Managing-Fears-Expectations-Costs-Of-Cleaning-Robots--19315

[27] Winda Benedetti (2006, 19th May). Human vs Robots: Sweeping Up is a dirty job, but you don't have to do it. *Hearst Seattle Media.* Retrieved April 23rd 2017 from: http://www.seattlepi.com/news/article/Humans-vs-Robots-Sweeping-up-is-a-dirty-job-1203867.php

[28] Winda Benedetti (1997, 1st December). Electrolux unveils prototype for robotic vacuum cleaner. *Electrolux.* Retrieved April 23rd 2017 from: http://www.electroluxgroup.com/en/electrolux-unveils-prototype-for-robot-vacuum-cleaner-4359/

[29] Seung-Hun Kim , Chi-Won Roh , Sung-Chul Kang and Min-Yong Park (2007, April) Outdoor Navigation of a Mobile Robot Using Differential GPS and Curb Detection. *IEEE International Conference on Robotics and Automation.* Retrieved April 25th 2017 from: http://ieeexplore.ieee.org/abstract/document/4209618/

[30] Ming-Shaung Chang , Jung-Hua Chou & Chun-Mu Wu (2010) Design and Implementation of a Novel Outdoor Road-Cleaning Robot. *Advanced Robotics, 24:1-2, 85-101.* Retrieved April 25th 2017 from: http://dx.doi.org/10.1163/016918609X12586141083777

[31] Hiroshi Yaguchi (1995) Robot introduction to cleaning work in the East Japan Railway Company. *Advanced Robotics, 10:4, 403-414.* Retrieved April 25th 2017 from: http://dx.doi.org/10.1163/156855396X00066

[32] Yuan Fu-cai, Sun Hai-liang , Hu Shi-jian Wang Li-zhu (2012, December) Design of Cleaning Robot for Swimming Pools. *MSIE.* Retrieved April 25th 2017 from: http://ieeexplore.ieee.org/abstract/document/5707629/

[33] Ryosuke Murai, Tatsuo Sakai, Hajime Kawano, Yoshihiko Matsukawa, Yukihiko Kitano, Yukio Honda, and Kenneth C. Campbell (2012, December) A Novel Visible Light Communication System for Enhanced Control of Autonomous Delivery Robots in a Hospital. *IEEE/SICE International Symposium on System Integration (SII),* Retrieved April 25th 2017 from: http://ieeexplore.ieee.org/abstract/document/6427311/citations?tabFilter=papers

[34] Simon Thiel, Dagmar Häbe, Micha Block (2009) Co-operative Robot Teams in a Hospital Environment. *Fraunhofer IAO.* Retrieved April 25th 2017 from: http://ieeexplore.ieee.org/abstract/document/5358271/

[35] John M. Evans (n.d) HelpMate. An Autonomous Mobile Robot Courier for Hospitals. *Transitions Research Corporation Danbury,CT, USA (203) 798-898.,* Retrieved April 25th 2017 from: http://ieeexplore.ieee.org/abstract/document/407629/

[36] Joseph L. Jones (2006, March) Robots at the Tipping Point. *IEEE Robotics & Automation Magazine.* Retrieved April 25th 2017 from: http://ieeexplore.ieee.org/abstract/document/1598056/

[37] Jordi Palacín, José Antonio Salse, Ignasi Valgañón, and Xavi Clua (2004, October) Building a Mobile Robot for a Floor-Cleaning Operation in Domestic Environments. *IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT.* Retrieved April 25th 2017 from: http://ieeexplore.ieee.org/abstract/document/1337300/

[38] Paolo Fiorini, Erwin Prassler (2000) Cleaning and Household Robots: A Technology Survey. *Jet Propulsion Laboratory, California Institute of Technology and FAW—Research Institute for Applied Knowledge Processing,* Retrieved April 25th 2017 from: https://link.springer.com/article/10.1023%2FA%3A1008954632763?LI=true

[39] Christian Hofner, Giinther Schmidt (1995).Path planning and guidance techniques for an autonomous mobile cleaning robot. *Department for Automatic Control Engineering (LSR), Technical University of Munich.* Retrieved April 25th 2017 from: http://www.sciencedirect.com/science/article/pii/092188909400034Y

[40] Julia Fink, Valérie Bauwens, Frédéric Kaplan and Pierre Dillenbourg. (2013, 13th June). Living with a Vacuum Cleaning Robot. Int J Soc Robot (2013) 5:389–408.Retrieved April 25th 2017 from: https://link.springer.com/article/10.1007/s12369-013-0190-2

[41] Tohru Miyake and Hidenori Ishihara (2003). Mechanisms and Basic Properties of Window Cleaning Robot. *IEEUASME lnternational Conference on Advanced Intelligent Mechatronic.*Retrieved April 25th 2017 from: http://ieeexplore.ieee.org/abstract/document/1225543/?part=1

[42] Xian Min Zhang, Nian Feng Wang and Yan Jiang Huang (2016). Mechanisms and Machine Science: Proceedings of ASIAN MMS 2016 & CCMMS 2016. *Springer Nature Singapore Pte Ltd.* Retrieved April 25th2017from:https://books.google.com.sg/books?id=R8yADQAAQBAJ&pg=PA112&lpg=PA112&dq=expensive+sensors+hinders+development+for+robots&source=bl&ots=CnbG8ihFdq&sig=YPYdssWF-nJ2IqltqGAlbLamJX8&hl=en&sa=X&ved=0ahUKEwjB4Pjz8sLTAhWHp48KHQGECaQQ6AEILjAC#v=onepage&q=expensive%20sensors%20hinders%20development%20for%20robots&f=false

[43] “marketsandmarkets (2018, January). Cleaning Robot Market by Type, Product (Floor-cleaning robot, Lawn-cleaning robot, Pool-cleaning robot, Window-cleaning robot), Application (Residential, Commercial, Industrial, Healthcare), and Geography - Global Forecast to 2023. marketsandmarkets.com. Retrieved Feb 01, 2018, from https://www.marketsandmarkets.com/Market-Reports/cleaning-robot-market-22726569.html

[44] Gao, X., Li, K., Wang, Y., Men, G., Zhou, D., & Kikuchi, K. (2007). A floor cleaning robot using Swedish wheels. In IEEE International Conference on Robotics and Biomimetics (ROBIO). IEEE. https://doi.org/10.1109/robio.2007.4522487

[45] Kakudou, T., Watanabe, K., & Nagai, I. (2011, October). Study on mobile mechanism for a stair cleaning robot-Design of translational locomotion mechanism. In Control, Automation and Systems (ICCAS), 11th International Conference on (pp. 1213-1216). IEEE. ISBN: 978-8993215038

[46] Yang, S. X., & Luo, C. (2004). A neural network approach to complete coverage path planning. IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), 34(1), 718–724. https://doi.org/10.1109/tsmcb.2003.811769

[47] Oh, J. S., Choi, Y. H., Park, J. B., & Zheng, Y. F. (2004). Complete coverage navigation of cleaning robots using triangular-cell-based map. IEEE Transactions on Industrial Electronics, 51(3), 718–726. https://doi.org/10.1109/tie.2004.825197

[48] Volos, C. K., Kyprianidis, I. M., & Stouboulos, I. N. (2013). Experimental investigation on coverage performance of a chaotic autonomous mobile robot. Robotics and Autonomous Systems, 61(12), 1314–1322. https://doi.org/10.1016/j.robot.2013.08.004

[49] Vallivaara, I., Haverinen, J., Kemppainen, A., & Roning, J. Magnetic field-based SLAM method for solving the localization problem in mobile robot floor-cleaning task. In 2011 15th International Conference on Advanced Robotics (ICAR). IEEE. https://doi.org/10.1109/icar.2011.6088632

[50] Fink, J., Bauwens, V., Kaplan, F., & Dillenbourg, P. (2013). Living with a Vacuum Cleaning Robot. International Journal of Social Robotics, 5(3), 389–408. https://doi.org/10.1007/s12369-013-0190-2

[51] Sakamoto, D., Honda, K., Inami, M., & Igarashi, T. (2009). Sketch and run: a stroke-based interface for home robots. In Proceedings of the 27th international conference on Human factors in computing systems - CHI 09. ACM Press. https://doi.org/10.1145/1518701.1518733

[52] Chaomin Luo, & Yang, S. X. (n.d.). A real-time cooperative sweeping strategy for multiple cleaning robots. In Proceedings of the IEEE Internatinal Symposium on Intelligent Control, 2002. IEEE. https://doi.org/10.1109/isic.2002.1157841.

[53] Janchiv, A., Batsaikhan, D., hwan Kim, G., & Lee, S. G. (2011, October). Complete coverage path planning for multi-robots based on. In Control, Automation and Systems (ICCAS), 11th International Conference on (pp. 824-827). IEEE. ISBN : 9788993215038

[54] Rhim, S., Ryu, J.-C., Park, K.-H., & Lee, S.-G. Performance evaluation criteria for autonomous cleaning robots. In 2007 International Symposium on Computational Intelligence in Robotics and Automation. IEEE. https://doi.org/10.1109/cira.2007.382916

[55] Wong, S. C., Middleton, L., MacDonald, B. A., & Auckland, N. (2002, November). Performance metrics for robot coverage tasks. In Proceedings of Australasian Conference on Robotics and Automation (Vol. 27, p. 29). ISBN : 0-909040-90-7

[56] Zheng, K., Chen, G., Cui, G., Chen, Y., Wu, F., & Chen, X. (2017). Performance metrics for coverage of cleaning robots with mocap system. In Intelligent Robotics and Applications (pp. 267–274). Springer International Publishing. https://doi.org/10.1007/978-3-319-65298-6_25

[57] Prassler, E., Hägele, M., & Siegwart, R. (2003, January). International contest for cleaning robots: Fun event or a first step towards benchmarking service robots. In Field and Service Robotics (pp. 447-456). Springer Berlin Heidelberg. https://doi.org/10.1007/10991459_43

[58] Tan, N., Rojas, N., Elara, M. R., Kee, V., & Sosa, R. (2015). Nested reconfigurable robots: theory, design, and realization. International Journal of Advanced Robotic Systems, 12(7), 110. https://doi.org/10.5772/60507

[59] Sun, Y., & Ma, S. ePaddle mechanism: Towards the development of a versatile amphibious locomotion mechanism. In 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE. https://doi.org/10.1109/iros.2011.6095135

[60] WEI, G., DAI, J. S., WANG, S., & LUO, H. (2011). Kinematic analysis and prototype of a metamorphic anthropomorphic hand with a reconfigurable palm. International Journal of Humanoid Robotics, 8(3), 459–479. https://doi.org/10.1142/s0219843611002538

[61] Nansai, S., Rojas, N., Elara, M. R., & Sosa, R. Exploration of adaptive gait patterns with a reconfigurable linkage mechanism. In 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE. https://doi.org/10.1109/iros.2013.6697027

[62] Chadil, N., Phadoongsidhi, M., Suwannasit, K., Manoonpong, P., & Laksanacharoen, P. A reconfigurable spherical robot. In 2011 IEEE International Conference on Robotics and Automation. IEEE. https://doi.org/10.1109/icra.2011.5979756

[63] Wei, H., Li, N., Tao, Y., Chen, Y., & Tan, J. (2012). Docking system design and self-assembly control of distributed swarm flying robots. International Journal of Advanced Robotic Systems, 9(5), 186. https://doi.org/10.5772/53233

[64] Wei, H., Cai, Y., Li, H., Li, D., & Wang, T. Sambot: A self-assembly modular robot for swarm robot. In 2010 IEEE International Conference on Robotics and Automation. IEEE. https://doi.org/10.1109/robot.2010.5509214

[65] Mintchev, S., Stefanini, C., Girin, A., Marrazza, S., Orofino, S., Lebastard, V., Boyer, F.An underwater reconfigurable robot with bioinspired electric sense. In 2012 IEEE International Conference on Robotics and Automation. IEEE. https://doi.org/10.1109/icra.2012.6224956

[66] Zhao, J., Cui, X., Zhu, Y., & Tang, S. A new self-reconfigurable modular robotic system UBot: Multi-mode locomotion and self-reconfiguration. In 2011 IEEE International Conference on Robotics and Automation. IEEE. https://doi.org/10.1109/icra.2011.5980293

[67] Kee, V., Rojas, N., Elara, M. R., & Sosa, R. (2014). Hinged-Tetro: A self-reconfigurable module for nested reconfiguration. In 2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics. IEEE. https://doi.org/10.1109/aim.2014.6878302

[68] Golomb, S. W. (1954). Checker Boards and Polyominoes. The american mathematical monthly, 61(10), 675. https://doi.org/10.2307/2307321

[69] Coffin, S., & Slocum, J. (2003). What's New in Polyomino Puzzles and Their Design. In Mathematical Properties of Sequences and Other Combinatorial Structures (pp. 113–119). Springer US. https://doi.org/10.1007/978-1-4615-0304-0_13

[70] Klarner, D. A. (1997, January). Polylominoes. In Handbook of Discrete and Computational Geometry (pp. 225-240). CRC Press, Inc. ISBN : 0849385245

[71] Frederickson, G. N. (2002). Hinged Dissections: Swinging and Twisting. Cambridge University Press. ISBN : 9780521811927

[72] Demaine, E. D., Demaine, M. L., Eppstein, D., Frederickson, G. N., & Friedman, E. (2005). Hinged dissection of polyominoes and polyforms. Computational Geometry, 31(3), 237–262. https://doi.org/10.1016/j.comgeo.2004.12.008

[73] Demaine, E. D., Demaine, M. L., Lindy, J. F., & Souvaine, D. L. (2005). Hinged Dissection of Polypolyhedra. In Lecture Notes in Computer Science (pp. 205–217). Springer Berlin Heidelberg. ISBN: 9783540317111

[74] Sarhangi, R. (2008). Making patterns on the surfaces of swing-hinged dissections. Bridges Leeuwarden Proceedings, 251-258. ISBN : 9780966520194

[75] Prabakaran, V., Elara, M. R., Pathmakumar, T., & Nansai, S. (2017). hTetro: A tetris inspired shape shifting floor cleaning robot. In 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE. https://doi.org/10.1109/icra.2017.79897.

[76] Oksanen, T., Visala, A. (2009). Coverage path planning algorithms for agricultural field machines. Journal of Field Robotics, 26(8), 651-668.

[77] H. Choset, E. Acar, A.A. Rizzi, J. Luntz, Exact cellular de- compositions in terms of critical points of Morse functions, in: Proc. IEEE Int. Conf. Robotics and Automation ICRA'00, Vol. 3, 2000, pp. 2270–2277.

[78] Canny, J. (1988). Constructing roadmaps of semi-algebraic sets I: Completeness. Artificial Intelligence, 37(1-3), 203-222.

[79] S.C. Wong, B.A. MacDonald, A topological coverage al- gorithm for mobile robots, in: Proc. IEEE/RSJ Int. Conf. In- telligent Robots and Systems, IROS 2003, Vol. 2, 2003, pp. 1685–1690

[80] Z.J. Butler, A.A. Rizzi, R.L. Hollis, Contact sensor-based coverage of rectilinear environments, in: Proc. IEEE Int Intel- ligent Control/Intelligent Systems and Semiotics Symposium, 1999, pp. 266–271.

[81] Zelinsky, A., Jarvis, R. A., Byrne, J. C., Yuta, S. (1993, November). Planning paths of complete coverage of an unstruc- tured environment by a mobile robot. In Proceedings of interna- tional conference on advanced robotics (Vol. 13, pp. 533-538).

[82] Gabriely, Y., Rimon, E. (2002, May). „An On-Line Cov- erage Algorithm of Grid Environments by a Mobile Robot ". In 2002 IEEE Conference on Robotics Automation, Washington DC.

[83] Paull, L., Saeedi, S., Li, H., Myers, V. (2010, August). An information gain based adaptive path planning method for an autonomous underwater vehicle using sidescan sonar. In CASE (pp. 835-840).

[84] Yang, S. X., Luo, C. (2004). A neural network approach to complete coverage path planning. IEEE Transactions on Sys- tems, Man, and Cybernetics, Part B (Cybernetics), 34(1), 718- 724.

[85] Xu, L. (2011). Graph planning for environmental cover- age.

[86] Cheng, P., Keller, J., Kumar, V. (2008, September). Time- optimal UAV trajectory planning for 3D urban structure cov- erage. In Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on (pp. 2750-2757). IEEE.

[87] Galceran, E., Carreras, M. (2013, May). Planning cov- erage paths on bathymetric maps for in-detail inspection of the ocean floor. In Robotics and Automation (ICRA), 2013 IEEE International Conference on (pp. 4159-4164). IEEE.

[88] Jin, J., Tang, L. (2011). Coverage path planning on three- dimensional terrain for arable farming. Journal of Field Robotics, 28(3), 424-440.

[89] Galceran, E., Carreras, M. (2013). A survey on coverage path planning for robotics. Robotics and Autonomous systems, 61(12), 1258-1276.

[90] Tan, N., Mohan, R.E. and Elangovan, K. A bio-inspired reconfigurable robot. In Advances in Reconfigurable Mecha- nisms and Robots II (pp. 483-493), 2016, Springer, Cham.

[91] Nansai, S., Rojas, N., Elara, M.R. and Sosa, R. Exploration of adaptive gait patterns with a reconfigurable linkage mech- anism. In Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on, 2013, November pp. 4661-4668. IEEE.

[92] Kee, V., Rojas, N., Elara, M.R. and Sosa, R. Hinged-Tetro: A self-reconfigurable module for nested reconfiguration. In Ad- vanced Intelligent Mechatronics (AIM), IEEE/ASME Interna- tional Conference on, 2014, July (pp. 1539-1546). IEEE.

[93] Veerajagadheswar, P., Elara, M. R., Pathmakumar, T., Ayyalusami, V. (2018). A Tiling-Theoretic Approach to Effi- cient Area Coverage in a Tetris-Inspired Floor Cleaning Robot. IEEE Access, 6, 35260-35271.

[94] Hameed, I. A., Bochtis, D. D., Sorensen, C. G. (2011). Driving angle and track sequence optimization for operational path planning using genetic algorithms. Applied Engineering in Agriculture, 27(6), 1077-1086.

[95] Hsu, P. M., Lin, C. L., Yang, M. Y. (2014). On the complete coverage path planning for mobile robots. Journal of In- telligent Robotic Systems, 74(3-4), 945-963.

[96] Gonzalez, E., Alarcon, M., Aristizabal, P., Parra, C. (2003, October). BSA: A coverage algorithm. In Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on (Vol. 2, pp. 1679-1684). IEEE.

[97] Gonzalez, E., Alvarez, O., Diaz, Y., Parra, C., Bustacara, C. (2005, April). BSA: a complete coverage algorithm. In Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on (pp. 2040-2044). IEEE.

[98] Golomb, S. W. (1954). Checker Boards and Polyominoes. The american mathematical monthly, 61(10), 675.

[99] Frederickson, G. N. (2002). Hinged Dissections: Swinging and Twisting. Cambridge University Press.

[100] Jho, C.W. and Lee, W.H. Video Puzzle Game Application of Polyomino Re-tiling. In Embedded and Multimedia Com- puting Technology and Service (pp. 363-369) , 2012. Springer, Dordrecht.

[101] Lo, K.Y., Fu, C.W. and Li, H. 3D polyomino puzzle. In ACM Transactions on Graphics (TOG) (Vol. 28, No. 5, p. 157), 2009, December. ACM.

[102] Lester, C. Tiling with T and skew tetrominoes. Querqus: Linfield Journal of Under, 1(1), p.3. October 2012.

[103] Nitica, V. The tilings of deficient squares by ribbon L- tetrominoes are diagonally cracked. arXiv preprint arXiv:1701.00419, 2017.

[104] gfredericks.com "A Polyomino Tiling Algorithm", Aug 2018. [Online]. Available: https://gfredericks.com/gfrlog/99

[105] Choset, H. M., Hutchinson, S., Lynch, K. M., Kantor, G., Burgard, W., Kavraki, L. E., Thrun, S. (2005). Principles of robot motion: theory, algorithms, and implementation. MIT press.

[106] Englot, B., Hover, F. S. (2012, June). Sampling-Based Coverage Path Planning for Inspection of Complex Structures. In Icaps.

[107] Golomb, S.W. Polyominoes: Puzzles, Patterns, Problems, and Packings; Princeton University Press: Princeton, NJ, USA, 1996; ISBN 9780691024448.

[108] Chu, I.-P.; Johnsonbaugh, R. Tiling deficient boards with trominoes. Math. Mag. 1986, 59, 34–40, doi:10.2307/2690016.

[109] Akagi, J.T.; Gaona, C.F.; Mendoza, F.; Villagra, M. Hard and Easy Instances of L-Tromino Tilings. arXiv 2017, arXiv:1710.04640.