Design and control of a robotic system based on mobile robots and manipulator arms for picking in logistics warehouses

Muhammad Aqib Khan

Normandie Université

THESE

Pour obtenir le diplôme de doctorat

Spécialité Automatique, Signal, Productique, Robotique

Préparée au sein de « Université Le Havre Normandie »

Conception et commande d'un système robotisé à base de robots mobiles et bras manipulateurs pour le picking en entrepôt logistique

Présentée et soutenue par
Aqib Khan

Thèse soutenue publiquement le 10 Décembre 2020 devant le jury composé de		
M. Bernard Bayle	Professeur des Universités / Télécom Physique Strasbourg, Icube Université de Strasbourg	Rapporteur
M. Pierre Blazevic	Professeur des Universités / ISTY, LISV, Université de Versailles Saint-Quentin-en-Yvelines	Rapporteur et Président du Jury
Mme Sophie Sakka	Maître de conférences-HDR / LS2N Ecole Centrale de Nantes	Examinatrice
M. Jean-François Brethé	Professeur des Universités / GREAH Université Le Havre Normandie	Directeur de thèse
M. François Guérin	Maître de conférences-HDR / GREAH Université Le Havre Normandie	Codirecteur de thèse

Thèse dirigée par Jean-François Brethé et François Guérin, **Laboratoire GREAH (EA 3220)**

UNIVERSITÉ LE HAVRE

ED PSIME

GREAH

Every step in the right direction, no matter how big or small, is a step towards progress .....

Acknowledgements

This doctoral thesis is a collaboration between Groupe de Recherche en Électrotechnique et Automatique du Havre (GREAH) of University of Le Havre and its industrial partner FM Logistic France. For giving me the opportunity to work on this industrial project and acquire the benefits of learning in terms of applied research, I have to thank FM logistic France for taking this initiative. I want to take this opportunity to express my gratitude to my manager Patrick Bellart (FM) for his support all the way till the end for providing me everything I required to finish the project. I truly appreciate his quality of having the full view and scope of the project in terms of understanding the challenges and constraints within and his valuation of achievements in presence of limitations was a true source of motivation.

I would like to thank my thesis director Pr. Jean-François Brethé for his supervision on this thesis, and I highly appreciate his flexibility for giving me the time and space to complete my work and achieve my desired goals. Our cordial relationship gave me a chance to benefit from his knowledge and wisdom he shared during his supervision, pertaining to academia, research, industry and personal matters. I owe a special thanks to my co-supervisor Dr. François Guérin for his relentless support and guidance throughout this thesis. Particularly with his experience in mobile robotics and control, he has been a mentor and a source of new ideas. His encouragement and supervision in the right direction, complemented with a correct insight of the problem and its solution at hand, and the camaraderie we built over the years led me to a successful completion of the project in all respects. I would also like to thank Mr. Hervé Pelvillain (IUT Le Havre), for his help in the development of the electronic cards and Mr. Philippe Saint-Martin (IUT Le Havre) for the fabrication of the gripper concerning the overall development of the prototype of the mobile manipulator.

I also want to thank the respectable jury members, Professor Bernard Bayle (ICube, Université de Strasbourg), Professor Pierre Blazevic (ISTY, LISV Université de Versailles), and Professor Sophie Sakka (LS2N, Ecole Centrale de Nantes), for reviewing my work and giving back an in-depth critical analysis to highlight the significant areas of this thesis and its presentation for the defense. I would like to express my special gratitude to my friend and my colleague Dr. Kharidine Benali for his joyful company, friendly advice and moral support in the tough and challenging times we faced together. I also want to thank my friends and current PhD students (Mark Bastourous, Wafae Sebbata, Ali Sayah) in the lab, without whom the work atmosphere would not have been more pleasurable.

I also dedicate this thesis and my success to my very close family members, whom I could not reach in time to see them one last time, before they left for their heavenly abode. In the end I would like to express all my gratitude to the love and prayers of my parents and my spiritual guide “Baba G”. Their prayers and guidance infused with love, and their concern for my well-being throughout my life enabled me to overcome any challenge, to achieve and become what I am today.

Abstract

Logistics is the backbone of industrial and commercial trading. It involves the storage and displacement of goods from one point to the other. Goods are stored in warehouses and shipped to retailers in pallets. Pallets are produced on a customer's order. Order picking is a cumbersome and fatigue induced process performed by humans. Fatigue is induced due to walking long distances, lifting heavy loads and acute picking postures. This results in musculoskeletal disorders and poor performance of the workers, thus decreasing the productivity and inducing delays in the supply chain. The demand of productivity has been further alleviated by the e-commerce effect. To mitigate this problem and overcome the constraints of production, automation is actively being employed in warehouses. This consists the use of various technologies and engineered infrastructures. Commissioning of huge infrastructures is complex and inflexible in terms of replaceability. Flexibility is introduced by commissioning robots for process automation. These robots consist of autonomous ground vehicles for transporting freight and static manipulators for pick and place. A static robot has limited workspace and the capability of a manipulator is significantly enhanced by adding a mobile base. Hence mobile manipulation is being exploited for pick & place and pallet production. A lot of effort and research is being dedicated in this domain and this thesis presents a first attempt to achieve autonomous palletization using mobile manipulation.

To acquire palletization by mobile manipulation requires the identification of the functional blocks, which are necessary to conceive a framework to achieve this task. A thorough state of the art has been prepared in this thesis corresponding to each element of the global framework. To realize the proof of concept, a prototype has been developed by leveraging existing technologies, which involves the integration of a mobile base with manipulator and development of the grasping system with a gripping element. For each functional block of the global framework, control execution strategies have been developed and tested in industrial environment. Specifically, localization is acquired by augmenting the existing environment with synthetic landmarks, a motion planning and control strategy is employed for global navigation and a rack tracking motion control has been developed for navigation inside the racks. To combine and execute all the elements without deadlocks a coordination framework is used as a global supervisor. The path planner for global navigation is based on the shortest distance between two points, and rack tracking is developed by applying the conventional Hough transform (mainly used with vision) to the lidar data and using the output in a nonlinear controller, while the motion planner for manipulation is based on linear trajectories.

The framework for supervisory control of the functional blocks is based on discrete event systems topology and state machines corresponding to each block have been modelized using Petri nets. Finally, the framework has been tested for a complete picking task on the mobile manipulator to validate the selection of strategies and performance of each functional element. The successful demonstration has been concluded as a first step towards the evolution of autonomous palletization.

Contents

Nomenclature

This nomenclature references the principal variables and abbreviations used in this thesis.

X_I	X coordinate of the inertial reference frame
Y_I	Y coordinate of the inertial reference frame
X_R	X coordinate of robot frame
Y_R	Y coordinate of robot frame
ξ_I	Pose of robot in inertial (global)
$\dot{\xi}_I$	Twist of robot in inertial frame
$\dot{\xi}_R$	Twist of robot in own frame
θ	Angle between the local and global frame
$R(\theta)$	Rotation matrix
F_i	Wheel frame
F_R	Robot frame
f_i	Roller frame
α_i	Angle between robot frame and wheel frame
l_i	Relative offset of wheel frame with respect to robot frame
δ_i	Rotation angle between wheel frame and robot frame
γ_i	Rotation angle between roller frame and wheel frame
φ_i	Rotation angle of the wheel
$x_{r,i}$	X coordinate of the roller frame 'i'
$y_{r,i}$	Y coordinate of the roller frame 'i'
$\dot{x}_i$	Wheel velocity along horizontal axis
$y_{r,i}$	Wheel velocity along vertical axis
J	Robot jacobian
T_i^{i-1}	Transformation matrix
$\vec{F}_{TH}$	Theoretical holding force
μ	Coefficient of friction
F_{s_i}	Force on each suction cup
P	Pressure for each suction cup

P_{Abs}	Absolute pressure
P_{atmos}	Atmospheric pressure
$P_{negative}$	Negative pressure or vacuum
$I(x, y)$	Map matrix
$d(x, y)$	Distance matrix
$C(x, y)$	Current cell matrix (mask)
X_P	Waypoint 'x' in the global coordinate frame
Y_p	Waypoint 'y' in the global coordinate frame
X_{WR}	Waypoint 'x' in the robot frame
Y_{WR}	Waypoint 'y' in the robot frame
d	Distance of the robot center from a waypoint
L	Length of the robot
e_{XR}	Error along horizontal distance
e_{Y_R}	Error along vertical distance
u_n	Linear velocity control input
T	Sampling time
τ_u	Time constant
U_M	Saturation velocity linear
ω_n	Rotational velocity control input
K_Y	Proportional gain
Ge	Distance error
m	Slope of line
c	Y intercept of line
ρ	Distance rho in polar form
$X_{lid(i)}$	X coordinate of the 'ith' lidar point
$Y_{lid(i)}$	Y coordinate of the 'ith' lidar point
d_i	Radial distance of the 'ith' lidar point
θ_i	Radial angle for the 'ith' lidar point
w	Width of the range image
h	Height of the range image
X_A	Rack 'x' coordinate
Y_A	Rack 'y' coordiante

w_R	Rack width
$u_{\max}$	Maximum linear velocity
$r_{\max}$	Maximum rotational velocity
$v_{\max}$	Maximum transverse velocity
λ	Sigmoid tuning parameter
k_u	Gain for linear control input
k_v	Gain for transverse control input
k_r	Gain for rotational control input
U_x	Robot linear velocity
V_y	Robot transverse velocity
R_z	Robot rotational velocity
θ_{ref}	Angle with respect to the rack
d_{ref}	Reference distance to move inside rack
M	Marking of the Petri net
$I(t,p)$	Set of input places
$O(t,p)$	Set of output places
a_{ij}	Incidence matrix
u_k	The 'k_{th}' firing or the control vector
ΔM	Marking Evolution

List of Figures

Chapter 1

Introduction

Modern day logistic is an evolutionary progression spanned across the historic timeline. The logistics has evolved from simple labor-intensive processes of pick, place and transport to the present-day engineering and managing of extremely complex supply chain networks spread across the globe. Traditional logistics in the early times made use of material transportation on mules and camel carts, but with the course of time the logistic industry has developed with the evolution of many innovations in technology and industrialization. Logistics picked up its pace after the mechanization of industry by the invention of steam engine, whereby the steam ships and the railroads streamlined the process flow, while the critical advances in automotive such as the invention of internal combustion engine and pneumatic tires played a crucial role in the progression of logistics. At the same time the advances in aviation and air transportation played a significant role in the speeding up of dispatch and delivery. Conclusively, the railroad, steamship, aircraft and the telegraph had a profound impact on the logistics during the last half of the 19th century. The emanation of conflicts and ensuing events over the course of history, significantly in the last century resulted in the realization of importance of logistics. More specifically the potential of logistics was realized in World War II, when man and machinery had to be moved in large numbers over large distances across the globe. From there onwards what followed in the development and advancement of logistics is consequently prevalent today.

Taking it a step further, the revolution of digital technology has given the power of managing information leading to the full control of situations to achieve operational excellence. The emergence of computing has completely changed the planning paradigm of logistic processes. Today the modern logistics processes are not only bound to merely transportation of material in conflicts, but have diffused to the industrial domain all the way extending to warehousing, material handling and freight transportation. The technological advances have provided tremendous value to the traditional supply chain in logistic areas, such as warehousing and distribution, transportation and manufacturing logistics. In the context of improvement, the warehouse automation has immensely enhanced warehouse operations producing a huge impact on improving efficiency of logistics. The automation varies from use of technologies for inventory management to autonomous shuttles for material transportation, robotic pickers to pallet producers, and then extending all the way to warehouse management

(WMS) and Enterprise Resource Planning (ERP) systems providing better advanced planning and scheduling for logistic processes.

1.1 Logistics Processes

Logistics involves the organization and coordinated process management of planning and control of efficient material handling from the point of distribution to the point of demand. The process framework is realized by integrating information technology, transportation, inventory management and storage facilities. Logistics as a whole, builds upon two primary processes. The movement of materials from the suppliers to utilizers or storage facilities such as warehouses is called inbound logistics, whereas movement of products from the stock to the end user is termed as outbound logistics. The management process of logistics consists of three major components, i.e. order processing, inventory management, and freight transportation. The demand of the customer is accepted in the form of an order. The order is processed and the availability of the requested product is verified by consulting the inventory, resources are allocated and job assignment is planned for the management of the order. The inventory management helps to keep an updated record of the available stocks in production facilities. This record is maintained virtually to track the stock limit in real time against the influx of the demands. The physical management of the product inventory is done by the storage and material handling process. Further freight transportation provides a quick delivery of the materials from the point of production to a distant point of consumption. This freight is usually transported by land, sea and air. The material is moved from the source to the destination by the alternate combination of the three modes of transportation.

1.2 Warehouses

Considering the significance of the operations of logistics, the storage facilities make up the core components in the process flow. Both inbound and outbound logistics make use of the warehouses for goods retrieval and material storage. The typical operations of warehouse involve reception of goods, their sorting, inventory-based storage, picking on order, repacking and then shipping to destination. The pick on order is based on the demand of a certain client. The demand is qualified by verifying the products from the inventory to fulfill the customer's order. The products are then picked from specified zones in the warehouse and stacked together to make up a pallet. The pallets are then shipped out to the destination.

1.2.1 Challenges

The order preparation or picking of products is a complex process. The products are stored in racks or material storing shelves inside the warehouses. The picking is performed

manually by human workers. Each item in the order-based pallet is picked from a specific rack. An employee preparing the order has to walk [1] around the warehouse to gather all the items for a specific order. This process is time consuming and fatigue induced at the same time. The movement around the warehouse consumes a lot of time since the employee has to walk long distances to get to specific locations. In addition, the retrieval of items from the shelves involves repetitive material handling activities of continuous handling or lifting of heavy loads with acute picking postures requiring twisting, bending, kneeling or stretching thus exposing the workers to musculoskeletal [2] disorders. The musculoskeletal disorders lead to a poor health and delayed recovery resulting in a poor performance of the workers leading to a poor management of order preparation and delays of deliveries on time. The arduous work leads to fatigue resulting in the increase of sick leaves. The rise in the number of sick workers results in the consumption of extra resources spent by the enterprises on the healthcare of their employees.

Figure 1.1 : An employee preparing the order[1]

The digital revolution has completely transformed the modern world in the last half century. Whereas the digital transformation has upgraded and enhanced all the domains of life, the most significant advancement has taken place in the field of information technology. Over the last three decades, the introduction of the internet has played a huge impact on the economic growth and prosperity by connecting people, businesses and enterprises. The rapid globalization of industry and supply chain is due to advancement in information technology in which the internet has acted as a bridge to connect and foster the communication channels for a smooth functioning and steady growth. The internet has transformed the structure of

1 http://www.fmlogistic.com/Medias/Mediatheque

value chains of all the sectors and types of companies. The exponential growth in the e-commerce business especially, is due to the empowerment provided by the internet to the consumers offering a direct route to reach the goal while cutting down the time.

The Amazon effect has played a key role in alleviating the situation of e-commerce trading, directly impacting the logistic business not only for Amazon itself but also for other key competing enterprises. The Amazon effect has changed the traditional shopping pattern, customer expectation and competitive trading landscape in the modern world business market. Following the lead of Amazon, the retailers like 'eBay' and 'Alibaba' have also added to the competition making the e-commerce universe more competitive and raised the bar to the next level for the proficiency of the supply chain management. Alibaba and eBay perform as middle man between sellers and buyers facilitating the sales through a network of websites, whereas Amazon sells goods directly while managing a network of large warehouses and complex operations of logistics, to deliver the products to the doorstep of the customer. The strategy of these retail giants such as Amazon is not only limited to trade productivity, but also they are utilizing Artificial Intelligence (AI) to increase the product sales by aggregating trading data and analyze it using machine learning to forecast demand, extract purchase patterns to improve customer's buying and give recommendations, detect marketplace trends, and improve shipping methodologies based on customer's feedback.

The growth in the e-commerce business has also led to one of the major challenges facing the logistic industry today. The rise of online sales has become inexorable. Due to increase in sales online the objective of meeting supply to demand has increased two folds, since a greater number of products have to be supplied that customers demand. This increase in sales is directly proportional to the demand of supply of required products. To manage the demand a greater number of resources are required in the given time, meaning to supply the products more skilled labor is required to handle and manage the supply chain. The availability of good skilled labor has declined in the last decade due to various factors, resulting in a big gap between the availability of skilled workers and logistic operations to fulfill the ever-growing demand of online retail business. While the giants like Amazon and Alibaba continue to thrive and advance the e-commerce industry, the gap between the workers required to meet the operational demand and deliver is rapidly widening.

1.2.2 Productivity

Enterprises like Amazon are managing large warehouses for the management and distribution of products, but the online business is just one example. In traditional trading for retail, the warehouse operations are primarily conceived on the nature of order received from

the retail client. The orders are prepared and packed in pallets and dispatched in bulk. The delivery of the orders has to be done on bi-weekly and monthly basis. To make the delivery proficient, there is always the requirement to increase the productivity in the warehouses. This is achieved by planning and organizing all the operations of storage, sorting, packing and shipping to save time and space. This is usually done by the employment of a warehouse management system. The objective is to reduce the cycle time of order preparation as much as possible and increase the storage space in the warehouse to manage more stock.

The e-commerce revolution has changed the traditional methodology of sorting, storage, picking, packing and shipping. Now the same course of action has to be used to pick and ship small volumes instead of bulk. Instead of packing in pallets, the packages have to be packed individually requiring care and extra handling for safety. The items have to delivered in 24 to 48 hours' time to the client in contrast to the delivery after one week. There is a continuous race with time since the e-commerce demand has put a pressure on the warehouse productivity, as now the warehouses have to keep up with the influx of the demands enforced by the online customers. As the order volume increases, the on-hand stock also has to increase in proportion, and warehouse efficiency to deliver has to be improved to avoid complacency. In addition, the customers require a forecast and an undamaged delivery to the doorstep. This requires the product to be tracked in real time, which in turn requires the IT framework to be enhanced to provide an efficient tracking capability.

1.2.3 Automation

Automation is the application of technology to accomplish a process or procedure without human assistance. It facilitates the minimization of manual labor resulting in improved ergonomics, reduction of labor costs and travel time, and preservation of space. The technology is used to automate a process or repetitive tasks in the manufacturing plants and this application is more cost effective in the long term since it reduces the number of required people to do the same job, while at the same time providing a faster and error free service.

Considering the situation of present-day logistic industry, automation is being employed to mitigate the challenges and constraints of production in the warehouses. Warehouse automation consists of the use of technologies which guarantee increase of productivity. This automation is built upon mechanical systems, actuators and control and the mechanization consists of automated storage and retrieval systems (AS/RS), automated guided vehicles (AGV), industrial robots for palletization, sorting systems for inventory, picking systems based on voice and related technologies. Along with the mechanization, computerized

systems consisting of warehouse management software solutions, are used to manage and control the physical tracking of the flow of goods in the warehouse.

1.2.3.1 Types

Numerous types of technologies are being used to automate the processes inside the warehouses. The e-commerce is further augmenting the use of automation to keep up with the pace of production. An overview of the basic automation existing in almost every warehouse of large and medium sized enterprises in the logistic industry is outlined below.

1.2.3.1.1 Automated Storage and retrieval systems

These systems consist of automatic bringing out of goods and placing back into storage. There are basic two types. They can either perform as "Unit-load" automated storage and handle large loads of goods such as full pallets, or operate as "Mini-load" systems handling smaller loads of goods.

A. Unit-Load AS/RS systems

- Handle larger loads of material of 1000 to 5000 pounds, typically pallets.
- Consist of a fixed-aisle unit-load crane (fixed to a single aisle) traveling between aisles to retrieve products
- Consists of moveable-aisle unit-load crane (not fixed to a single aisle) moving along the aisles to retrieve products.

B. Mini-Load AS/RS Systems

- Typically handle smaller loads such as totes, trays and cartons. Also called as "tote-stacking" or "case-handling" systems.
- Well suited for operations requiring storage location for a large number of SKUs lacking floor space.
- Common types are mini-load AS/RS cranes and shuttles, carousel-based AS/RS, Vertical Lift Modules and Goods-to-Person (G2P).

In both the unit load and mini-load handling systems, a crane travels between narrow rows of goods while moving either pallets or totes.

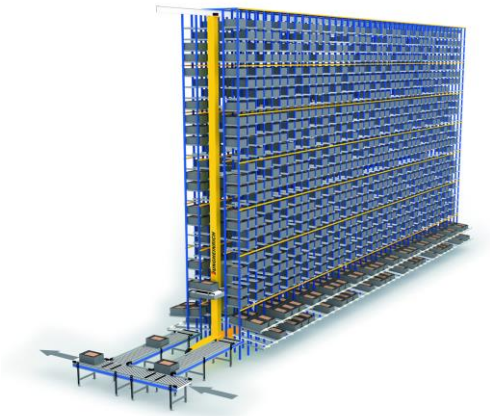


Figure 1.2 : Automated Storage and Retrieval Systems[2]

1.2.3.1.2 Sorting Systems

The sorting systems consist mostly of conveyors performing the sortation of products. The items are identified on the conveyors and then diverted to their specific destination. The configuration and layout of the sorting systems depends upon the activity to perform. Line or linear sorters move the products in a straight line along their length. These sorters are either belt or chain driven and the package diverting mechanism is integrated onto the sorters. The other type of sorters known as loop sorter or circle sorter consists of a series of cells linked together on the track of the conveyor system. The products are identified by scanners and diverted to respective locations by diverting mechanisms. The common examples of circle sorters are Sliding shoe, Tilt tray and Cross-belt sorters. The tilt tray sorter consists of trays mounted onto carts on the conveyor running in a continuous loop. The items are inducted either manually or automatically via the induction stations at multiple locations throughout the loop.

Figure 1.3 : Automatic Sorters[3]

2 https://www.jungheinrich.co.th/products/racking-and-warehouse-equipment/small-parts-storage/automatic-small-parts-warehouse-small-parts-racking/

1.2.3.1.3 Automated Tracking Systems

The automated tracking systems provide a real time information of the availability, record management and flow control of goods. This tracking is acquired by the use of inventory management systems, scanners and perceptual guidance systems for picking.

1.2.3.1.3.1 Inventory management Systems

It is always required to have a real time information of overstocking and understocking in distribution centers and acquire a knowledge of available capacity in the warehouse. This is achieved by maintaining a record of the total inventory. The inventory management systems (IMS) consist of the equipment and hardware to automate the documentation process of the goods inside the warehouse. The material used to record and process the identification of the products consists of barcode labels, rack labels and signs, Radio Frequency Identification (RFID) tags and Quick Response (QR) codes. The hardware used to scan these materials mostly consists of hand held digital scanners embedded with supported software. The IMS is integrated with the Warehouse Management Systems (WMS) to track the inventory in real time to ensure its matching with the orders placed. The IMS keeps a record of the number and characteristics of Stock Keeping Units (SKUs) and the level of the current inventory in the warehouse indicating the availability of space for incoming products. To update the inventory and have a current status, the workers use the handheld scanners to scan the labels of the products during the storing and shipping procedures. Hence the record maintains a real time status of the products and goods coming in and going out of the warehouse.

1.2.3.1.4 Pick to light

Pick to light systems are supervisory systems for order fulfillment. These systems aid the workers by guiding them for manual pick and record keeping for shipment. These systems consist of alphanumeric displays to guide the workers to the right storage location and indicate the number of items to be picked. The workers scan a barcode and LED displays illuminate to guide them to the next pick location. The operator after placing the picked items in the container presses a verification button and the display continues to illuminate to guide the workers to the next pick location. The same systems can be used in reverse as in put to light for storing items. The pick to light system is employed to reduce the time to walk to the pick location and time taken to read paper-based records by the workers. The pick to light system is installed with the software which is able to integrate with the WMS and supply chain management. These systems improve the picking productivity by increasing the picking accuracy by avoiding identification errors and efficient record management of the products retrieved or stored.

Figure 1.4 : Pick to Light Scanners[4]

1.2.3.1.5 Robot arms and AGVs

Conventional industrial robots are being used for picking and palletizing tasks in the warehouses. These robots along with the sorting systems perform the job of traditional picking done mainly by human workers. These units are static and are installed at the picking locations, next to a conveyor to pick up individual packages and build up a pallet. The picking robot is generally a robotic arm equipped with vision and associated sensors to determine the structure and configuration of the box or package to grasp it. These flexible arms are capable of performing multiple tasks, including individual picking, loading and unloading cartons and producing pallets.

Figure 1.5 : Robot[5] Arm and Automated Guided Vehicle[6]

3 https://www.okurayusoki.co.jp/eng/products/plant/sorting/crossbelt/index.html

4 http://www.fmlogistic.com/Medias/Mediatheque

Likewise, the Automated Guided Vehicles (AGVs) are used to transport materials from one location to the other. These AGVs are primarily performing the loading and unloading of pallets, boxes and containers. The AGVs navigate autonomously in the warehouse, following a digital path without the intervention of a human, or either follow a worker during material loading and unloading, all the while operating in a fleet to speed up the material transportation within the warehouse.

1.2.3.2 Advantages

Warehouse automation extensively benefits the productivity of the warehouse and logistic operations, producing a significant impact on the enterprise's gains and profits. For the improvement of business, the benefits in the long term are well accounted for, but the significant primary ones on the macro level are highlighted

1.2.3.2.1 Reduced costs

By using machines, the need for manual labor can drastically be reduced cutting down in the labor costs. The machines keep working all the time and replace the extra workers required to work overtime. Especially during the holiday seasons, the retailers are faced with overtimes due to higher demand load and extra money has to be paid to the workers for over times, whereas the machines can be run for a long time with no incurring costs. Thus, utilizing automation is much cheaper in the long run as compared to spending on manual labor from time to time when demand is invoked.

1.2.3.2.2 Increased speeds

Product retrieval and order fulfillment is also sped up by automation. Machines do the job of handling, quick sorting and storage of goods, and this has a direct impact on the inventory management system. Faster storage and retrieval results in faster update of the changes in the inventory and a real time tracking of inventory status, as contrast to the manual recording of the items by workers.

1.2.3.2.3 Maximization of space

Space is a run time necessity during cyclic operations in the warehouses. Automation employs the use of machines, either robots or AS/RS for product handling in terms of retrieval and storage. The need to walk to an aisle to retrieve an item is eliminated and traditional layout of wide racks with pallets for storage, is replaced by moveable shelves carrying goods, moved around by carrier robots. This is activity is based on "Goods to Person" concept. These shelves are not very high taking up less space than a pallet and require less room to maneuver by the robots. The height is kept small to keep the balance during maneuvering. These product

5 https://sealing-system.dk/en/packaging-solutions/kuka-kr-120-pa/

6 http://www.directindustry.com/prod/jungheinrich/product-1078-1668375.html

containing shelves can be installed in small aisles configurations creating more room for storage. The example has been set by Amazon and Alibaba, efficiently eliminating the need for wider aisles for people and robots to move through and producing more space for additional storage.

1.2.3.2.4 Enhanced safety

The employment of AS/RS brings the products to the workers rather than the workers going to the products to pick. This directly reduces the footprint of movement of workers and related equipment traffic through the facility. Also, the safety of the products being handled by the AS/RS is less compromised as compared to manual picking and handling from one point to the other.

1.2.3.3 Problems

Even though automation facilitates higher production rates and increased productivity, yet it does not come with the mode of simplification. Sophisticating the process does not make the system simplified as a whole. One of the significant caveats of employing automation is the installation of huge infrastructure in the form of AS/RS and conveyors lines. The installation of heavy conveyors and retrieval systems affects the global layout of the warehouse. New buffer and safety zones have to be introduced with respect to the operation of equipment and infrastructure in use. These safety zones ensure the safety of the workers as well as that of the equipment. The layout and orientation of work methodology changes which requires the training of workers required to handle the automation equipment. This is also accompanied with the maintenance costs incurred to maintain the infrastructure at the optimum level. The maintenance levels for the employed automation in the current times have tripled with the exponential increase of orders in e-commerce trading.

Another bigger problem is that the deployment of infrastructure comes at the cost of a huge initial capital, even though the Return of Investment (ROI) is granted in the long run. An infrastructure once deployed cannot be removed restricting the flexibility to change the layout when required. Another major fact in the logistic industry is that the client contracts are based on short times. A typical contract lasts from six months to at most three years only. Therefore, installing a heavy capitalized infrastructure for a short-term ROI based on limited time contract is a risky option. The deployed automation must have the scalability and flexibility to change and move to a new application and location based on the fluctuation of the client's needs.

1.3 Flexibility with robotization

To tackle the previous mentioned challenges of picking, increasing productivity while meeting the e-commerce demand, automation has solved the problem to a greater extent, but at the cost of a complex system layout and functioning. It is not possible to employ full automation to a business case since it implies building a new infrastructure loaded with automated technology from scratch. Instead the preferred solution is to introduce flexibility in the existing automation which is achievable by the use of robotics technology. Industrial robots were already introduced in the manufacturing industry five decades ago. Till date they are performing various tasks of additive manufacturing such as assembling, welding, painting, gluing, and subtractive manufacturing such as milling, cutting, grinding, polishing along with miscellaneous operations of machine tending, picking, packing and inspection. Realizing their potential for the outcome, conventional industrial robots quickly migrated to logistics. In warehouses these robots are performing the operations of pick, place, pack and palletize. However, these robots are static and can maneuver to operate, but cannot move to another location for a different operation.

To add the capability of transporting the materials to different locations, AGV's were also introduced in the warehouses. The previous generation of AGV's were able to navigate by following wired or magnetic tape embedded in the floor. With the evolution of time the AGVs got more advanced and the present generation of AGVs are equipped with sophisticated sensors and Artificial Intelligence giving them the capability to plan their own routes and avoid obstacles while transporting materials efficiently. These autonomous mobile robots are either working as shuttles, transporting materials in carts executing the "goods to man" framework, or as autonomous forklifts transporting heavy loads from one place to the other. The next step is to enhance the flexibility further and combine the static element of pick and place and the mobile element of transportation to achieve mobile manipulation. Then use these mobile units to pick items and deliver where necessary. This approach will necessarily cut down the commissioning of automated sorters and long conveyor lines, and AS/RS consisting of huge infrastructure of racks and cranes for product extraction. With the recent change in trends and evolution of collaborative robots "Cobots" on the factory floors, the goal is to have a cobotic solution to work alongside human workers while aiding them in orders fulfillment.

1.4 European Projects

Combining the elements of mobility and manipulation increases the scope of application and versatility. Realizing the potential of such an application especially in the

logistic industry, a significant amount of importance is being given to direct the research and development in this direction by the scientific community and industrial enterprises likewise. In this context it is worthwhile to give a perspective of the work conducted by European conglomerates in the context of mobile manipulation in the last five years. These groups consist of large enterprises braided together with research laboratories, to advance the work in the context of industrial application by leveraging the potential of academia. All these projects are part of European Union's Horizon 2020 Research and Innovation program. A comprehensive overview of these groups is given below

1.4.1 ILIAD

ILIAD[7] stands for Intra-Logistics with Integrated Automatic Deployment, of safe and scalable fleets of robots in shared spaces. The project commenced at the start of 2017 and will last three years with a budget of almost 7 million Euros and consists of four universities and five industrial partners. The goal of the project is the automation of the Intralogistics services of the food distribution sector. This automation consists of utilizing flexible robotics solution which are able to integrate with the existing warehouse infrastructure and facilities. The work is particularly focused on the use case of autonomous robots for logistic automation and the research on the associated aspects that are influenced by the autonomy of the robots. Specific features of the work being done in ILIAD focus on:

- the autonomous deployment of robot fleets in the infrastructure free environment
- continuous long-term operation of the fleet and its evolution and adaptation to changing warehouse conditions
- development of fleet management framework with robust motion planning and coordination
- development of a framework of human awareness for motion prediction

and learning the human behavior models over scales of time and generate robot motions based on the learning acquired from behavior models that is to predict a more conformative motion of the robot. The idea is to have the robots an understanding of the social context they are operating within and how they should behave and evolve when the social context changes. The focus is also on the development of novel end effectors and manipulators for utilization in mobile manipulation to handle, pack and unpack pallets. This approach is augmented with the development of perceptive skills to localize the goods to pick them. Further the work in the context of Human Robot Interaction (HRI) is focused on the safety of humans in the shared environment with the robots. This is achieved by designing safe robot motion control strategies based on the data acquired by studying the basic human injury biomechanics during

collisions. The work focus is primarily on the building and improving the safety standards of the shared environment with the robots. In a nutshell, ILIAD is focused to provide the market with a fully autonomous flexible picking solution which is able to integrate with the existing infrastructure of the warehouses without the change of a layout, while leveraging existing technologies without the need to reproduce something new.

1.4.2 REFILLS

REFILLS[8] stands for Robot Enabling Fully Integrated Logistics Lines for Supermarkets. The project is dedicated to improving logistics in a supermarket. The project commenced at the start of 2017 and consists of two universities and five industrial enterprises as collaborating partners. The duration of the project is three years and the objective is to improve and automate the main in-store logistics processes for retail shops. The task of the robot will be to do smart shelf refilling in the stores. The robots will use the knowledge base of the stores acquired through semantic mapping to identify the shelves and identify the missing items and then perform refilling of those shelves. All this will be done by human worker in the loop. Where and when it is required the robots will work alongside the human workers for product monitoring and product refilling and in the absence of the workers, the robots will shift to full autonomous operations after closing hours. The work focusses on the development of individual robotic modules to accomplish the scenarios of store monitoring, collaborative and autonomous shelf refilling.

In store monitoring, fully autonomous robot carrier units will transport passive mobility units to desired location for scanning. The purpose of scanning is to build a semantic map of the environment to have a current more informed representation of the products on the shelves. For the scenario of presorting, autonomous robots will unpack the pallets and do the sorting of packages when the pallets arrive in the store. For the scenario of collaborative shelf refilling, the passive mobility autonomous mobile manipulators will do the task of refilling the shelves in semi-autonomous mode in the presence of the worker. This will be the learning phase of the activity and the robot will do the same activity in full autonomous mode after closing hours in the absence of any worker.

1.4.3 PICK PLACE

Pick and Place[9] is also part of European Union's Horizon 2020 Research and Innovation Program, starting at the start of 2018 with a total budget of 3.085 million and comprises of six partners located in Spain, Germany, Italy and Turkey. The project focuses on the picking, packing and unpacking of products involving manipulating with high variability. This involves the development of a new generation of multifunctional grippers to handle the products with

7 http://iliad-project.eu/

8 http://www.refills-project.eu/

different weights, rigidity and morphology and capable to access complicated grasping positions. This is followed by developing a reactive grasp planning capability based on the cognitive skills developed by learning human grasping skills. In addition, the work focuses on the development of collaborative capabilities of the robot to work in a shared environment. To this end the robot must have the perceptive capability to monitor the environment and the worker, and be able to do on the fly dynamic motion planning to avoid collision or execute compliant motions for task execution. The compliant motions will be executed by using power and force limiting sensors in the robot while avoidance maneuvers will be achieved by exploiting speed and separation monitoring within the shared workspace of the coworker.

1.4.4 COLROBOT

The project Collaborative Robotics[10] for Assembly and Kitting in Smart Manufacturing, commenced at the start of 2016 and ended at the start of 2019 with a budget of 7 million Euros. The consortium consisted of 12 partners with four universities and eight industrial enterprises. The goal of the project is to develop a collaborative mobile solution which is able to deliver parts and tools to the workers. The robot will have the capability to navigate autonomously on the factory floor to pick up the required parts and tools while preparing the kits for assembly, along with the capability to hold a workpiece with the worker for an operational engagement. Globally the work focuses on the interaction and collaboration between the worker and the robot. The interaction focuses on two aspects, the cognitive and the physical. Cognitive interaction will be based on the learning behaviors and will be achieved by robot learning and responding to human gestures, whereas the physical interaction will be achieved by using touch command on robot panels and programming by demonstrations.

In the context of HRI, gesture recognition consists of discerning human body posture. To recognize upper human body posture and gestures concerning the torso and the arms, including the hands, inertial measurement sensors and gesture control arm bands are used in the gesture recognition framework. Since each specific task is encoded with a certain gesture, the robot is expected to perform a task based on the recognition of a certain gesture. An overall task will be a sequence of operations that the robot has to perform. The work in the project is also focused in the development of robust localization and navigation system for the mobile manipulator based on a dynamic map-based representation of the world evolving over time. In addition, the work is augmented by the development of grasping technology, ie. reliable and flexible grippers able to pick variable types of parts with varying size and shape in multiple dynamic scenarios.

9 http://pick-place.eu/

10 https://www.colrobot.eu/

1.4.5 SecondHands

The project "SecondHands[11]" involves the development of a robotic platform to help the maintenance workers in the industrial environment. The robot will be designed as a semi humanoid and will be able to work in a semi structured environment of industry. The robot will be equipped with a variety of sensors e.g. force sensors, cameras and audio for speech synthesis and will be able to perceive, manipulate and navigate to a point for task completion. The robot will have a full understanding of the environment in the form of the knowledge base acquired after learning and will have the advanced sensory and perceptive skills to deal with a dynamic environment. The objective is to acquire an autonomous platform to do or help in the mechanical maintenance work of equipment alongside the workers. Since it has to offer help, therefore the robot will be designed in the perspective of Human Robot Interaction (HRI). Based on machine learning it will be able to learn the behavior of the worker. It will also be able to adapt its behavior in physical coordination while working with the worker. The robot will be able to render worker's request through vision and if not possible, it will interpret the speech of the worker and process the request to trigger the corresponding action. The robot will be able to observe and learn the skills of the worker over time and build a knowledge base, and then perform independently with the knowledge of previously collected worker demonstrations. The robot will handle a large amount of data for learning, collaboration and environment perception and for this the data handling architecture will be cloud based. SecondHands is a collaboration between four universities and an industrial partner.

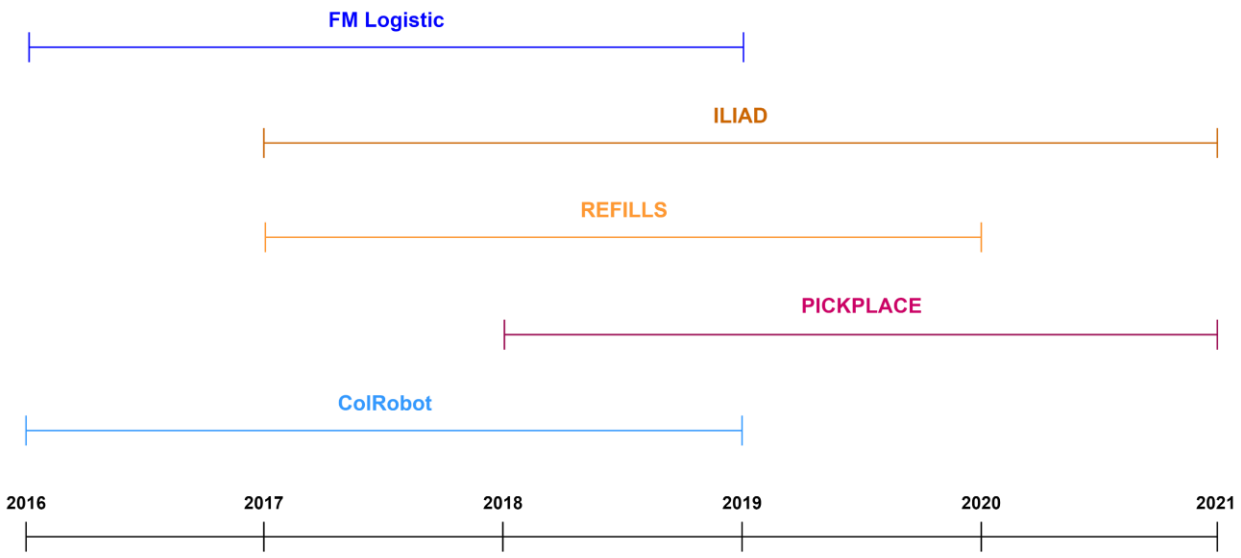


Figure 1.6 : European Manipulation Projects Time Lines

11 https://secondhands.eu/

1.5 Objective of research

The objective of this research project is multifold. The aim of the enterprise is to achieve autonomous palletization by application of robotics technology. Therefore, the research objectives have been laid down and organized in the relevant context. The first goal of this research is to study and identify the primary functional elements which can produce a feasible framework to realize the solution of autonomous palletization. After identification the next step is to prepare the state of the art of each brick, to assess and know what exists and how to proceed. After the conclusion of assessing what has already been done in this domain, the next phase is the development of a prototype as a proof of concept by leveraging the existing technologies. This development will be realized in the context of functional blocks already identified before. Then the next step is to develop methodologies and execution strategies pertaining to each functional block in a systematic way to effectively contribute to a global framework. Specifically, this involves the development of a localization and navigation strategy, motion planning inside the racks, motion planning for manipulation, development of a grasping platform, and a global control and coordination framework to execute the global task. The final objective is then to combine all the elements and demonstrate the execution of the concept.

1.6 Contributions and Structure

Considering the contribution of this research, the technique developed for the motion control inside the racks is particular. Without making a claim, to the authors best knowledge, it has been observed by going through the state of the art in detail, that the use of any similar kind of methodology has not been realized in an industrial warehouse context before. The Hough transform has already been used (mainly with vision) for motion planning for navigation, but particularly for rack tracking and alignment motion using lidar data and non-linear control, this is a first attempt presented in this thesis. In addition, for the management of the global framework of functional elements, the discrete event systems approach (modeling with petri nets) has been utilized in industrial scenarios before, but for the management of execution blocks for a picking task for palletization, the use of discrete event systems with communication exchange, is also a first attempt in this thesis.

To give an overview of the thesis structure, every chapter contains a thorough bibliography of the relevant functional element emphasized. Chapter 2 gives an in-depth review on mobile manipulation, gripper types and gripping technology, collaborative robots and their types, holonomic and non-holonomic mobile robots. Chapter 3 presents the types of robots used to develop the prototype, development of the prototype with hardware details,

gripper selection analysis, its design and associated grasping system. Chapter 4 covers a review of localization methodologies, method used in this thesis, navigation methodology adopted for global navigation, utilization of the path planner, development of algorithm, development of the control, methodology and technique used specifically for rack navigation, development of algorithm, development of controllers and motion planning for manipulation. Chapter 5 is dedicated to the development of the global supervisory framework (behavior control) of the robot. It covers an overview of robot behavior control methodologies and the approach adopted in this thesis. The development of state machines and their analysis in simulation and execution in real time. Last but not least chapter 6 concludes with the lessons learnt in this research, the achievements and the future perspective.

Chapter 2

Mobile Manipulators

The phenomenon of globalization has impacted the industrialization and production of manufacturing enterprises, changing the course of their evolution, requiring a constant upgradation of the production systems to meet the demands of market. Today the world of manufacturing and production has changed with the focus shifting from mass productions to customized ones, scaling up the demand on flexibility of automation. In this context the robotics-based production systems form an essential block of the industrial manufacturing and production lines. Robotic arms based on human morphology were introduced in the industry four decades ago. Today they are performing operations requiring full scale manipulability and applications of material handling, machining, picking, welding, painting etc. However, the conventional industrial robots today are inflexible and inadequate for automation flexibility. This inflexibility of the robots owes to being fixed and dedicated to one kind of application only at one time, and working in a limited work space. Being immoveable, they cannot be adapted to changes in the production requirements. To overcome the limitations and increase their working range by extending their workspace can be achieved by placing the standard manipulator on a mobile base. By providing mobility to the existing static platforms, flexibility of the robots and autonomy of industrial automation is enhanced altogether leading to task flexibility of manufacturing applications. By placing the manipulator on the mobile base, the robot can travel between different destinations points, thus spanning a large workspace. This is more advantageous as compared to traditional static robots for adapting to changing requirements and performing a variety of assembly tasks.

Adding mobility function to manipulation of the arm constitutes a typical characteristic of the mobile manipulator. As compared to the fixed base manipulator, the dexterous workspace gets enlarged considerably due to the movement of the mobile base. A particular point in space can be reached in multiple configurations, ie. by moving the mobile base or the arm or by moving both simultaneously, due to the independent mobility of the manipulator and mobile base. The significance of the increased manipulability and dexterity leads to the fact that the overall mechanism becomes kinematically redundant due to the addition of the mobile base, which is specifically different from the redundancy acquired by the introduction of an extra joint. This redundancy produces dexterous motion, allowing the mobile manipulator system to be configured for maximum performance in the presence of constraints

such as joint limits, actuator torques or obstacle avoidance. Adding a mobile base adds an extra degree of freedom to the overall system making it more dexterous and redundant. The redundancy allows to achieve a secondary task along with the execution of the primary task. This secondary task can be avoiding obstacle, optimizing joint torques or keeping a fixed tool position while following an assembly line.

Mobile manipulators provide an increased efficiency in material handling and manipulation, with the capability to perform tasks in an environment inaccessible or hazardous to humans. Due to increase of manipulability by the addition of mobility, these robots are being used for material handling and transfer, maintenance and repair, and readily integrated to the factory floors for rapid manufacturing and production. Considering the advantage of manipulability with mobility, a significant amount of work has already been done in the past on mobile manipulators. The research has covered a significant domain of application with the focus ranging from application of material handling for pick-place and cooperation, tracking and scanning in assembly in operations, motion planning for obstacle avoidance and task commutation, stability analysis for tip-over and control, complete design for a specific application and collaboration for production in manufacturing industry. The research has particularly focused on the exploitation of redundancy for the development of these control applications and a comprehensive overview of the work done on mobile manipulation in the past till now is given in the subsequent paragraphs.

2.1 Mobile manipulator research

The first mobile manipulator was introduced in 1984 called “MORO” (Mobiler Roboter) [3] and was used to handle and deliver tools on the shop floor. It was a robot arm installed on a mobile base. The authors in [3] from the university of Aalborg Denmark, provide an explicit concept of autonomous mobile manipulator and their industrial applications. They have presented a framework for mobile manipulators based on the aspect of definition, classification, architecture and application. They have given a comprehensive overview on the state of the art on mobile manipulators on robotic platforms developed from 1984 to 2011, and have highlighted the missing links between the research of academia and industrial applications. They have identified the major entry barriers for mobile manipulators by reviewing the twelve robotic application requirements from an industrial and academic point of view. Further the same authors have presented in [4] their robotic platform to elaborate the concept and working principle of mobile manipulation. A mobile manipulator called the “Little Helper” consisting of a non-holonomic base and 6 DOF industrial arm, equipped with vision and tool changing system is developed based on the concept of a versatile and flexible automation solution. The prototype is designed with off the shelf hardware components and

modular servers-based software architect, making it easy to integrate in an industrial environment. The robot was successfully tested in an industrial environment by feeding the parts to a machine and performing "pick place" of objects on human request. Going back into perspective of research done in the past, in [5] the authors worked on the motion planning of mobile manipulators, based on the compensation of dynamic disturbances for the stability of the mobile manipulators and [6] on the analysis of dynamics of mobile manipulators. Pin and Culioli [7] presented their research and evaluation of a multi-criteria optimization scheme in the presence of multiple task requirements and constraints. The approach is adopted to allow the mobile manipulator to efficiently commute between the tasks in a sequence of tasks in the presence of changing task requirements. They further extended their approach in the next work [9] by extending the optimization criterion for planning the task commutation, which was optimizing the min-max criterion for various task objectives. The static min-max optimization criterion for obstacle avoidance, strength, reach and joint limits was implemented on the HERMIES-III mobile platform for planning of commutation configurations.

Huang et al. [8] presented a control scheme for maintaining or recovering the stability utilizing the method of zero motion planning by stability potential field considering the goal and prohibitive stability states. They further extended their work on stability and presented the methods in [18] and [22] for coordinating vehicle motion planning taking into account manipulator task constraints and its motion planning for platform stability. Yamamoto et al. [10] developed a control algorithm for coordinating the motion of a mobile manipulator. The objective was to control the mobile base so that the manipulator was maintained at a configuration maximizing the manipulability measure. The control measured the joint displacements of the manipulator as inputs and produced the motions for the platform to bring the manipulator into the preferred operating region of manipulability. The algorithm was tested with verified simulation results and then on an actual mobile manipulator with a 6 DOF PUMA 250 arm and a mobile base. Further they developed a control method in [11] for obstacle avoidance incorporating motion coordination. The obstacle avoidance scheme was based on the superquadratic potential functions with coordination in the preferred operating regions allowing the controller to force the mobile manipulator to retain optimal or sub-optimal configurations while avoiding obstacles.

Keiji and Shin in [12] make use of the action primitives for control of the mobile manipulator. These action primitives are executed in sequence, constituting the global door opening behavior. The robot is able to open the door by executing a specific action primitive and using a force compliant control while pushing the door knob to open. Khatib et al. in [13] extended their previous work based on the methodologies of 'Operational space formulation,

Dexterous dynamic coordination, Augmented Object Model and the Virtual linkage model for control of multi-arm robot systems. They extended these methodologies for fixed base manipulators and applied it to mobile manipulators. They utilized two mobile manipulators in a cooperative operation for handling a workpiece, with a decentralized controller based on the augmented object and virtual linkage model. Papadopoulos and Rey in [14] and [23] worked on the tipover stability of the mobile manipulators operating on an uneven terrain while being subjected to external forces and inertial loads. They presented a new tipover stability margin called the Force angle stability margin, which being sensitive to changes to center of mass height, would generate tipover prediction and prevention in real-time application. In [15] the authors worked on the motion planning algorithm for mobile manipulators to execute multiple tasks in sequence. The end effector executed the task of tracking a prespecified trajectory and the required motion planning for commutation configuration for the platform was resolved through a global optimization problem.

The authors in [16] have studied the cooperation of position controlled mobile manipulator on an uneven ground, based on the feedback of position controllers. They studied the compliance needed for cooperation based on feedback control laws. In the same year the authors in [17] also studied the cooperation of multiple mobile manipulators handling a heavy load based on a decentralized impedance controller. Peterson et al. [19] use a mobile manipulator for door opening task with the use of a compliant force torque control technique. The mobile manipulator is kept in an ideal configuration with a controller and at the same time a compliant force torque control is used to approach and grasp the door handle. The control was tested successfully on a holonomic mobile base equipped with a PUMA 50 arm. Velinskyin et al. [20] worked on the design of a mobile manipulator and presented a systematic kinematic analysis for the manipulator mounted on the mobile base for the application of automated highway construction and maintenance. The objective was to generate a design tool by studying the effect of manipulator mounting positions on the overall mobility of the system based on manipulability. Considering the differential kinematics, the manipulability was extended by scaling joint velocities by their maximum values producing scaled manipulability ellipses to study the effect of mounting positions of manipulator. The evaluation was carried out on a platform consisting of a 3 DOF manipulator mounted on a wheel base with two driving wheels.

The authors in [21] presented their work on the trajectory planning and control of mobile manipulators. They studied two examples for their analysis, a differential drive and a car like platform with a two-link manipulator. The work consisted of a model-based planning and control methodology for mobile manipulators to follow desired end effector and base trajectories simultaneously, without violating non-holonomic constraints. The model-based

controller was developed to produce the desired motion of the base for the end effector to track a trajectory for a crack sealing application. Inoue et al. [24] verified their control methodology for the stability of the motion of the mobile manipulator in the presence of an external force to the end effector. In the work [25] the authors evaluated the dynamic stability of a mobile manipulator with 1DOF mobile base and two link manipulators based on the "Zero Moment Point" (ZMP). They studied the stability of the platform for mobility and manipulation using the ZMP by solving the redundancy resolution of the system to optimize the online ZMP compensation representing the stability of the whole platform. Nagatnai and Tanaka in [26] developed a path planning algorithm for the mobile base by keeping the manipulability at the tip of the mounted manipulator. This enabled the locomotion controller to be separated from the manipulator control and the cooperative motion was achieved by the communication between the controllers. They tested the algorithm on a mobile manipulator with a 7 DOF arm for drawing a segment on the wall, ie, end effector tracing a straight-line trajectory.

In [27] Yamamoto et al. worked on the coordination of wheeled mobile manipulators under a collision avoidance situation. Two mobile manipulators were able to change their configurations dynamically to avoid collisions with each other, while maintaining the support for the handled object, and this approach was based on the kinematic and dynamic manipulability for multiple robots simultaneously transporting a single object. In another work [28] the authors presented the development of a book browsing system which uses remotely teleoperated mobile manipulator to grab a book. The robot is able to navigate to the book shelf, identify and grab a book, open the pages and send the images to the user. The robot has been tested in a real-world library environment. In the same year Dong et al. in [29] proposed a motion planning framework for a mobile manipulator in which the base moved at discrete poses while the manipulator moved the end effector to track a segment of the trajectory. The proposed methodology is the opposite of simultaneous motion of the base and end effector to track a trajectory as a redundant system. The control is based on the poses incorporating directional manipulability which depends on the joint angles as well as the path to be tracked. The proposed method was tested on a commercial mobile manipulator for a mock up welding application.

Furuno et al. [30] worked on the method of trajectory planning of non-holonomic mobile manipulators considering the dynamic stability. The proposed method utilized the given end-effector trajectory to generate the mobile manipulator trajectory for stability. A combined dynamic model of the mobile manipulator was analyzed under the ZMP criterion (Huang et al.) for the system stability. Further in [31] the authors presented their work on cooperation of mobile manipulators for performing assembly tasks on a behavior-based

approach. The robots were required to grab and handle the beams to assemble a structure. the robots performed the cooperation in a leader follower cooperative strategy with synchronized control. The identification for the assembly components was done with the use of fiducial markers using stereo vision, and the formation for the cooperative control was maintained by the use of force torque feedback of velocity control. The work in [32] presents a mobile manipulator with a holonomic base and 7 DOF arm equipped with a 4 DOF hand. The objective of the work is to develop and show an end-effector centric control framework, ie. tool-point control where the global task is performed by executing subordinate behaviors by exploiting the redundancy of the platform using the null space projections. The task level framework with the null space projections serves as the underlaying control scheme for the robot. The robot is able to achieve dexterous manipulation with collision free motion through the use of elastic roadmap framework which produces collision free navigation for manipulation, fulfilling position and force constraints imposed by the task as well as obstacles in the environment.

Based on the humanoid robotics approach, Thibodeau et al. [33] studied the effect of whole-body posture control on the ability of a wheeled mobile manipulator. They evaluated the results on a wheeled bi-manual mobile manipulator for pushing and pulling tasks and demonstrated the effectiveness of the approach, as the whole-body posture control allows to apply more force on the environment without increasing the footprint of the platform. Najjaran and Goldenberg in [34] presented the real time motion planning, map building and path planning of an autonomous mobile manipulator for outdoors landmine scanning and detection. The mobile manipulator scanned the area by navigating on the obstacle free path from the terrain map generated by the fusion of laser range finder and ultrasonic range finder. In [35] the work presents the use of mobile manipulator for dynamically tracking a moving vehicle body on an assembly line with two tracking methods. The robot is demonstrated to maintain a constant contact with the moving part by the force feedback controller of the arm through visual pose estimation of a marker and using a laser pose estimation method. In [36] the same authors extend the work of the same robot for a performing a peg in hole type task. The motion of the base and the manipulator is coordinated through resolved motion rate control, and the 7 DOF arm is used to place the end effector on the target location by visual servoing on a marker. The peg is then inserted by the force feedback controller of the end effector maintaining a constant contact with the surface. The system is thus able to do the insertion precisely by employing coordinated control of combined visual and force servoing incorporating a reactive task control.

The authors in [37] have used a mobile manipulator for a door opening task. To tackle the complexity in the planning motions for door opening, they have proposed a graph-based

representation with the use of anytime variant of A* for finding a feasible path and cost function for every action. They have used this approach to take into account the wide variance of conditions for door opening. The framework was tested on a PR2 robot by opening doors with both push and pull strategy with smooth coordinated motion of both arm and mobile base. This approach was successful in opening the doors without collision, however it did not take into account the dynamics of the door. In the same year Advait and Charles presented their assistive mobile manipulator EL-E [38] which is able to retrieve and deliver objects. The robot has been developed to assist the people with motor impairment disabilities for performing pick and place tasks. The robot is able to grasp an object from a flat surface and put it on another flat surface. The location of the object and the placement location is illuminated with the help of a laser pointer and the robot uses a laser scanner to estimate the flat surface from the point cloud. The robot executes navigation and achieves grasping with specialized behaviors which utilize sparse low dimensional task relevant features. The authors evaluated the performance of grasping and manipulation by grabbing multiple objects with side and overhead grasps.

Khatib et al. [39] have worked on the development of a holonomic mobile manipulator system by developing a Powered Caster Wheel (PCV) system equipped with a robot arm. Based on the augmented object model approach they have developed the dynamic models for the robot and tested the effectiveness of the control for the modeled parallel redundant system. The modeling and control produce smooth and accurate motions with the coordination of the arm with the base. In a recent work Madsen et al. [40] evaluated the performance of two mobile manipulators in real world manufacturing environment. Two mobile manipulators have been implemented in a real-world production to collaborate and carry out production of a component of a pump. The assembly of the rotor was performed by one robot while the second was used to transport the finished assembled parts to the warehouse, and feed new parts to the workstations. The mobile manipulators were deployed to test the demand of automation flexibility in a real-world industrial setting for industrial tasks.

2.2 End effector

Considering the effectiveness and utilization of the mobile manipulator, the end effector is the focal point of the development of motion control and coordination of the mobile manipulator. The control paradigms are based on the precise movement of the end effector for dexterous manipulation required for grasping. Considering robotic mechanism and designs and serial manipulators in general, the end effector is the device at the end of the last link of a serial chain. This is put at the end by attaching it to the wrist to interact with the work environment. The design of the end effector depends on the application and dynamics of the

robot. There are two types of end effectors. The “Grippers” and “End of Arm Tools” (EOAT). The EOAT are attached to the end effector to perform operations of manufacturing and monitoring such as welding, drilling. These tools consist of various types and sizes depending on the application and workpiece. The grippers grasp and manipulate objects they interact with. They are used to handle the workpieces during various operations in the environment, in manufacturing applications in industry. These mechanical structures are designed with moveable mechanical parts to contact and take hold of the object.

2.2.1 Grasping strategy

Considering the grasping strategy, the gripper design is influenced by two fundamental aspects. The workpiece or the part that the gripper has to work with and the process or operation which will involve the use of the gripper.

2.2.1.1 Workpiece characteristics

The knowledge of what the gripper has to grasp exactly is critical for design considerations. The gripper might be designed to pick up only one object or it might be designed flexible to have the capability of picking up various objects of different size and shape. The object shape decides how much gripping surface is available, which kind of grasp can be done or is possible. Flat surfaces can be handled with vacuum or magnetic types while other shapes can be handled by jaw, claws or multiple fingers. The size of the part decides the reach for the gripper. The gripper must have the capacity or the enclosing volume in a specified configuration to accommodate the part with a stable grasp. At the same time the gripper must have enough force to withstand the weight of the part for a secure and stable hold. The weight of the part decides the maximum gripping force of retention. The retention force can be assumed to be proportional to the weight to of the object [176], and it further depends on the prehension points provided by the geometry of the object along with the friction offered by the contact made with the gripper and the workpiece. The retention or gripping force is also affected by the situation/distribution of the mass center of gravity of the object and requires the application of counterbalance torques and forces to retain the balance and retention of hold.

2.2.1.2 Gripping Process

The second aspect for the design of the grippers focuses on the actual operation that involves the “use” of the gripper. One of the basic parameters of process governing the design is “Prehension”. The prehension strategy or the way to approach consists of the properties of the location for gripping as well as the accessibility to these location points, to reach and access the part without collisions. The prehension or approach is directly related to object placement

as in certain cases the access points cannot be achieved when the workpiece is lying in indeterminate position, or when several randomly scattered objects are placed in overlapping configurations on a plane. Another variation is when they are placed in a closed volume with known prehension points but difficult to access and the gripper requires selectively removing each object and accessing the right one for gripping. Then another is the case when the gripper has to pick up not one but several items together such as bottles arranged in columns. The prehension strategy is also critical when the gripper has to perform highly precise and accurate gripping in the case of joining and insertion operations of parts where the object has to be grasped at precise prehension point.

2.2.2 Gripper Types

The type and design of the gripper depends on the type of application and characteristics of the object to grasp. Broadly defining the grippers can be passive, conforming to the shape of the object based on the passive compliance or they can be active powered by an actuator. Passive grippers conform to the geometry of the object giving them the ability to grasp a wide range of objects without placement adjustments. An example of a passive gripper is [177] which makes use of an elastic membrane filled with granular material conforming to the shape of the object on contact and vacuum hardened to grip the object rigidly. Still the passive grippers are limited in use, since passive compliance cannot be achieved with every object. The Active grippers are the ones which require the physical activation of the gripper mechanical parts to grasp. Their design is generally based on the anthropomorphic fingers like links and consist of jaws or clamps for grasping. The jaws close with sufficient mechanical force to hold the object firmly. The active grippers are further classified into Impactive, Ingressive, Astrictive and Contigutive grippers.

The Impactive grippers are based on the direct application of force from two directions. The contact is based on the impact of gripper jaws with the workpiece. These jaws are subjected to closure by the mechanical motion of links and connected assemblies as the closure is based on the kinematic model of the gripper. The closure of jaws can be parallel, rotational, planar, or closing towards a central axis, e.g. A three fingered hand. Based on the number of jaws or clamps the most common types of impactive grippers are two finger clamp type, three finger and four finger type hand. For grasping the jaws can execute parallel motion, moving towards or away from each other, or they can execute angular motion moving to close in a swing motion, tracing a curved path similar to motion of a scissor. The jaws can also move radially towards the center of the gripper to grasp hold of. A common example is of a three-jaw chuck. The impactive grippers are actuated either by electromechanical drives, hydraulic drives, pneumatic drives or magnetic drive systems. The electromechanical drive systems

consist of grasping motors to transfer the actuation power to the links for grasping. The hydraulic and pneumatic drive systems consist of hydraulic motor or pneumatic pump connected to a piston cylinder assembly with the electrohydraulic or solenoid valves for precise actuation control of the jaws. The magnetic or electromagnetic grippes consist of electromagnetic actuator for the opening or closing of jaws.

The ingressive grippers make use of intrusive or non-intrusive methods to grasp the objects. Being intrusive, the grippers are equipped with protrusions to penetrate the object's surface and make a hold, or being non-intrusive the grippers grasp the surface of the object by pinching mechanism. The astrictive grippers make use of a continuous holding force (astriction) to grasp instead of applying a direct compressive stress to the surface of workpiece. The most common method of astrictive gripping is vacuum gripping. The vacuum grippers consist of suction cups to mate with and hold the object by vacuum suction. The energy used for suction or astrictive forces is 'Air' and is supplied by a vacuum source. The vacuum source can be produced by various sources, eg. A vacuum pump, Suction Bellows, Pneumatic cylinders and Venturi suction generators. The most common type of sources used in the industry is the vacuum pump and compressors. The suction cups rely on the negative pressure developed during mating and contact with the surface. These cups can be active relying solely on the suction force directly supplied by the vacuum pump or can be passive depending on the suction generated due to their elastic materialistic properties.

The magnetic grippers make use of the magnetic prehension force for grasping. These grippers are used extensively for picking ferrous metal parts. The jaws of the magnetic grippers are fitted with the permanent magnet or electromagnets and the object is firmly attached on contact. The gripping force of the permanent magnet grippers is fixed whereas the force for electromagnet grippers can be controlled providing more retention stability.

2.2.3 Grippers in the industry

The development and use of grippers started from industrial applications. The first robot to perform picking operations with a gripper was the UNIMATE Robot installed in 1961 in the General Motors assembly plant. The robot was used to grasp hot pieces of die cast metal. Today the grippers used in the industry for similar pick and place applications are mostly Impactive and Astrictive grippers. The impactive grippers for heavy duty tasks consist of large clamps or forks, either pneumatically or hydraulically actuated providing the prehension to the object by grabbing it from the sides. These grippers are designed to withstand heavy loads and provide a stable grip during lifting. They are capable of lifting large size packages. A

common example of such grippers are the ones used in logistics for lifting heavy packages to make pallets. One such example of grippers is given below in figure.

Figure 2.1 : Industrial Grippers Clamp type

The second type of gripper used in extensively in the industry are the Astrictive grippers. The astriction or the attraction can be vacuum suction, magnetic adhesion or electro-adhesion. A module of astrictive grippers is used [41] in the packing industry for packing of light and small objects such as chocolates and biscuits, and heavy objects such as cartons. The astriction is also used [42] in a modular vacuum conveyor system to pack objects neatly in trays after they come off the assembly line. Further by providing strong gripping forces, they can pick [43] up slippery and oily metal sheets. Another common application is the picking up of large glass panes during the manufacturing process [44]. In [45], three robotic packing cells were utilized to pack soft drinks into boxes. The end-effector could pick up 12 soft drinks at a time with a vacuum suction cup for every drink. In [46] a flexible assembly stations is introduced for aerospace applications, which consist of 8 actuators with suction cup grippers. This assembly station with the gripper is used to hold the changing surfaces of fuselage and wings during manufacturing. The grippers are fitted with load cells, while the actuators are extended slowly to search for the fuselage. On contact with the part, the suction cups become activated and the suction cup heads are locked in position. Earlier based on a model [47], a suction cup gripper was developed for handling limp materials such as fabrics. The objective was to use the gripper to prevent the fabric from deforming during manual or automatic cutting operations. The end-effector consisted of a flat surface with 0.5 cm holes for suction and outer 0.1 cm holes for positive pressure but this gripper did not have tactile feedback sensors for better part handling. A highly flexible and moldable suction cup was also designed[12] to pick up leather plies and it was able to pick the plies without leaving any imprints on the leather as compare to other commercial vacuum cups.

12 Dini, G.; Failli, F.; Sebastiani, F. Development of Automated Systems for Manipulation and Quality Control of Natural Leather Plies. 10 February 2005. Available online: http://www2.ing.unipi.it/leather_project/ vacuum_cup.htm (accessed on 29 March 2015).

2.3 Collaborative Robots (Cobots)

One of the industrial automation tools introduced thirty years ago, were industrial robots. These robots are being used for a large range of applications namely, manufacturing, machining, welding, painting, assembling and material handling. They are designed to be big and rugged to handle the 'industrial' task-oriented applications. They are usually placed in closed enclosures to prevent any interaction with the dynamic entities which might hinder the operation of the robot, for either the robot might get damaged or the object with which the robot collides. These industrial robots are big and powerful and their initial setup and installation is cumbersome requiring an infrastructure implementation with a cost. Further to reach the high level of production and manufacturing goals, industrial automation is consistently being subjected to mass customization. This introduces workspace uncertainty and reduction of flexibility for automation to track it. Human beings can track the uncertainty and variability but are restricted by their physical capabilities in terms of strength, speed, stamina and repeatability. The solution is to utilize a balance in automation and flexibility to achieve the required goals of production. This approach has resulted in Human Robot Collaboration (HRC), enabling humans and robots to operate jointly to complete collaborative production tasks. Due to the limitations of the industrial robot and requirement of flexibility, a new kind of robot has been introduced to the industry, designed to overcome traditional constraints of industrial robots. The main objective is to make a safe use of these robots alongside the dynamic entities with which they can interact in their workspace. Generally human beings are responsible for controlling the industrial robots, and therefore the operations can be made more secure if the industrial robots are designed to collaborate with their operators. Such new form of robots is called collaborative robots or "Cobots".

The principle is to create a co-worker, to help a human execute tasks that are too hard, cyclic or complex to achieve. They are made to interact with the environment or at least adapt to the changes it to some degree. They are equipped with safety measures loaded with integrated sensors, passive compliance and overcurrent detection. The integrated sensors will feel external forces and if the force is too high the robot will stop its movement. Similarly, passive compliance is produced by mechanical components. If an external force acts on a joint, the joint will submit to the force and either move along the direction of force (in teach mode) or move in the opposite direction in the case of a collision. Also, an overcurrent can be detected by the software when a collision occurs. This acts as a safety future for executing a stop condition. Cobots are used where collaboration is required while enhancing flexibility of automation and increasing production.

2.3.1 Types

While the term collaborative robot has been coined for the robot arms being used in the industry, the definition in general does not restrict its classification, and is applied to a broad range of robots performing collaboration and assistance to operators. Broadly speaking, based on the type of application or assistance required, the collaborative robots can be classified as

- Service Robots
- Recommender systems
- Industrial collaborative arms

2.3.1.1 Service Robots

The service robots are categorized as the collaborative robots which have the quality of mobility and dexterity that can be leveraged for collaborative operations. Based on the type of activity requiring collaborative behavior, these robots are engaged for service as humanoids, AGVs and drones, mobile healthcare units and surgical robots. The humanoids developed for collaboration are being used for team work to support the human operators in various operations. Two such examples are the NASA's Robonaut[12] and Valkyrie[14], which are humanoid robots developed to serve as an astronaut's assistant in space station maintenance operations as well as planetary exploration. Another example is half humanoid [48] developed to help in the assistance of the maintenance of automation equipment. The humanoids equipped with emotion aware capability are also being used to interact with humans to develop a socio-psychological eco system. In that system the robots are helping the elderly and engage with the patients to reciprocate responses by gestures for emotional healing.

Figure 2.2 : Humanoids Robonaut[12], Armar6[13], Vylkeri[14] and Baxter

The mobile robots are either traversing the ground terrain for collaboration or aerial robots (drones) moving in the air scanning the terrain or transporting materials. Examples of ground mobile robots are AGVs being used in logistics for material transportation, mobile robots for outdoors planetary exploration (Mars Rover) and mobile health care units providing long distance interaction and transportation of medicine and drug supply in hospitals. The surgical robots are used in collaboration with the surgical activity of the physician. They are being used for highly precise minimal invasive surgeries. One such example is the commonly known surgical robot daVinci [49].

Figure 2.3 : Aerial Drone, AGV and Mobile Rover

2.3.1.2 Recommender Systems

The recommender systems are smart software systems based on artificial intelligence, used to guide and assist customers for making smart choices online. Many web-based applications are now equipped with these systems to present the content in a more comprehensible way to make the activity of customers easy, by making the content easy to access and understand, easy to make a choice and easy to get a feedback in case of a problem.

2.3.1.3 Industrial Collaborative Arms

The Conventional industrial robots are designed to be big, powerful and strong to complete large-scale batch productions. They are designed to do heavy tasks and having a bigger size, they are constrained in a protective environment (fences), shielding them from external environmental factors and vice versa. Typical applications of these robots are, welding, painting, assembly, pick and place, packaging, palletizing and transportation. They are capable of carrying out repetitive operations with a high degree of accuracy. They are programmed to work autonomously without any intervention during the operation. However, one of the drawbacks of these industrial robot is that the environment needs to be isolated from intervention for a safe operation both for the robot and the operator. To meet the limitations of the conventional industrial robot and to make the production more flexible, the Cobot is now in the industry. Cobots are the collaborative version of the industrial arms.

12 https://spectrum.ieee.org/automaton/robotics/industrial-robots/kit-armar6-humanoid
13 https://en.wikipedia.org/wiki/Robonaut
14 https://fr.wikipedia.org/wiki/Valkyrie_(robot)

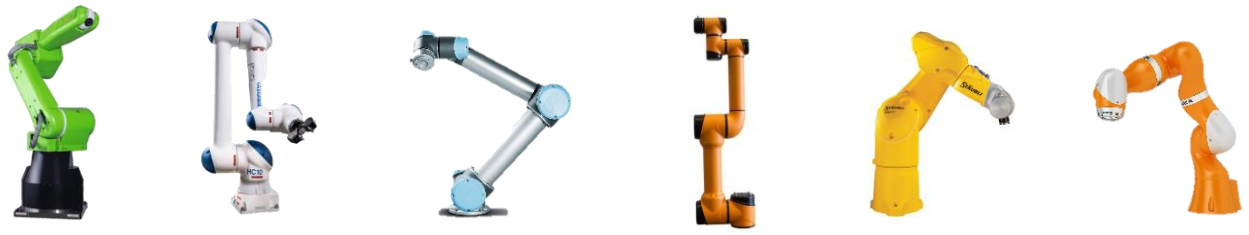


Figure 2.4 : Common Cobotic Arms being used in the Industry

These robots are specifically designed to collaborate with the operators. They come with essential features such as being light weight having round edges, which makes them safe to use in an environment which is open to intervention and interruption. The collaborative robots are designed to work in an open environment alongside operators, sharing the same industrial workspace. This workspace is termed as collaborative workspace. These robots are not confined in protective cages or any other serious protective measures. Traditionally an industrial robot does not come with enhanced safety features enabling it to monitor its environment and operate safely in presence of objects. On the other hand, the Cobots are designed specifically to be collaborative and are built with force and torque sensors enabling them to detect an impact and react accordingly. According to ISO[15] 10218 part 1 and part 2 there are four types of collaborative behaviors based on the performance of their sensorial capability.

- Safety rated monitored stop
- Speed and separation monitoring
- Hand guiding
- Power and force limiting

2.3.1.3.1 Safety rated monitored stop

The robot operating in this mode works in a predetermined safety zone and it stops upon the intrusion of the zone. This feature provides the flexibility to be in the robot's workspace and work on the part at the same time.

2.3.1.3.2 Speed and separation monitoring

This behavior of the robot is much similar to the above. The environment of the robot is monitored by sensors, such as vision to track the movement of the objects and workers. The robot is designed to act as a function of the safety zones. If a human worker or a moving object is within a certain limit of the safety zone, the robot will respond with slowing down gradually

on each crossing of a designated safety zone. The robot comes to a halt when the inner most zone is infiltrated.

2.3.1.3.3 Hand guiding

This feature is used for path teaching to the robot. The robot is taught trajectories to perform post executions. The robot operates in automatic mode and records the trajectory (motion) through force feedback. The force sensing is integrated onto the end effector. Due to this mode the robot is also able to move and support the weight of a heavy object while an operator manipulates it into position.

2.3.1.3.4 Power and force limiting

The robots operating in this mode are made with round and cushioned surfaces integrated with collision detection sensors for detecting a contact and operation shutdown. They are able to detect any abnormal forces on their surfaces while working, and respond instantly by stopping completely, or either shift to passive compliance mode when coming in contact with an operator.

2.3.2 Cobotic Paradigm Research

Due to the far-reaching benefits of Cobotic technology and its effect on the industry in terms of productive outcome, the collaborative robots are reshaping the future. But the implementation of the Cobotic paradigm or utilization of this technological approach, is not merely based on the first hand commissioning and operation of a robot for a specific application. Instead it involves a thorough research analysis of a requirement, constraints and involved elements, based on which a reproduction of an effective strategic methodology is sought for the required collaboration. From the perspective of industrial requirement, the scientific community is progressively contributing in the development of effective strategies for collaboration. In this regard the work primarily focuses on the domains of Safety for collaboration in terms of risk management and collision detection, Anticipation in terms of gesture and intent recognition, motion prediction, Behavior adaptation based on architecture and modeling, Controller designs for compliance, and programming methods for optimal collaboration and interaction consisting of proactive path planning and task allocation. In the preceding sections a comprehensive review of the past and recent progress in research on collaboration is presented. The objective of this overview is to stretch the canvas on the existing strategies and development and how they correlate with the research presented in this thesis. The research in this thesis focuses on the utilization of a mobile manipulator platform, produced by putting together the elements of collaborative robotic technology, since the provision is to use the platform in a dynamic environment in collaboration with the human

15 https://www.iso.org/standard/51330.html

operators to increase productivity. While some of the works presented below do not directly link with the thesis research perspective in the context of exact objective of operation, they however make an indirect connection with the situation and manifestation of factors that influence the long-term autonomy and execution of platforms exhibiting collaborative behavior.

2.3.2.1 Safety based Collaboration

Based on the collaborative safety, the work in [50] proposes a safety-based planning strategy called the "Coexistence Hazard Avoidance Technology" (CHAT). The work explicitly focuses on the development of a risk management software simulator. The objective is to simulate and check collaborative behaviors beforehand to validate the performance of platforms. CHAT is developed to maintain a safety between humans and cobots, and is loaded into the risk management software to check the dynamic planning methods for safe operations of robots. The simulator demonstrates the functioning of CHAT, incorporating two concepts ie. Robotic hazard triangle and Safety policy logic. CHAT is tested on a target platform to analyze the hazards to verify that it is able to comprehensively manage the safe functions of the robots within the collaborative application. Considering safety, the authors in [51] presents the employment of appropriate intrinsic and functional safety design measures for collaborative robots in order to reduce the human injury risks in physical interaction. The focus is on the mechanical and control design of the cobot manipulator for human safety in clamping human accidents. Collision detection and reaction measures were implemented in experiments by varying the parameters of robot inertia, velocity and joint stiffness and it was concluded that joint stiffness plays an important role on collaborative safety, considerably affecting the performance of collision detection. From the thesis research point of view, this safety is critical when the robot is picking items in autonomous mode in the vicinity of other workers in the warehouse.

The work in [52] in the context of collaborative behavior "speed and separation monitoring[15]" derived from ISO/TS15665 focuses on the composition of safety constraints with the production ones. An algorithm is derived in order to maximize the productivity while guaranteeing a safe operating distance of the robot by tracking the human in the workspace and maintaining the execution speed corresponding to safety constraints. The methodology proposes metrics for safety evaluation of human robot collaboration depending on the relative distance and velocity between the robot and human. The research in [53] is in the context of safety requirement of "power and force limiting[15]" specifying safe collisions. Two methods are developed for computing the maximal path velocities complying to safety requirement of which the first addresses collisions with stationary humans, providing a velocity bound based

on the solution to a generalized eigen value problem. The second method applies to collision with moving humans and is based on polynomial optimization. Chen et al. in [54] have presented a real time collision free motion planning and control for a cobot in the context of human robot collaborative safety. The motion planning for collision avoidance is achieved by extracting the object position from a point cloud and then using a combination of potential field and virtual force constraints to avoid the obstacle and execute the task under Cartesian constraints. The strategy adopted in [55] is in the context of 'safe coexistence in a shared space' giving a proactive approach for safe motion planning. The path planning for cobots consists of considering the probability distribution of the volume of human occupancy during robot motion for optimizing the path of the robot for collision avoidance. The proposed approach was tested in a realistic collaborative interaction with the robot dodging the human in advance. This is critical while the robot has to perform picking operations and it has to react on the intervention of the operator close by.

For safe collaboration and minimizing the injury in collisions the work in [56] shows the development of a novel impedance controller based on the concept of awareness of maximum amount of energy a human can tolerate without sustaining injury. The controller allows a safe human-robot interaction due to power and energy limitations enforced by scaling the stiffness and damping of the controller. The overall effect of the chosen control is to enforce a compliant behavior when the energy limits are violated. Further for collaborative assembly in the context of 'speed and separation monitoring[15]' and 'safety monitored stop[15]' the authors [57] have presented a human-aware single axis mobility robotic system incorporating "in time" motion planning based on human motion prediction to execute efficient and safe motions during assembly. Chung et al. in [58] present the safe collaboration by collision monitoring between the worker and the environment and the robot. The approach is based on deep learning and the frame work consist of deep neural network model to learn, detect and recognize any occurrence of collision. The authors validated the performance of the method by observing high sensitivity to collisions and low susceptibility to false alarms as the robot was able to recognize a collision and react preemptively to stop or slow down. Collision monitoring is critical when working inside the racks for picking operations alongside human workers.

2.3.2.2 Anticipation based Collaboration

In the context of Anticipation, the work in [59] presents the intuitive interaction between human and robots with identification of gestures relevant for coworking tasks from human observations. The objective is to transfer the human communication modalities to the robots such as the gestures observed in human to human interaction during collaborative

assembly, and then enable the robots to perform explicit interaction executing common gestures to communicate directly with human beings during collaborative work. This work pertains to the thesis research since the collaborative activity of picking involves the anticipation of worker's actions and the items they want to pick. Another similar work [60] features the embodiment of gestures in the articulated cobot arm to be recognized by human observers. The main objective was to check whether the hesitation gestures exhibited by robot can equally be recognized by a worker as a gesture exhibited by human arms in the case of conflict, and this was achieved by reproducing the human wrist hesitation motion trajectories in the robot wrist. Reaching for the same object at the same time represents a conflict and the overall goal of this work was the conflict resolution of human robot collaboration while working in an industrial scenario. This refers to the fact that during picking, the same item to be picked from a location in a warehouse might present a conflict between the robot and the worker. Again, on anticipative interaction [61] the authors have focused on the issue of non-verbal communication involved in human robot interaction and have presented a method to detect which kind of robot assistance is required based on 'haptic' cues conveyed by the operator. The focus is on the approach of how an operator can communicate his intention to the robot via haptic signals to enable the robot to comply to the worker's actions for collaboration. The method is tested on a common task of co-manipulation of a bulky and rigid object, involving carrying and precise positioning with the help of a cobot (arm), while highlighting the relationship between the relevant wrench components of operator's hand displacement measurements and his intentions. In this context the compliance of cobot is critical for picking up and placing heavy packages on a pallet during collaborative pallet production. The strategy in [62] for 2D case and in [63] for 3D case focuses on the development of anticipative kinematic limitation algorithm, forcing the end-effector to slide along the kinematic constraints rather than stopping completely. The framework generates the respective velocities to make the end-effector slide along the surface (boundary) of the limitation without having to infringe the limitation to generate a repulsion.

2.3.2.3 Collaborative Manipulation

The authors in [64] focuses on the framework of simultaneous manipulation based on an early prediction of human motion. The manipulation is done by the robot and worker independently in close proximity. The method is based on predicting the movements of a human by learning a task specific model of human motion in the offline phase and then use this model to predict human motions in the online phase to simultaneously select a task that interferes the least with humans. The prediction system, produces a workspace occupancy prediction by computing the swept volume of learned motion trajectories, and the motion planner then plans trajectories minimizing a penetration cost in the workspace occupancy

interleaving planning and execution. Predicting the human motion in advance not only improves navigation for obstacle avoidance, but also enhances the picking efficiency by reacting beforehand to avoid conflicting human motions. Sastry et al. in [65] focus on the planning of handoff configurations by the robot by using personalized kinematic models. The human kinematics are estimated directly from motion capture data and used to optimize the hand off configurations (object transfer points) while handing off the object to the operator. The proposed approach was evaluated by observing the human torso movement during hand off operations by the robot. This approach was tested on a dual arm torso for handing the objects to the worker. The thesis research focuses on the utilization of a single arm, but the work is relative in the context when the mobile manipulator utilizes a dual arm configuration. Further on collaboration [66] focuses on the development of a collaborative architecture for human robot collaboration assembly task. The collaborative architecture consists of software modules of face recognition for user identification, gesture recognition for user communication and human like behavior for exhibiting head motions and eye gaze system. The head motions are executed to give a reactive feedback of refusal or acceptance on a gesture recognition and the eye gaze system is used to track the end effector. Likewise, in [67] the authors have focused on the recognition of real time subtasks to provide context aware assistance during generic assembly tasks. The recognition is achieved using multimodal recurrent neural networks with Long Short-Term Memory units (LSTM). The methodology is implemented on a robot that uses subtask recognition system to provide predictive assistance during assembling. Kruse et al. [68] have worked on the collaborative manipulation of highly deformable materials. The work presents a force-vision hybrid controller for a dual arm robot to collaboratively manipulate a fabric with a human. The robot and the human grasp opposite ends of the fabric and the control objective for the robot is to keep the fabric taut while complying to the motions of the operator.

2.3.2.4 Behavior based Collaboration

The work in [69] on collaboration shows the execution of the complex robot behaviors based on hierarchical state machine. The high-level control approach incorporates operator collaboration based on state machine topology, in which the operator can influence the mission execution and modify the behaviors in runtime. This work is closely related to the thesis research since the same state-machine topology is adopted to implement the different frameworks of mobile manipulation. Further another work [70] based on behavior-based collaboration presents a modular cross platform system for authoring robot task plans. These task plans are built and executed on a behavior tree system aided by a user interface. The overall system called 'COSTAR' consists of perception integrated with an abstracted world representation allowing to create task plans robust to environmental variation. Further the

work in [71] shows the learning of human preferences for executing concurrent collaborative tasks. The objective of the work is to make the robot learn and adapt to human preferences by understanding and modeling the concurrent aspects of the shared task and execute it in the way the user prefers. The robot executes the task and simultaneously learns how the user wants to execute it and achieves this by modeling the concurrent multiagent cooperative elements as semi-markov decision problem to learn the human preferences over time.

Haddadin et al. in [72] have developed a layered architecture of multilevel planners for collaborative assembly planning and operation. The top-level planning utilizes an abstract world model incorporating a multiagent human robot team approach to build the collaborative assembly planning, while the execution of the task is done via the agent level planners using hierarchical and concurrent state machines. In the same sense of collaboration in manufacturing systems, the authors in [73] have worked on the optimal tasks allocation in human machine collaboration while taking into account the human fatigue impact on dynamic processes. To capture the stochastic uncertainties of fatigue dynamics under different task assignments, the human fatigue is modelled as a continuous time Markov decision process. Overall the stochastic hybrid feature of manufacturing process with time and event driven dynamics is modelled as a controlled stochastic petri net. The objective of modeling with stochastic petri net is to analyze how human fatigue impacts the process dynamics and how to evolve the task allocation strategy under such conditions. The motion aware paradigm is tested on the collaborative robot 'CobotSAM' to deliver parts for automotive assembly tasks. Human fatigue is a crucial aspect to consider for picking operations. This is one of the main reasons for the employment of collaborative technology to mitigate human fatigue via collaborative help. Mobile manipulation is utilized to reduce the burden of picking on human workers, and this process can be made more efficient if the human fatigue can be modeled in real time to produce effective control and task allocation strategies to complement the fatigue effect. And last but not least the work in [74] gives a comprehensive overview on the collaborative technologies with their use case in the industry aided with the research for programming and communication for an effective collaboration. Specifically, it gives a detailed description of collaborative scenarios in the industry and research domain, with the applied safety standards, programming methodologies and practices. It also highlights the modes of communication for conveying the intent of collaboration and complying to the intent of the user. Further it gives a detailed account of the programming aspects of cobots leading to the optimization of ergonomics, worker's action and performance factors influencing the process, and task parameters to yield the optimal cobot behaviors. While presenting the general programming structure elaborating cobot programming features used for collaborative industrial scenarios,

it provides an insight into the future directions and gives conclusive recommendations for deployments of cobots in industrial settings.

2.4 Mobile Robots (AGV)

With the development of automated guided vehicles (1953) reaching to the fully autonomous cars of today, the mobile robots have come a long way of evolution in research and development. This research was mainly driven by the need to have the full autonomy of the platforms to share the workload of the human operators. Without the need to reflect on the dimensions of the research resolving into evolution of technology, the mobile robots can be broadly classified into three main types based on their operating environment. i.e. Aerial, Ground and Water. Most of these mobile robots are autonomous and are performing various tasks extending human functional capabilities while increasing productivity. The utilization of a specific type of robot depends on the exact requirement. The aerial robots or drones are preforming missions of product delivery in logistics, aerial scanning & photography for surveillance in commercial and defense applications and delivery of pesticides in agriculture. The ground robots are transporting materials in manufacturing, harvesting crops in agriculture, traversing and mapping mines, performing social interaction in public places and scanning terrain and collecting samples of rocks in space. The marine robots are operating under water, scanning and mapping the sea floor, performing wreckage and pipeline inspection in subsea depths, and doing mine detection and reconnaissance for the navy.

Considering the locomotion on land, the robots are classified into legged and wheeled mobile robots. Seigwart et al. [75] provide a comprehensive classification of both types. With the respect to the presented research the focus here is on the ground mobile robots and more specifically on wheeled robots. A wheeled mobile robot functions as the moveable base of the mobile manipulator. The wheel is an essential part of the mobile robot. Each wheel contributes to the total motion of the robot. The wheels pose specific kinematic constraint due to its shape and geometry. These kinematic constraints as a whole influence the robot's motion and describe the robot's kinematic constraints in general. The wheels are fixed to the robot's chassis, therefore their constraints combine to from the constraints on the overall motion of the robot. In essence the maneuverability, controllability and stability of the robot depends on the wheel geometry and the constraint it poses due to this geometry. Based on the mobility of the robot chassis owing to the kinematic constraints imposed by the type of wheels, the mobile robots are broadly classified into holonomic and nonholonomic robots. Considering the constraints [76], a constraint of the form of $f(q,\dot{q},t)=0$ is holonomic if it is integrable to the form $f(q,t)=0$, otherwise it is nonholonomic. More specifically the holonomic constraints can be expressed as an explicit function of position variable only, whereas the non-holonomic

constraint is expressed as a derivative of the position and is non integrable. The basic kinematic constraints imposed by the conventional wheels are the “no slip” and “no slide” constraints. These constrains are non-holonomic and the mobile robots respecting or not respecting these constraints are classified into holonomic and non-holonomic robots.

2.4.1 Non-holonomic robots

The non-holonomic robots have one or more non-holonomic constraints which restricts their motion, on the other hand the holonomic robots are able to move in all the directions on a plane, i.e. to achieve any pose (x, y, θ) in the environment. The influence of the no slide constraint is to restrict the motion in lateral (y-axis) direction. The non-holonomic robots cannot move in the lateral direction and it requires additional maneuvering to reach a point which is easily achievable by holonomic robots. Based on the kinematic models, a brief overview of common types of non-holonomic robots is given below.

2.4.1.1 Unicycle

The unicycle or the kinematic model of the unicycle forms the basis of many types of non-holonomic wheeled mobile robots. The unicycle consists of a conventional fixed wheel rolling on a horizontal plane while keeping itself vertical. The configuration of the unicycle is represented by a vector of generalized coordinates. The position coordinates of the point Q with the ground in the fixed coordinates frame O_{xy} and its orientation angle ϕ with respect to the x-axis is given by $[x, y, \phi]$. The linear velocity of the wheel is given by the speed vector V_Q.

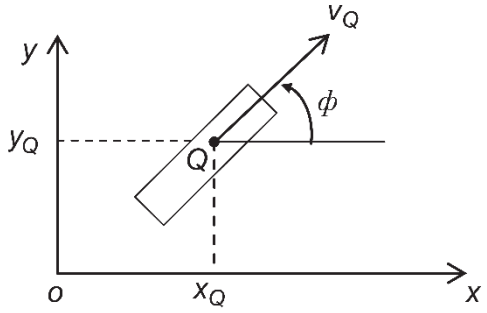


Figure 2.5 : Unicycle Type Kinematic Configuration [76]

2.4.1.2 Differential Drive

The further extension of the unicycle is the differential drive robot consisting of two wheels mounted on the same axes of rotation. The two wheels are non-steerable and are motorized independently. The two wheels are further supported by the addition of a castor or

spherical wheel for to provide at least three contact points to the chassis with the ground. The third wheel is unactuated (passive) and offers no kinematic constraint. The robot orientation varies with the speed of both the wheels separately. If both the wheels turn at the same rate the robot moves forward, if the wheels turn at different rates then the robot follows a curved path along the arc of a circle centered at the Instantaneous Center of Curvature (ICC). Figure below shows the geometric description of the differential drive robot. The ICC changes with the change in the speed vector of the wheels.

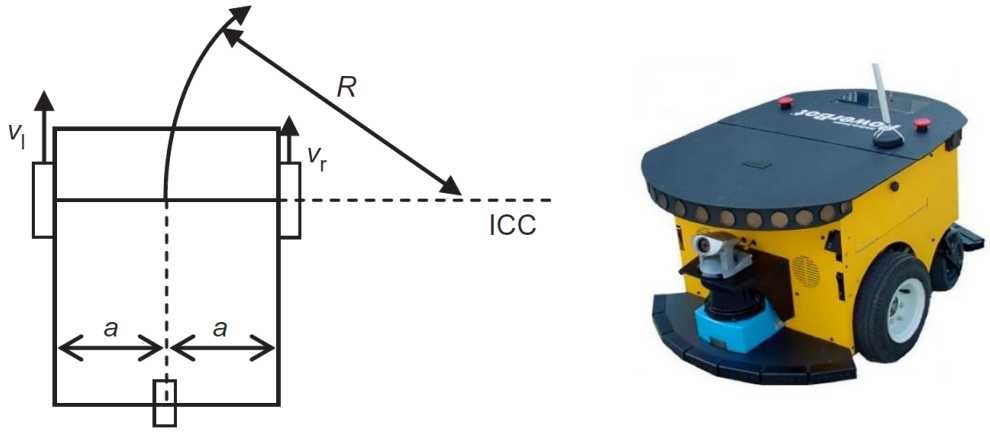


Figure 2.6 : Differential Drive Type Configuration and Powerbot Robot [76]

A common example of the differential drive robot is the Powerbot mobile robot. The Powerbot is a research platform and consists of differential drive wheels and passive castors. The robot can have a maximum speed of 2.1 m/s with a max payload of 75 kg. The basic configuration comes with a 28 module Omnidirectional sonar array for environment detection and LIDAR for laser-based mapping and navigation. A robot arm can also be fitted on the top to convert it into a mobile manipulator.

2.4.1.3 Tricycle

The second type of non-holonomic robot is the tricycle. The robot has two unactuated wheels on a common rotational axis and one driving wheel at the front. The robot has three wheels to have three-point contact of chassis stability. The linear and angular velocities of the wheels are fully decoupled. The front wheel is actuated and it is used for driving and steering. The robot moves forward when the orientation of the front wheel is kept straight. When the orientation of the wheel changes the robot moves in a circle of radius of '*R*', the center of which the lies at Instantaneous Center of Rotation (ICR).

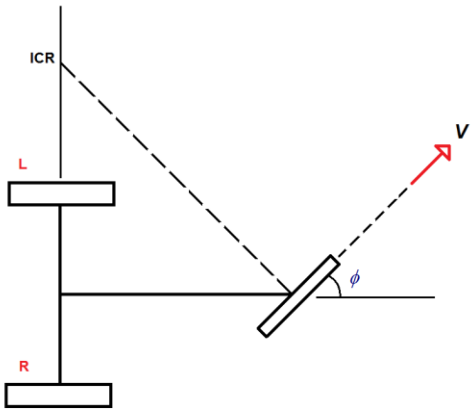


Figure 2.7 : Tricycle Type Configuration

2.4.1.4 Ackerman Steering

The Ackerman steering or carlike configuration differentiates from the standard differential drive robot due to the steering mode of the wheels in the configuration. The Ackerman consists of multiple wheels but generally there are two fixed wheels at the rear and two steerable wheels at the front or vice a versa. The steering of the front wheels effects the change in ICR. The robots can have several wheels but the wheel configuration is such that they must have a single ICR. This configuration is designed to ensure that at turns the wheels of all axes have a common ICR to prevent the geometric wheel slippage. Because of all the zero motion lines or axis of rotation meet at a single point, there is a single solution for robot motion placing the ICR at this point. The Ackerman steering configuration is kinematically equivalent to a single orientable wheel and the robot is classified as (1,1) robot. The Ackerman steering configuration is stable at high speeds but the steering mechanism is complicated.

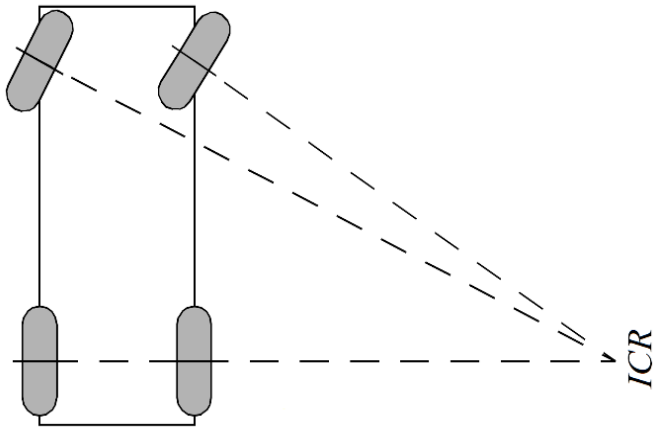


Figure 2.8 : Ackerman Steering Configuration [75]

2.4.1.5 Skid Steering

The skid steering is a special from of the differential drive and consist of vehicles whose drives are constrained by chains or tracks. Common examples are the bulldozers and armored vehicles. Also, another configuration of wheels on the sides. The wheels are constrained to move at the same time on each side. This type of robots has increased maneuverability in uneven terrains, but higher friction due to multiple contact points with the tracks or multiple wheels. To turn the wheels on one side are driven forward and the wheels on the opposite side are driven in reverse and the robot needs a considerable amount of slippage, i.e the wheels are required to skid on the ground.

2.4.2 Holonomic Robots

A holonomic robot is the one which has zero non-holonomic constraints. The robot is able to reach any pose (x, y, θ) in the environment. The goal positions of all the three axes are achieved simultaneously, since the differential degrees of freedom (DDOF) or the mobility is equal to the degrees of freedom (DOF) of the workspace. The number of dimensions in the velocity space of the robot is the number of independent achievable velocities and this is called DDOF. The holonomic robot is also called omnidirectional robot with DDOF=3 [75]. The omnidirectional robot can be constructed by using three or more omnidirectional or specific wheels types which produce omnidirectional motion. Based on the wheel types the omnidirectional robots are classified as follows

1. Spherical wheel omnidirectional
2. Castor wheel omnidirectional
3. Synchronous drive omnidirectional
4. Swedish wheel omnidirectional

2.4.2.1 Spherical wheel omnidirectional

This type of omni robot consists the use of spherical or ball type wheels. The ball wheels do not pose any direct constraint on the motion since the rotation axis of the wheel can have any arbitrary direction. The rotation of the sphere is constrained by the rollers, making a rolling contact with the sphere. The rollers consist of actuated and passive ones. The rolling contacts make non-holonomic contact constraints and the resultant motion of the sphere is holonomic. The robot can be move with a desired linear/angular velocity making smooth and continuous contact between the ground and the sphere, however the design of the spheres supporting assembly is quite complex and is able to support only a low payload due to point

contact between the sphere and rollers. These drawbacks limit the practical application of the spheres for building spherical wheel omni robot.

2.4.2.2 Castor wheel omnidirectional

A holonomic robot can also be constructed with the castor wheels. There should be at least two actuated castor wheels. The third wheel can be passive. Regardless of the wheel orientations, any arbitrary velocities can be generated for controlling the robot. However, the use of castor wheels might lead to instability of the platform when turning the wheels abruptly at high speeds.

2.4.2.3 Synchronous drive omnidirectional

Like the castor wheels, another variation is the use of centered orientable wheels to build an omni robot. However, in contrast to the castor wheel, the wheel orientation of the centered wheel should always be aligned with the desired direction of velocity. The wheels are mechanically coupled such that all of them can rotate simultaneously at the same speed about their steering axis. The mechanical synchronization is achieved with the use of a chain, belt or gear drive. The synchronous robot is a holonomic robot but it cannot drive and rotate at the same time. To change the axis of motion in a plane it first has to stop and then realign its wheels to the new direction of motion.

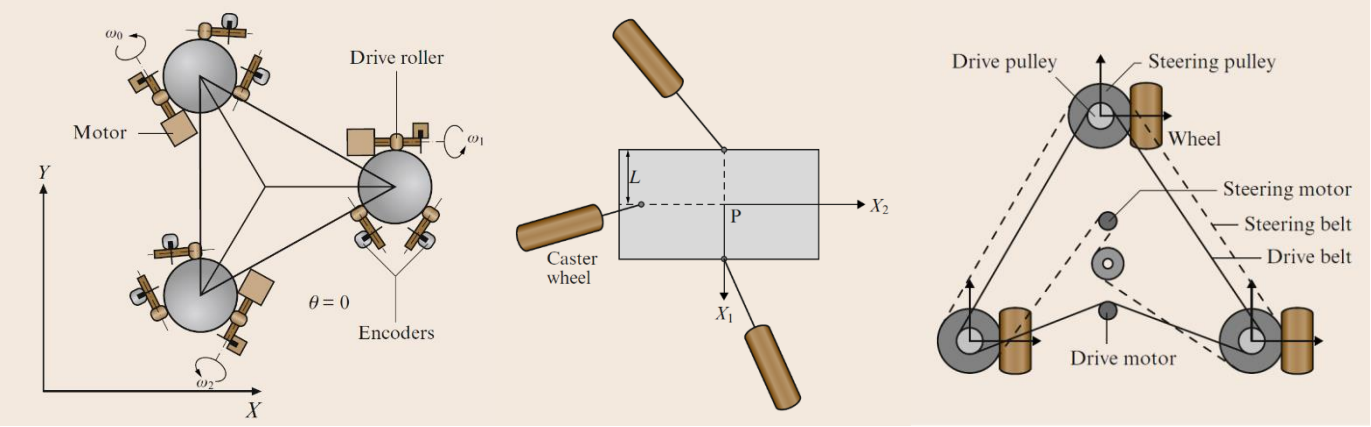


Figure 2.9 : Spherical, Castor and Synchronous omni drive configuration [77]

2.4.2.4 Swedish wheel omnidirectional

This type of robot is the most common and control /movement efficient omni robot. It is constructed with the use of specialized wheels called the 'Swedish' or 'Mecanum' wheels, invented by the Swedish engineer Bengt Ilon[16] in 1973. The wheels are active non-steerable and are also called universal wheels. The wheels have a special property such that they are

constructed with passive free rollers on the outer rim of the wheel. These rollers are employed to eliminate the non-holonomic velocity constraint. In the figure below, the force 'F' produced by the rotation of the wheel acts on the ground via the roller contact. This force is decomposed into $'F_1'$ parallel to the roller axis and $'F_2'$ perpendicular to the roller axis. The force $'F_2'$ produces a small roller rotation speed $'v_r'$ but $'F_1'$ parallel to the roller axis exerts the force on the wheel and thereby on the robot, hence resulting into hub speed $'v_h'$. The actual velocity of the robot is the combination of $'v_r'$ and $'v_h'$. These passive rollers rotate freely on their axis of rotation resulting in the lateral motion of the wheel. As a result, while controlling the drive velocity of the wheel, the lateral velocity is passively determined by the actuation of other wheels. The omni drive can be achieved by the use of three or four wheels and the rollers are mounted onto the wheel either at an angle of 90° or 45° with respect to the hub's circumference.

Figure 2.10 : Forces acting on the wheel and Omni wheels with rollers mounted at 90° and rollers mounted at 45°

2.4.2.4.1 Movement

As shown in the figure below in the four wheels, one pair (1,3) is called the left-handed wheels and the other pair (2,4) is called the right-handed wheels. The mounting angle of rollers in the left-handed wheels is 45° and the angle in the right-handed wheels is -45°. The figure explains the basic motions of the mecanum wheel robot. (A) is forward motion, (B) left sliding, (c) clockwise turning (on spot), (D) backward motion, (E) right sliding, and (F) anticlockwise turning. The arrows correspond to the motion of the wheels and also the motion of the robot. The motions are easily deducible as for forward motions all the wheels should move in the

16 https://en.wikipedia.org/wiki/Mecanum_wheel

same direction. For sliding left 1 and 3 should move in the same direction, and 1 and 4 in the other. By switching the direction of the alternating pairs, the robot will slide to the right.

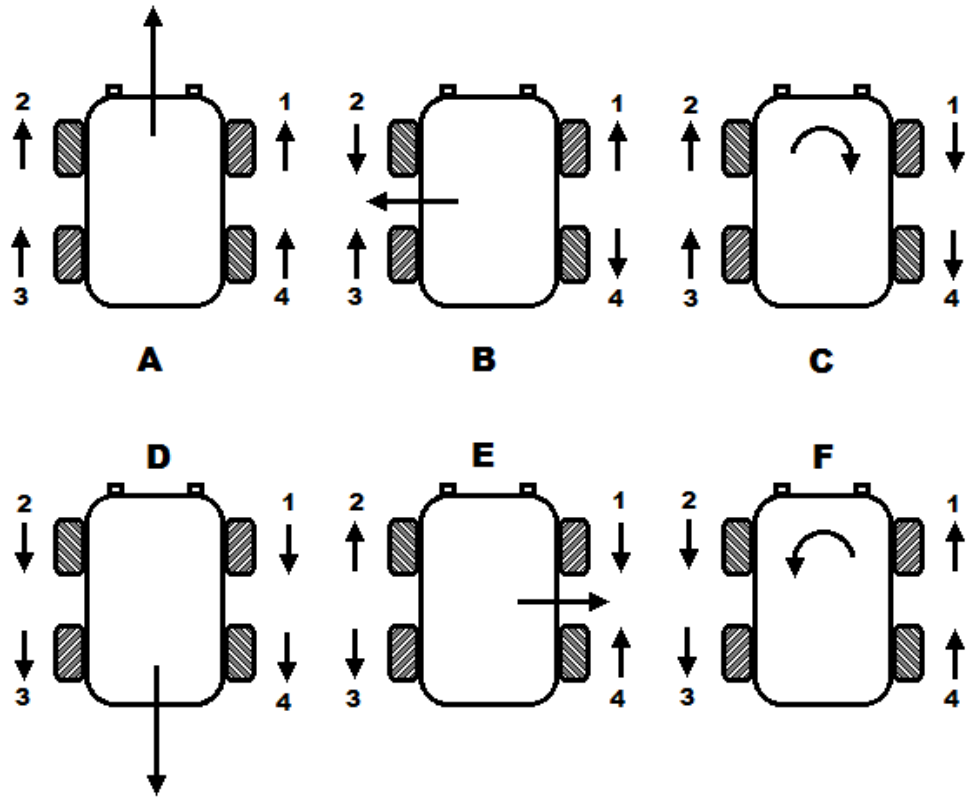


Figure 2.11 : Basic motions of the Robot due to rotations of the wheels

Chapter 3

Platform Hardware Integration

As explained in the introduction, the objective of this research project is to develop a functional framework for product manipulation in the logistic industry. The manipulation concerns the pick, place and pack aspect and formerly speaking the activity to be performed is to do mobile picking for pallet production inside the warehouses. The functional framework to achieve this is materialized on the foundations of scientific concepts realized through the integration of hardware elements. In this chapter the focus is on the hardware elements, while the former part is deliberated to the subsequent chapters. Considering the problem of picking which involves the grasping and displacing of objects, while reflecting on the anthropomorphic approach of object retrieval and handling, the common sense dictates the use of a mobile element combined with a grasping element complemented with adequate articulation. This instinct is endorsed by the fact, that the same approach has been pursued in the previous undertakings of related research and implementation, leading to the conclusion that the optimal system to accomplish autonomous mobile manipulation is to use a mobile robot base with a robotic arm. The design choice of employing exactly the type of mobile base and the type of arm with required degree of articulation, is influenced by the constraints of the process requirement that involve the application and the environment.

The application consists of the task to do and the environment comprises of the operating conditions. Occasionally several applications require nominal control and redundancy to accomplish a task. This is the case when the task requirement encompasses a few parameters of the control to achieve the task than the given degrees of freedom of the platform. In this case the task can be accomplished without the implementation of a complex design structure. Usually the task is complemented with high precision of control and execution, which requires redundant degrees of freedom of the platform to facilitate the precision of control to achieve the objective. A platform providing redundant degrees of freedom generally has a sophisticated design but facilitates the objective at the same time. For example, considering the task of picking, the combination of a holonomic base with a six degree of freedom arm provides redundancy which obviously favors the realization of objective. The redundancy is also linked to the environment or operating conditions. Cluttered environments with limited space of movement require more precise control which can be executed exploiting the available redundancy. Likewise, the dynamic entities in the

environment also pose a significant challenge to the motion planning and control of the platform. For the undertaken research to fulfill the objective of the project, the selectivity of the platform and hardware components was based on the degree of difficulty present and the accompanying constraints of the task of picking and palletizing. In the subsequent sections a comprehensive overview of the platforms used for this research is presented.

3.1 Mobile Base

As compared to a fixed base, a mobile base extends the capability of locomotion of anything connected to it. The base is capable of moving around in the environment, enhancing the workspace of the elements integrated to the base. This degree of enhancement also depends on the design of the base and the type of locomotion it provides. Another factor to consider in the enhancement is the control element. Beyond the extendibility of the workspace, some designs provide more precise control as compared to others. Holonomic robots have the capability to move in all directions on a plane as compared to non-holonomic ones, thus providing more precision on control of motion execution. The focus here is also on the utilization of a holonomic base to meet the constraints of picking in terms of more precise control.

3.1.1 Hardware Type

The mobile base used for this research is the omnidirectional mobile robot "Summit_XlS" developed by the Spanish company Robotnik. The robot is equipped with four mecanum wheels and each wheel has 500-watt brushless DC motor with hall effect sensors. Further the wheels are fitted with encoders to give precise odometry for the robot. The robot can be operated manually with a remote control or by programming and has a top speed of 3 m/s in all the axis. The weight of the robot is 70 kg with a payload capacity of 120kg and the robot comes with an IP54/65 industrial grade casing. The chassis has a width of 50 cm and length of 65 cm including the wheels. The robot operates on 58.4 DC voltage to run all the modules on optimum level and has an autonomy of 10 hours on continuous operation. It is fitted with a core I7 embedded with 4-8 GB of ram able to run Linux (Ubuntu) as the standard operating system. The embedded PC is able to connect to a network through the Wi-Fi module (on board router). Robot Operating System (ROS) is used as the programming OS for the robot for development and research. Run time libraries of ROS are readily available online and from the distributor.

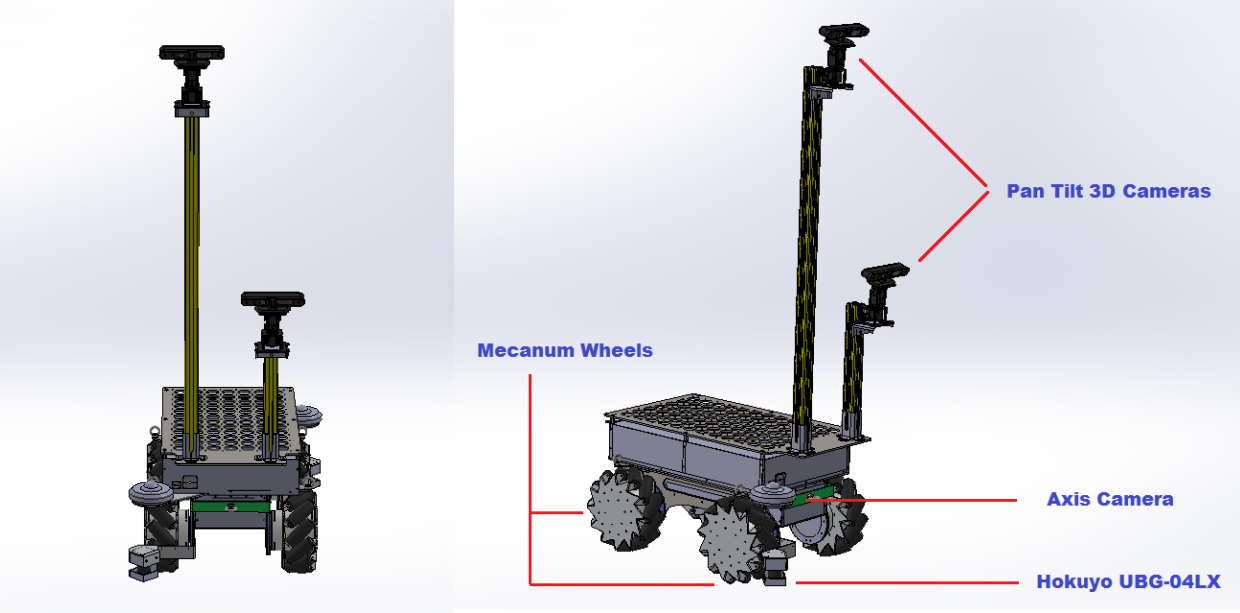
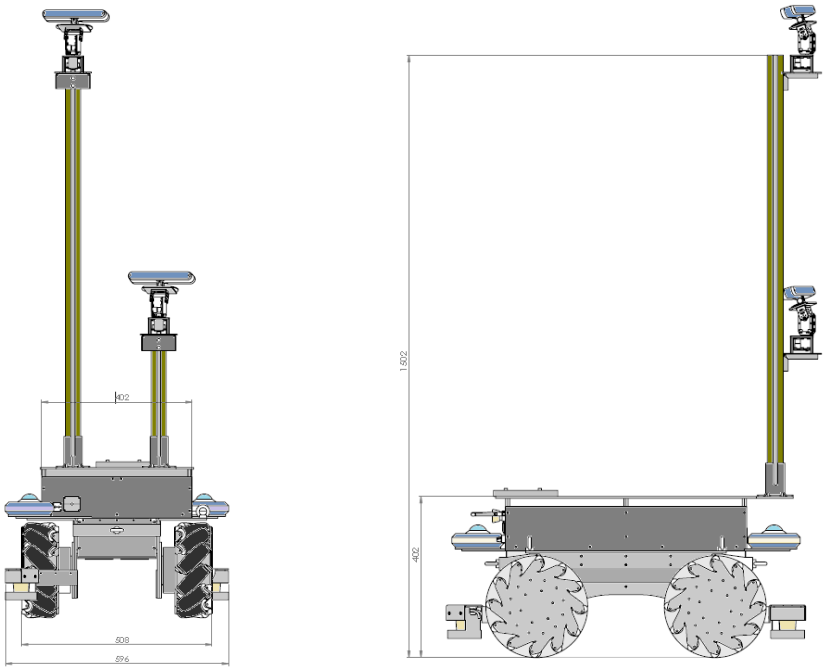


Figure 3.1 : Omnidirectional Robot SummitXLS

The robot is equipped with four RGB cameras. Two RGB cameras "Axis[Ref]" are installed on the base, with one camera on front right at an angle of 45° with the orientation of the robot and the second one on rear left at an angle of 45°. This arrangement of cameras is to provide the robot a full 360° view of the environment around the robot since the cameras have an angle of ±180° pan, a tilt angle of 90° and a 3X digital zoom. Two additional 3D depth cameras are mounted to the robot at the front. These cameras are connected to two separate booms integrated to the base of the robot. The height of the booms is 1.5 meters.



The 3D cameras "Orbec Astra[ref]" facing front provide a depth data for the environment perception and three-dimensional map reconstruction. The range of the cameras is 0.6 to 8 meters. The field of view is 60°H x 49.5°V x 73°A and they provide both the depth and RGB image resolution of 640x480 @ 30 FPS. Further on the base of the robot, two Hokuyo laser range finders are also installed. The placement of the Lidars follows the configuration of 'Axis' cameras, with one lidar fixed at an angle of 45° at front right and the other lidar fixed at rear left at an angle of 45°. This arrangement of lidar is used to measure the distance in all the directions and give the robot almost a 360° view of obstacles around the robot. The range coverage area of the lidars due the present arrangement is shown in figures below. The maximum measurement range of the lidar is 30 meters with a scan angle of 270° and 0.25° of angular resolution.

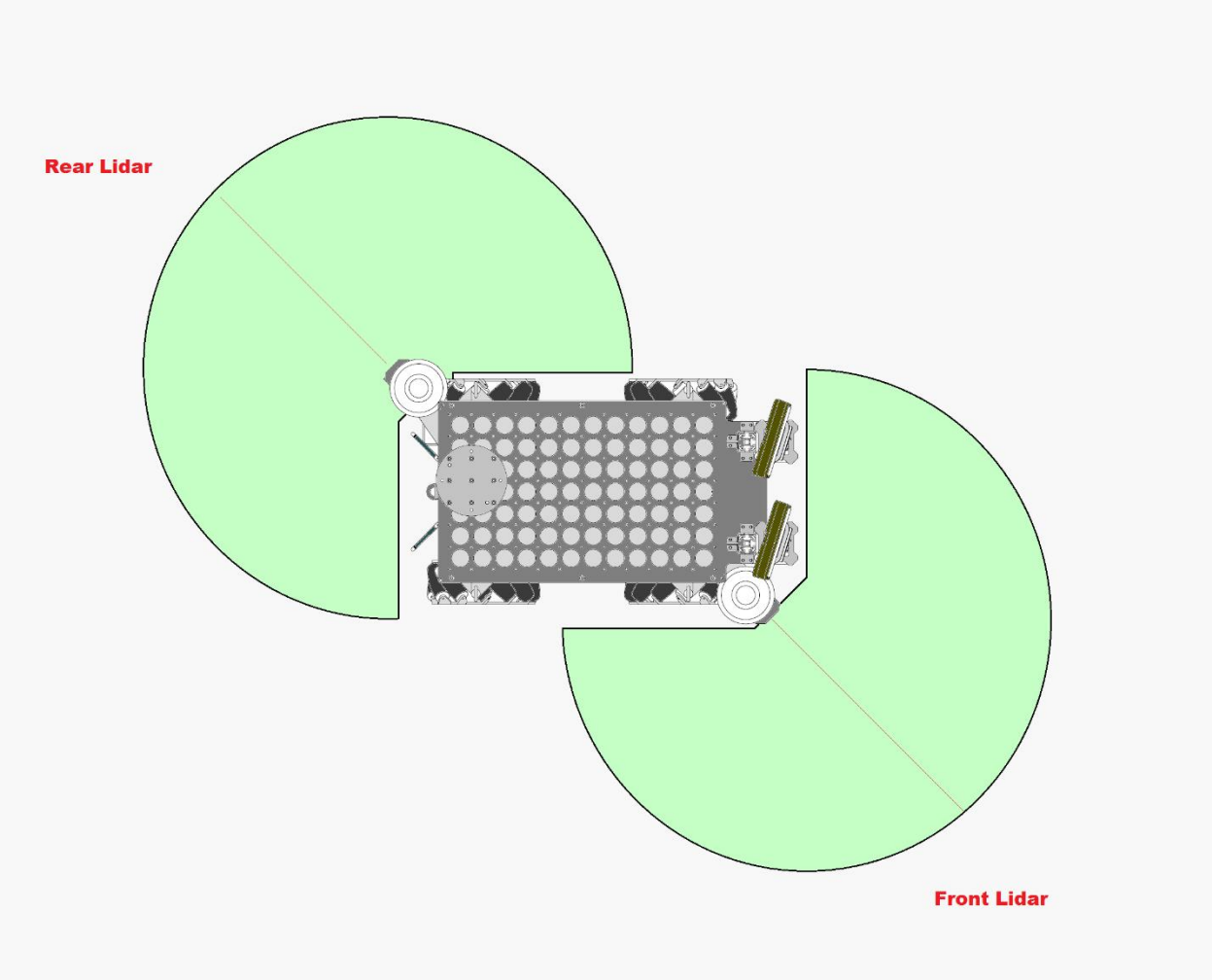


Figure 3.2 : Lidars Configuration coverage Profiles

3.1.2 Kinematic Model Mobile base

The motion of a mobile robot is governed by the kinematic constraints imposed by the wheels. Each wheel moves under a kinematic constraint and contributes to the robot motion. Since the wheels are connected to the robot chassis, therefore the overall motion of the robot is due to the combined constraints of all the wheels. The motion of the robot in a global reference can be studied by studying and expressing the constraint of the wheels in the global reference. The robot below in the figure is modeled as a rigid body on wheels moving in a plane. The total dimensionality of the robot chassis in the plane is three, i.e. two for position on the plane and one for orientation along the vertical axis orthogonal to the plane. By only considering the robot chassis here, the degrees of freedom internal to the robot and its wheels are ignored. An omnidirectional or holonomic robot consist of mecanum wheels and due to the structure and design of these wheels the robot is able to move in all the three axes in the plane. It is able to move in 'x' 'y' and rotate around 'z'. The axis '$'X_I'$' and $'Y_I'$ define the inertial reference as the global reference frame from origin 'O'. The position of the robot is given by point 'P' relative to the two axes $'X_R'$ and $'Y_R'$ which define the local frame of reference of the robot. The position of the robot or point 'P' is given by the $'x_I'$and $'y_I'$ coordinates in the global reference with the angle $'\theta'$ between the global and local reference. The pose of the robot is given by

$$\xi_I = [x_I \;\; y_I \;\; \theta]^T$$

Then the motion of the robot in the robot frame is given by "$\dot{\xi}_I$", and the motion of the robot in the global frame is given by "$\dot{\xi}_I$", where

$$\dot{\xi}_R = [\dot{x}_r \;\; \dot{y}_r \;\; \dot{\theta}] \qquad \text{and} \qquad \dot{\xi}_I = [\dot{x}_I \;\; \dot{y}_I \;\; \dot{\theta}]$$

The motion of the robot is transformed along the component axis of the global or inertial frame with the use of the rotation mapping. The rotation matrix is used to map the motion of the robot from local frame of reference to the global frame of reference. The mapping is given by

$$\dot{\xi}_I = R(\theta)\,\dot{\xi}_R \qquad \text{where} \qquad R(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The mapping between the global frame and to robot frame is given by the orthogonal matrix and is

$$\dot{\xi}_R = R(\theta)^{-1}\ \dot{\xi}_I \qquad \text{where} \qquad R(\theta)^{-1} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Considering the motion constraints of the wheels, there are in general two types of motion constraints for the four basic types of wheels, including the mecanum type. For the motion it is assumed that the plane of the wheel is always vertical and there is always a single point of contact between the wheel and the motion plane(ground). Further it is assumed that there is no sliding at this contact point, i.e. the wheel undergoes pure rolling and rotation about the vertical axis through the contact point. Of the two, the first constraint enforces that the wheel must roll in the direction of motion and the second constraint enforces that the wheel must not slide orthogonal to the wheel plane, i.e. it should not slip. For 'n' mecanum wheels the kinematic motion model [78] of the omnidirectional robot sums up the wheel's constraints in the jacobian matrix which allows the sensing and control of these constraints. Consider a mecanum wheel $'i'$ the figure below in which the wheel frame $'F_i'$ and the roller frame $'f_i'$ are fixed in the robot frame $'F_R'$. The position of the wheel frame $'F_i'$ with respect to the robot frame is given by $'\alpha_i'$, $'l_i'$ and $'\delta_i'$.

`

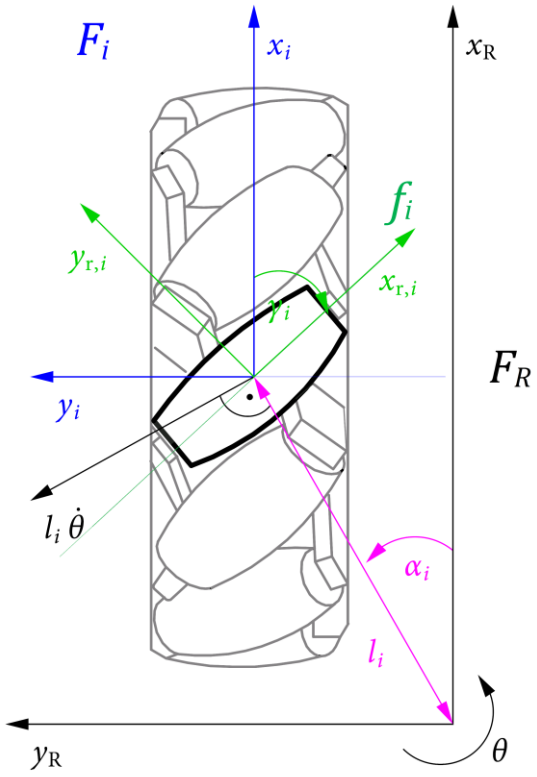


Figure 3.3 : Mecanum wheel Frame assignment

δ_i is the rotation angle between F_i & F_R and is usually equal to zero when the wheel is fixed parallel to the robot chassis. $'\gamma_i'$ is the angle between the roller frame and the wheel frame. $'\varphi_i(t)'$ is the rotation angle of the wheel around its horizontal axis of rotation. The wheel is driven in the direction of it's $'x_i'$axis. The wheel is able to rotate around $'z_i'$ axis about the contact point when the chassis of the robot turns around $'\theta'$. For the motion of the wheel, it is assumed that there is only one roller that makes contact with the ground and the contact point stays always in the center of the roller and the wheel. Both $'f_i'$ and $'F_i'$ have their origin in the contact point. Considering the roller frame $'f_{r,i}'$ for a specific wheel $'i'$, the $'x_{r,i}'$axis lies in the shaft of the roller whereas the wheel is able to move freely in the direction of $'y_{r,i}'$ axis.

For the wheel the instantaneous motion rolling constraint is due to the specific orientation of the small rollers. The zero component of the velocity of the wheel lies along the axis of rotation of the roller. Without spinning the main wheel, the movement in this direction is not possible without slipping. This motion constraint is given by (3.1) by transforming the robot velocities $'\dot{x}_R'$, $'\dot{y}_R'$, and $'l.\dot{\theta}'$ and the wheel's velocity $\dot{x}_i = r.\dot{\varphi}_i'$ in the contact point of the wheel to the roller's frame i.e.

$$\dot{x}_{r,i}^{R} = \dot{x}_R \cos(\gamma_i + \delta_i) + \dot{y}_R \sin(\gamma_i + \delta_i) + l_i \dot{\theta} \cos(\alpha_i + \pi/2 - \delta_i - \gamma_i) \quad \& \quad \dot{x}_{r,i}^{\varphi} = r\dot{\varphi}_i \cos(\gamma_i)$$

$$\dot{y}_{r,i}^{R} = -\dot{x}_R \sin(\gamma_i + \delta_i) + \dot{y}_R \cos(\gamma_i + \delta_i) + l_i \dot{\theta} \sin(\alpha_i + \pi/2 - \delta_i - \gamma_i) \quad \& \quad \dot{y}_{r,i}^{\varphi} = -r\dot{\varphi}_i \sin(\gamma_i)$$

The roller cannot move in the direction of it's shaft so $\dot{x}_{r,i}^{R} = \dot{x}_{r,i}^{\varphi}$ which leads to the rolling constraint

$$\dot{x}_R \cos(\gamma_i + \delta_i) + \dot{y}_R \sin(\gamma_i + \delta_i) + l_i \dot{\theta} \cos(\alpha_i + \pi/2 - \delta_i - \gamma_i) = r\dot{\varphi}_i \cos(\gamma_i) \qquad (3.1)$$

As in the case of the normal wheels, due to the free rotation of the rollers the sliding constraint in the case of mecanum wheels does not enforce that the component of wheel's motion orthogonal to the plane of the wheel is zero. Hence the wheel is able to slide in that direction. The sliding motion is given by the equation (3.2)

$$-\dot{x}_R \sin(\gamma_i + \delta_i) + \dot{y}_R \cos(\gamma_i + \delta_i) + l_i \dot{\theta} \sin(\alpha_i + \pi/2 - \delta_i - \gamma_i) = -r\dot{\varphi}_i \sin(\gamma_i) - r_{roller}\dot{\varphi}_{roller} \qquad (3.2)$$

From (3.1) the inverse kinematic equation of a wheel $'i'$ is given by

$$\dot{\varphi}_i = \frac{1}{r.\cos(\gamma_i)} \begin{pmatrix} \cos(\delta_i + \gamma_i) & \sin(\delta_i + \gamma_i) & l_i \sin(\delta_i + \gamma_i - \alpha_i) \end{pmatrix} . \dot{\xi}_R \qquad (3.3)$$

The angle 'gamma' ensures that the effective direction along which the rolling constraint hold is along the zero component of velocity i.e. along the roller's axis, rather than the wheel plane as in the case of a normal wheel. The rolling constraints for 'n' wheels can be combined together in a single matrix. i.e.

$$\begin{pmatrix} \dot{\varphi}_i \\ . \\ \dot{\varphi}_n \end{pmatrix} = \begin{pmatrix} \frac{1}{r.\cos(\gamma_i)}\left(\cos(\delta_i+\gamma_i) & \sin(\delta_i+\gamma_i) & l_i\sin(\delta_i+\gamma_i-\alpha_i)\right) \\ . \\ \frac{1}{r.\cos(\gamma_n)}\left(\cos(\delta_n+\gamma_n) & \sin(\delta_n+\gamma_n) & l_n\sin(\delta_n+\gamma_n-\alpha_n)\right) \end{pmatrix} \cdot \begin{pmatrix} \dot{x}_R \\ \dot{y}_R \\ \dot{\theta} \end{pmatrix} \tag{3.4}$$

For the four wheels of an omnidirectional platform the inverse kinematic is given by the jacobian matrix obtained by substituting the parameters of the wheel and platform in the above matrix.

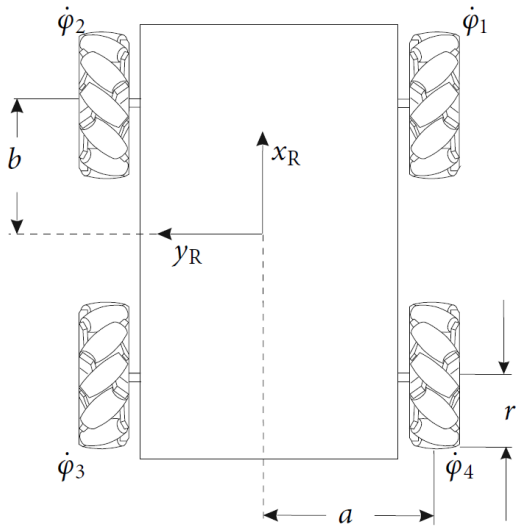


Figure 3.4 : Omnidirectional platform with wheels

w_i	w_1	w_2	w_3	w_4
α_i	$-\tan^{-1}\left(\frac{b}{a}\right)$	$\tan^{-1}\left(\frac{b}{a}\right)$	$\frac{\pi}{2}+\tan^{-1}\left(\frac{b}{a}\right)$	$-\frac{\pi}{2}-\tan^{-1}\left(\frac{b}{a}\right)$
γ_i	$\frac{\pi}{4}$	$-\frac{\pi}{4}$	$\frac{\pi}{4}$	$-\frac{\pi}{4}$

For the four wheel mobile robot shown in the figure above the position of the wheels with respect to the robot frame is given by $\delta_i = 0$ and $l_i = \sqrt{a^2 + b^2}$. Using the configuration parameters in the table above the inverse kinematic is given by (3.5)

$$\dot{\varphi} = J\dot{\xi}_R \Rightarrow \begin{pmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \\ \dot{\varphi}_3 \\ \dot{\varphi}_4 \end{pmatrix} = \frac{1}{r}\begin{pmatrix} 1 & 1 & (a+b) \\ 1 & -1 & -(a+b) \\ 1 & 1 & -(a+b) \\ 1 & -1 & (a+b) \end{pmatrix} \cdot \begin{pmatrix} \dot{x}_R \\ \dot{y}_R \\ \dot{\theta} \end{pmatrix} \tag{3.5}$$

With the jacobian 'J'

$$J = \frac{1}{r}\begin{pmatrix} 1 & 1 & (a+b) \\ 1 & -1 & -(a+b) \\ 1 & 1 & -(a+b) \\ 1 & -1 & (a+b) \end{pmatrix} \tag{3.6}$$

Where 'r' is the radius of each wheel and dimensions of the platform are 'a' and 'b'. The forward kinematics is given by

$$\dot{\xi}_R = J^{+}\dot{\varphi} \Rightarrow \begin{pmatrix} \dot{x}_R \\ \dot{y}_R \\ \dot{\theta} \end{pmatrix} = \frac{r}{4}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ \frac{1}{a+b} & \frac{-1}{a+b} & \frac{-1}{a+b} & \frac{1}{a+b} \end{pmatrix} \cdot \begin{pmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \\ \dot{\varphi}_3 \\ \dot{\varphi}_4 \end{pmatrix} \tag{3.7}$$

where $J^{+} = (J^T J^{-1}) J^T$ when the jacobian matrix is of full rank. Equation (3.5) is used in the motion controller to control the speeds of the robot in the robot frame by controlling the kinematic constraints. The forward kinematics i.e. equation (3.7) is used to execute the odometry by the motion controller.

3.1.3 Kinematic Model Robotic manipulator

The robotic arm or the manipulator is generally made up of links connected together with joints. The joints can be revolute or prismatic. When the robot moves each link moves relative to the other link. The motion of the links or robot as a whole can be described by prescribing the values of certain link parameters. In essence with the help of these four parameters the robotic arm can kinematically be modeled. These four parameters describe the physical attributes and relative orientation of the links along with their connection to the

respective joints. Of the four, two parameters describe the link itself and two describe the links connection to the neighboring link. The modeling of the robot by the use of these parameters correspond to 'Denavit Hartenberg' notation. The four parameters are assigned by first specifying the frames to the links. The relative displacement and orientation of the links is analyzed with the transformation between the frames. The coordinate transformations between the coordinate frames attached to all the links give the relative movement of each link with respect to other e.g. the rotation of the end effector with respect to the base. The method to kinematically model the manipulator consist of first establishing the link frames and then specifying the position and orientation of the frame $\{i\}$ with respect to frame $\{i-1\}$ with the help of four DH parameters. The convention used to attach the frames is that the z-axis $\hat{Z}_i$ of the frame $\{i\}$ is coincident with the joint $'i'$ axis. The origin of the frame is located at the intersection of the link length $'a_i'$ and joint $'i'$ axis. The x-axis $\hat{X}_i$ of the frame points along the link length $'a_i'$ in the direction from joint $'i'$ to $'i+1'$. Then $\hat{Y}_i$ is formed by the right-hand rule to complete the frame. More detail on frame assignment can be found in Ref[craig].

Once the link frames have been attached to the links according to the convention, then the link parameters or DH parameters are defined as

$a_i \rightarrow$ The distance from $\hat{Z}_i$ to $\hat{Z}_{i+1}$, measured along $\hat{X}_i$

$\alpha_i \rightarrow$ The angle from $\hat{Z}_i$ to $\hat{Z}_{i+1}$, measured about $\hat{X}_i$

$d_i \rightarrow$ The distance from $\hat{X}_{i-1}$ to $\hat{X}_i$, measured along $\hat{Z}_i$

$\theta_i \rightarrow$ The angle from $\hat{X}_{i-1}$ to $\hat{X}_i$, measured about $\hat{Z}_i$

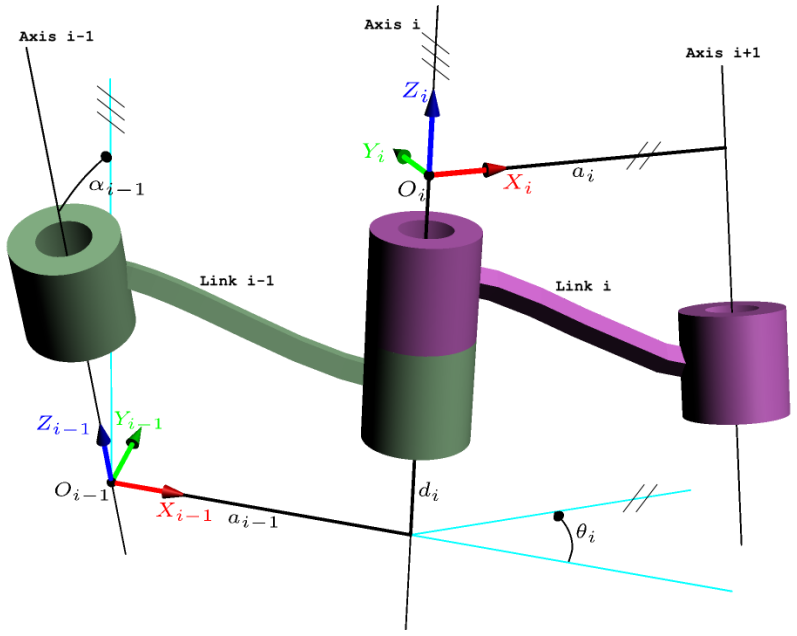


Figure 3.5 : DH Frames

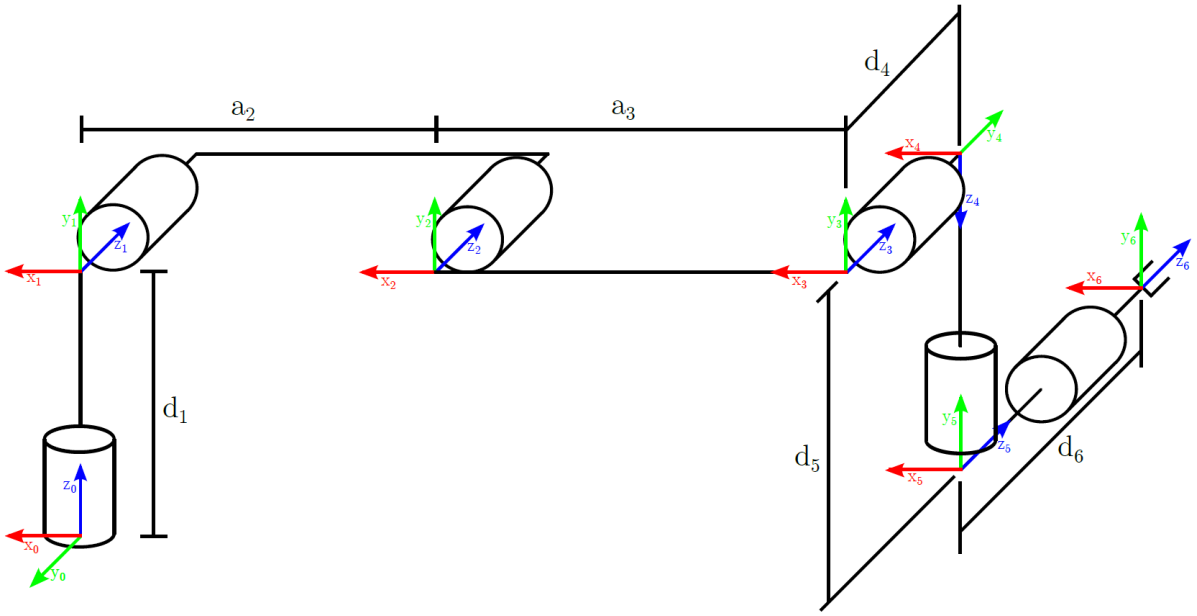


Figure 3.6 : DH Convention Frames Assignement UR5

With all the joint angles at zero figure above shows the common assignment of DH parameters to the UR5 robot. The DH parameters for the above frame assignments are given in the table below

Joint	a_i	α_i	d_i	θ_i
1	0	$\pi/2$	0.089159	θ_1
2	-0.425	0	0	θ_2
3	-0.39255	0	0	θ_3
4	0	$\pi/2$	0.10915	θ_4
5	0	$-\pi/2$	0.09465	θ_5
6	0	0	0.0823	θ_6

Table 1 : UR5 DH Parameters

The homogeneous transformations to get the forward kinematic for the 6DOF arm are then given by transformation from the base frame to the gripper frame i.e.

$$T_6^0(\theta_1,\theta_2,\theta_3,\theta_4,\theta_5,\theta_6) = T_1^0(\theta_1)\,T_2^1(\theta_2)\,T_3^2(\theta_3)\,T_4^3(\theta_4)\,T_5^4(\theta_5)\,T_6^5(\theta_6) \qquad (3.8)$$

The coordinate transformations between the DH frames are given by the compound transformation matrix representing the transformation from the base frame to the target frame

or frame $\{i\}$ relative to frame $\{i-1\}$.This transformation is a function of the link (DH) parameters given by

$$T_i^{i-1} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & \alpha_{i-1} \\ \sin\theta_i \cos(\alpha_{i-1}) & \cos\theta_i \cos(\alpha_{i-1}) & -\sin(\alpha_{i-1}) & -\sin(\alpha_{i-1})d_i \\ \sin\theta_i \sin(\alpha_{i-1}) & \cos\theta_i \sin(\alpha_{i-1}) & \cos(\alpha_{i-1}) & \cos(\alpha_{i-1})d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.9)$$

3.1.4 Hardware Integration

For the undertaken research the Mobile manipulator is developed by integrating the arm on the mobile base. The integration involves the composition of mechanical structure and electronic interfaces for the full functioning of mobile manipulator consisting of dual robotic platforms. An overview of the structural development of the platform is given in the following sections.

3.1.4.1 Mechanical Structure

The objective is to develop a structure which is able to accommodate the hardware and associated accessories of the arm, without increasing the weight of the mobile base and overall platform. For this purpose, lightweight aluminum profile bars with slots are used to fabricate a cage to enclose all the accessories of the arm and also connect extra elements if required. This casing/housing structure is assembled with straight rectangular profile rods to prevent the need for mechanical machining of a structure.

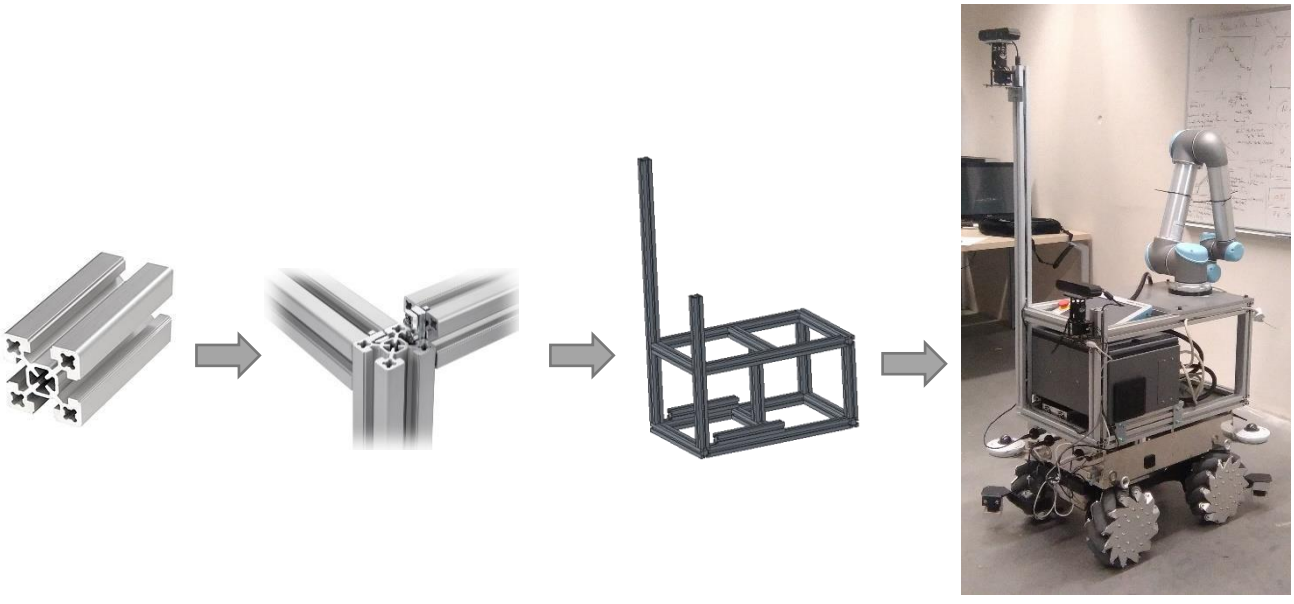


Figure 3.7 : Housing Structure For Mobile Manipulator

These slotted rectangular profiles are connected together by flange brackets forming a rectangular cage to house the controller box, associated electronic boards and the base for the arm. This structure with the slotted aluminum profile provides the flexibility to connect any hardware component to it, and is fitted to the base of the robot with flanges consisting of shock absorbing rubber bushes. This limits the vibrations induced in the frame by the mecanum wheels during the movement of the robot.

3.1.5 Electronic Interfaces

The mobile manipulator requires power for a smooth functioning. The operating voltage of the mobile robot is 52 volts. The robot arm UR5 operates at 24 volts whereas the Control Unit (CB-UR) uses 48 volts for the powering all the components. Since the arm uses vacuum gripper for grasping the packages, the vacuum pump and connected solenoid valve for activation of vacuum is operated at 24 volts. The power unit on the mobile robot itself is sufficient only to power up its own components. Therefore, the mobile manipulator as a whole requires an auxiliary power source to provide power to the associated components involved in the operation. An extra battery of 48 volts is added to the platform to power up the Control Unit (CB-UR) of the UR5 arm, the vacuum pump and the solenoid valve. Below is the representation of schematics of general layout of the electronic components to provide power for functioning of the components.

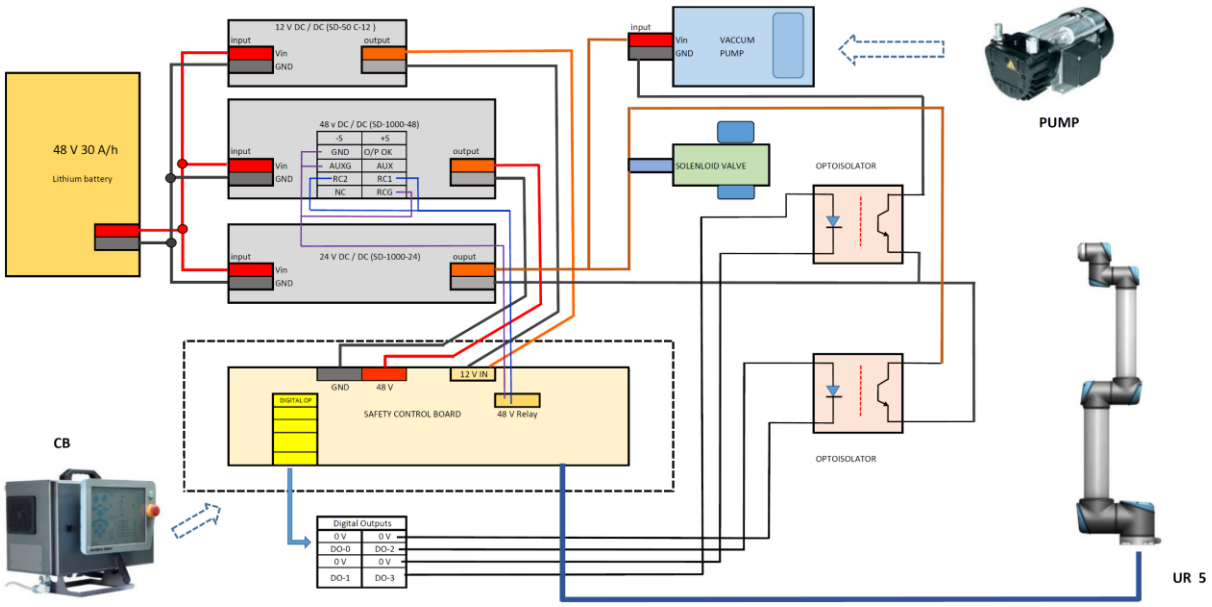


Figure 3.8 : Electronic Component layout and Connections For Arm and Pump

Three DC/DC converters are used to step down the voltage and provide power. One DC converter with 48V output is connected to the control board of the CB-UR5 while the second

DC converter with 24V output is connected to the vacuum pump and solenoid valve through switches (opto-isolators). The signal for these switches is provided from the digital IO's of the CB-UR5. The CB-UR5 is connected to the mobile robot through local area network (LAN). The pump and solenoid valves can be activated by sending the signal to CB-UR5 through the network. In a similar manner the UR5 arm is also engaged by sending the signal to the control unit on the LAN. All the commands to the UR5 control unit are sent through Robot Operating System (ROS) on the mobile robot, providing a single input interface to the user. Hence both the mobile robot, the arm, the vacuum pump and the valve can be controlled at the same time in ROS. The CB-UR5 consists of the embedded software and electronics used to control the UR5 arm. It is provided with a Polyscope robot user interface handheld unit. The handheld unit is called the 'Teach Pendant' which contains the Polyscope software and is used to manually control and program the robot. In the present configuration the UR5 arm can be programmed and controlled both, from the mobile robot (through ROS interface) as well as through the Teach pendant.

3.2 Gripper Selection

The real-time nature of the project, the objective and requirement of a feasible solution necessitates the research to look into the selection of a prehension tool and strategy based on some metrics that align with the requirement. These metrics define all the elements of requirement analysis of the objective (activity) and correspond to the proposed solution. Below is given an overview of these metrics and the strategy opted to produce a solution.

3.2.1 Objective

The global objective is to pick and place objects in the context of a specified activity in the presence of interaction and environment constraints.

3.2.2 Activity

Goods are stored in the warehouse and are dispatched to the client in bulk based on the order. These goods are picked from various locations inside the warehouse and packed together in the form of pallets. The retrieval of goods and production of pallets is termed as "Order picking Activity". This activity is performed by human workers and has to be transformed into an autonomous one with little or no human intervention. The items have to be picked up with a gripping tool well suited to the application.

3.2.3 Environment

The warehouse environment consists of storage locations with a large number of stock keeping units (SKUs) and packing stations utilizing different materials for packing. Typical number of SKUs in a large warehouse range from 15000 to 25000 and are present with a variation in size, shape, weight and genre. The size variation typically ranges from 5 cm to 500 cm with a weight variation of 1 gram to 500 KG. The products come mostly in parallelepiped shape, while some are present in oval, circular and spherical shapes for example food containers and shampoo bottles whereas some products (powdered form) do not have a regular shape and are packed in flexible plastic bags. All the products in general are mostly made of cardboard, plastic, metal and glass. The products are stored at the storage locations inside the racks in the form of pallets with multiple layers. Often the layers are separated with sheets of paper, polyethylene or cellophane, called the Grip sheet to stabilize the cartons or any kind of packages in the pallet.

3.2.4 Focus

For the order preparation there are two kinds of pallets produced generally. The order can either consist of homogenous pallets or a heterogenous one. In a homogenous pallet all the products are the same and mostly packed in boxes. The boxes are stacked together to form layers of the pallet. Retrieval and handling of the box type SKUs is easy since they can be grasped from any planar side. The packing of pallets is regular. In heterogenous pallets the products are not the same and are packed in the pallets with a variation of size, shape and placement. Making a homogenous pallet is more convenient since it requires the handling of similar boxes and stacking then together to form layers, whereas producing a heterogenous pallets is difficult since it requires the retrieval of small and large, regular and irregular shaped

items to stack together in random placement configurations. Retrieval of irregular shaped SKUs is difficult since it requires dexterous manipulation for handling and placement. The scope and focus of the project at hand is to perform box picking and not unit picking, where unit picking refers to the picking of small and irregular items. The approach is to handle and produce homogenous pallets only.

Figure 3.9 : Homogenous and Heterogenous Pallets

3.2.5 Statistical Analysis

The global focus of the project is to do both box picking and unit picking, but as already mentioned, for our research the focus has been narrowed down. Restricting the focus to box picking only, has made the strategy more convenient to choose the range of products to pick. The SKUs or boxes are present with a variation of size, shape and weight. It's not feasible to cover the whole variation. To simplify the approach a specific strategy has to be undertaken to cover a nominal range of the whole distribution. This strategy is based on constraining certain parameters of the products and choosing one specific criterion. The criterion chosen in the undertaken research was the weight. It was decided to use 5 kg weight limit on the products to pick. Two primary factors dominated this choice. One was the availability of the platform (robotic arm) with its payload limit of 5 kg for picking, and the second was the statistical analysis done on the products under the 20 kg limit. From the analysis it was concluded that the distribution of products with the 5 kg limit lies above average. This range is 61% of the whole distribution as can be seen in the figure below. Selecting the weight criterion and choosing the 5 kg limit proved to be a suitable basis to cover a nominal range of products to be picked.

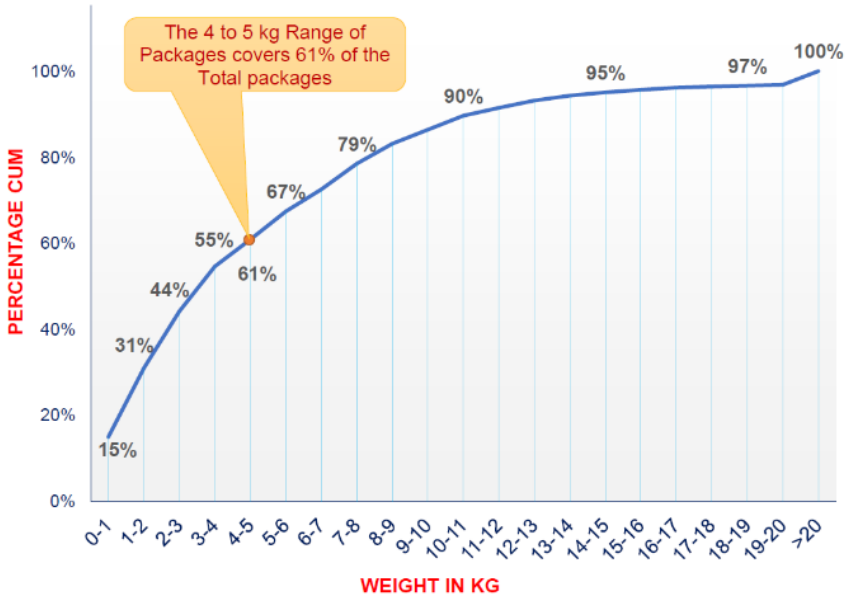


Figure 3.10 : Percentage of Product Range with 5 Kg

Another fact was also taken into consideration regarding the chosen criterion. The weight and shape factors are crucial in the design choice/selection of the gripping tool. The weight decides the optimal size of the gripper and the shape decides what kind of gripping technology is ideal for grasping. To have an idea of the geometry and shape of the products, another analysis for 5 kg limit was done to extract the dimensions of all the products which lead to the conclusion that the products are mostly parallelepiped i.e. packed in boxes. Figure below shows the distribution of products dimensions. The max and min values helped to infer the dimensions of the gripper to use.

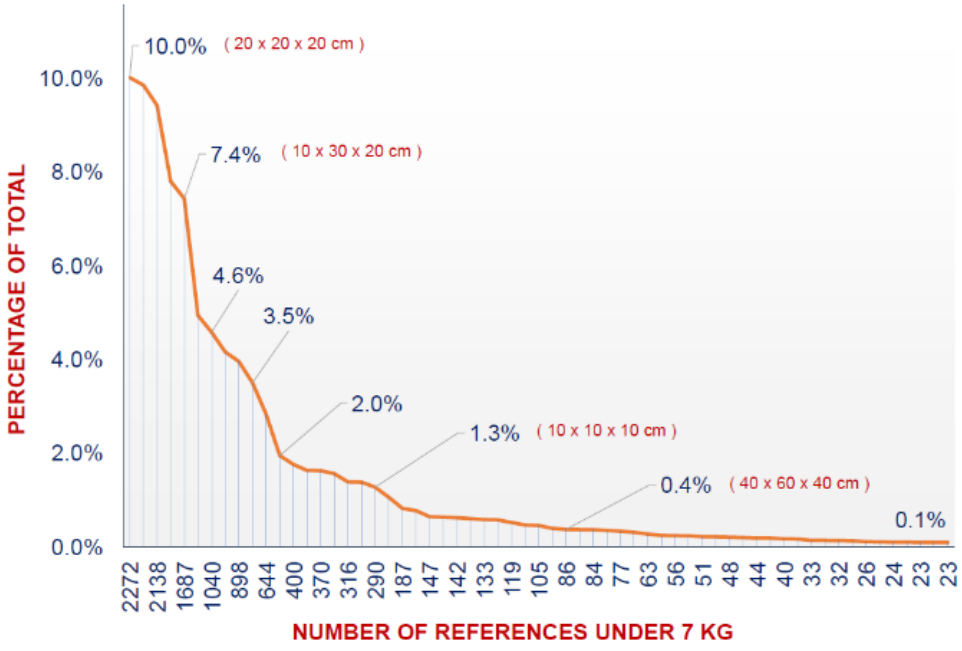


Figure 3.11 : Product Dimension Distribution

The maximum dimensions were found to be (20x20x20) cm and the minimum values were (10x10x10) cm. Based on these dimension tolerances, a custom vacuum gripper with four suction cups was designed and fabricated as a prototype for picking experiments. The suction cups are fixed in the gripper in a rectangular geometric configuration and provide a planar coverage cross section of 13 cm (130mm). The dimensions of the gripper plate containing the suction cups is 12 x 8 cm. The gripper was designed with lightweight aluminum plates, where the rigidity in the structure is provided by the simplicity of the shape. A nozzle connector is placed in the center of the gripper to connect vacuum pipes to the suction cups. For the experiments, different boxes with dimensions ranging from 20 to 40 cm under the 5 kg limit were picked up successfully.

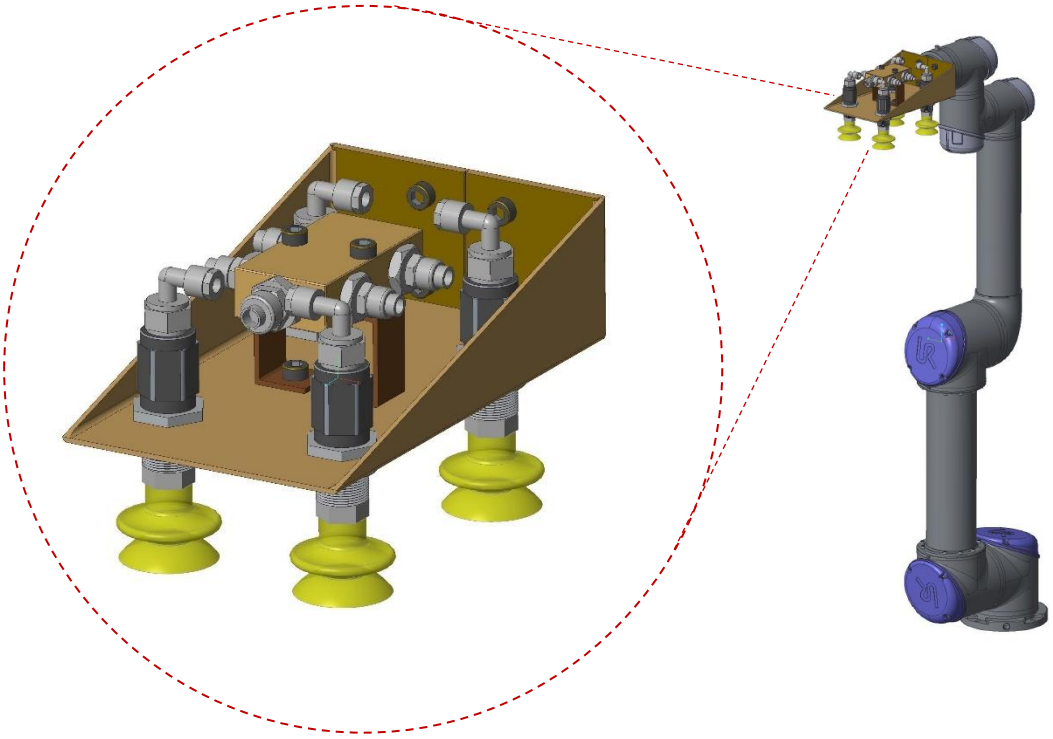


Figure 3.12 : CAD Design of the Custom Vacuum Gripper

Another crucial factor in grasping is the surface material of the package. Some products are made of cardboard and others are of plastic. They are often covered with cellophane, providing less friction to grasp and hold. Parallelepiped products in the range of 5 kg are not very large in size, and offer the most two obvious choices for a gripping strategy. If it is required that the object should be grasped from two sides, then a clamp or finger type gripper can be used. But if the object offers a wide planar surface to grip from one side, then a vacuum gripper is the preferred choice. We also restricted ourselves to vacuum gripping.

3.3 Selection

Based on the aspects of requirement analysis previously, an overview of gripper selection and design methodology are well explained in the subsequent sections.

3.3.1 Design characteristics

The design characteristics pertain to the usability of the gripping tool. The tool is to be used for picking packages with nominal loading and unloading cycles. There is a requirement on the accuracy of placement of the gripping tool when making contact. The prehension points should be equidistant from the gripper center of gravity to provide a good mating surface. The tool must be able to access the side or face of the box with good mating contact and precision. During picking the gripping force should be strong enough to ensure a secure hold to prevent the package from falling. While operating in industrial environment, there is a high probability of packages covered by dust and oily extracts, providing a weak gripping hold since the layer of dust or oil blocks enough friction for a good grip between contact surfaces.

3.3.2 Type

For a strong gripping hold, the weight of the packages will be used to calculate the required retention forces. The basis for computing the retention force is to counter the force due to the weight of the object to handle and maintain a good grip. To decide what kind of gripping technology to use, the accessibility aspect of the objects has to be considered. The objects which have at all times two sides accessible can be gripped better with a clamp type gripper, but if only one side is visible or accessible, which is usually the case, then a vacuum gripper is the preferred choice as the suction cups can grab a workpiece from either one side.

3.3.3 Force calculations

After making the choice of suitable gripping technology to use, there are two types of retention forces required during gripping. One is the "Lifting" force used to lift the package after making contact and the second is the "Holding" force, which is required to ensure a hold during moving. The lifting force is given by the relation

$$\vec{F}_{TH} = m.(\vec{g}+\vec{a}).s \tag{3.10}$$

where $\vec{F}_{TH}$ is the theoretical holding force, '**m**' is the mass of the box, '**g**' is the acceleration due to gravity (9.81 m/s^2), '**a**' is the acceleration of the system, and '**s**' is the safety factor (minimum value 1.5, for critical inhomogeneous or porous materials or rough surfaces 2.0 or

higher). Since the weight limit is 5 kg, therefore using this limit the available theoretical lifting force is

$$\vec{F}_{TH} = 103.1 \text{ N} \tag{3.11}$$

The holding or manipulation force is given by the relation

$$\vec{F}_{TH} = m(\vec{g} + \frac{\vec{a}}{\mu}).s \tag{3.12}$$

where $'\mu'$ is the coefficient of friction and the holding force is calculated as

$$\vec{F}_{TH} = 108.1 \text{ N} \tag{3.13}$$

The friction coefficient is chosen as 0.5 for wood, metal, glass etc. Then a safety factor of 2.0 is chosen for grabbing from the top. The holding force is used during manipulation. Using this force each suction cup force will be calculated. The min-max values of the dimensions of the packages was extracted from the analysis. Then the geometry of the boxes to be handled was taken on average as 20 x 20 x 20 cm ie. L x W x H. A minimum of four suction cups of 40mm diameter were selected to accommodate these dimensions. Four suction cups are enough to cover a face of 20 x 20 cm. By dividing the total force on four suction cups, the force on each cup was found to be

$$F_{s_i} = 27.025 \text{ N} \tag{3.14}$$

This calculated value is the hypothetical value. Using this force, the negative pressure or the vacuum required for each suction cup is computed as

$$P = \frac{F}{A} \quad \rightarrow \quad P = 0.215 \text{ bars} \tag{3.15}$$

This is the vacuum to produce the required force to lift or grab the mass. It is important to mention that the pressure is the same for the entire vacuum system meaning the pressure required for each suction cup and all the cups remains the same. This is the absolute pressure and the fact that the actual negative pressure or the vacuum required is expressed in terms of absolute pressure. Using the relation below to express the above negative pressure in terms of absolute pressure is

$$P_{Abs} = P_{atmos} + P_{negative} \quad \Rightarrow \quad P_{Abs} = 1.013 + (-0.215) \quad \Rightarrow \quad P_{Abs} = 0.798 \text{ bars} \tag{3.16}$$

Considering the pressure distribution shown below in the figures, we need to find the absolute pressure that corresponds to the vacuum pressure required. The standard atmospheric pressure is 1.0132 bar. The Gage pressure is zero at this pressure. The absolute pressure is measured

with respect to a full vacuum. A full vacuum has an absolute pressure of 0 PSIA or -1.0132 bar. From the given reference, to acquire a vacuum of -0.125 bar or 1.812 PSIV, an absolute pressure of 800 mbar, 0.8 bar or 11.603 PSIA is required to be generated by the vacuum generation source.

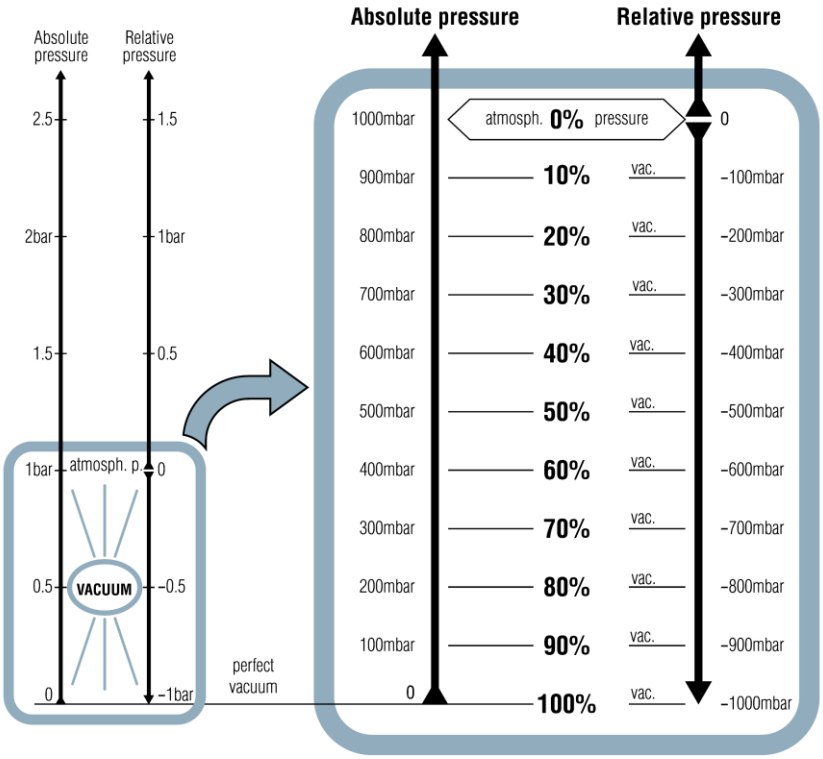


Figure 3.13 : Pressure distribution of Relative and Absolute pressure

3.3.4 Vacuum source

The generation of vacuum requires the computation of flow rate which is calculated by taking into account the volume of air in the suction cups and connecting hoses. Once the flow rate was known, a commercial pump was selected providing a higher nominal flow rate to fulfill the requirement and compensate any vacuum loses. The common technology used to produce vacuum is by rotary vane pump or venturi pump. The rotary vane pump provides a continuous vacuum for constant consumption and is not suitable for periodic vacuum generation. On the other hand, venturi pump is convenient for periodic production. The use of rotary pump requires less elements of integration as compared to venturi pump which requires a compressor and an air reservoir as extra elements and for this reason the installation of rotary pump is preferred over the venturi due to the limitation of space on the robot. The component layout of the vacuum pump which forms the complete circuit is given as under.

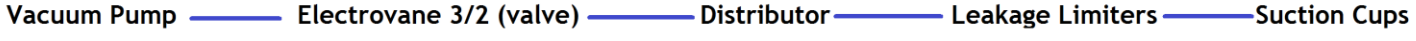


Figure 3.14 : Vacuum Pump Elements Connection Scheme

3.3.5 Stability

An important thing to consider is the gripper stability post grasping. The stability can be separated into static stability, when the gripper holds the box at a fixed position and dynamic stability when the gripper holding the box moves. The stability has to be ensured since if the grip is not stable the package will fall. To this end a stability analysis was done to see the variation of mass center of gravity (COG) of the box as it moves in space. The COG changes during the motion and the resulting components of weight will impart moments and forces on the package due to which the grip will be relaxed resulting in its instability. To study every change of COG for positions at each instant of time over the whole trajectory of motion i.e. dynamic analysis was beyond the scope of the research. Instead just to follow pursuit, a stability analysis for one or more static configurations was done by considering a certain position of gripper with package, and calculating the total force due to moments about different axis with respect to COG. This force was concluded as the instability force when exceeding the maximum gripping force. To remain in the stable region the gripping force must always be greater than the instability force during the motion of the package while being retained.

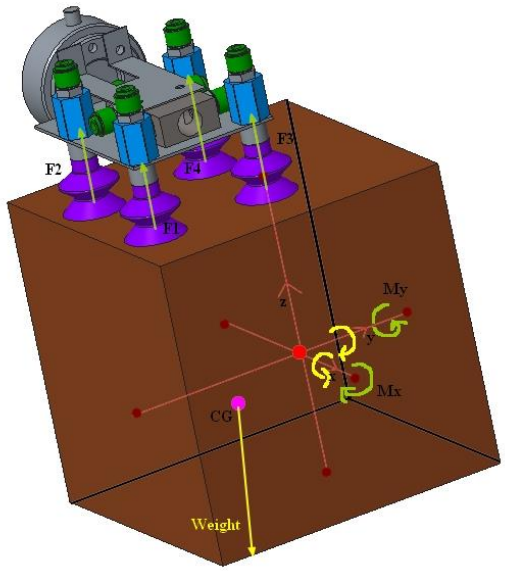


Figure 3.15 : Weight distribution and the resulting moments

From the execution of the algorithm the stable regions were extracted in simulation. The stable regions are shown by green zones, while the unstable regions are shown by red zones below in the plots. The blue points correspond to the position of the suction cups. As long as the COG is kept within the green zone during the motion, the grip will remain stable, but if the external moments are high enough to cause the COG to drift towards the red zone, then the grip will be lost. This stability analysis can be regarded as local since it corresponds to a certain position of the gripper holding the box at a certain angle. In contrast a global stability ensures the full stability of grip during the entire motion of gripper with box, and can be acquired by analyzing the evolution of stability regions of each point of trajectory of motion.

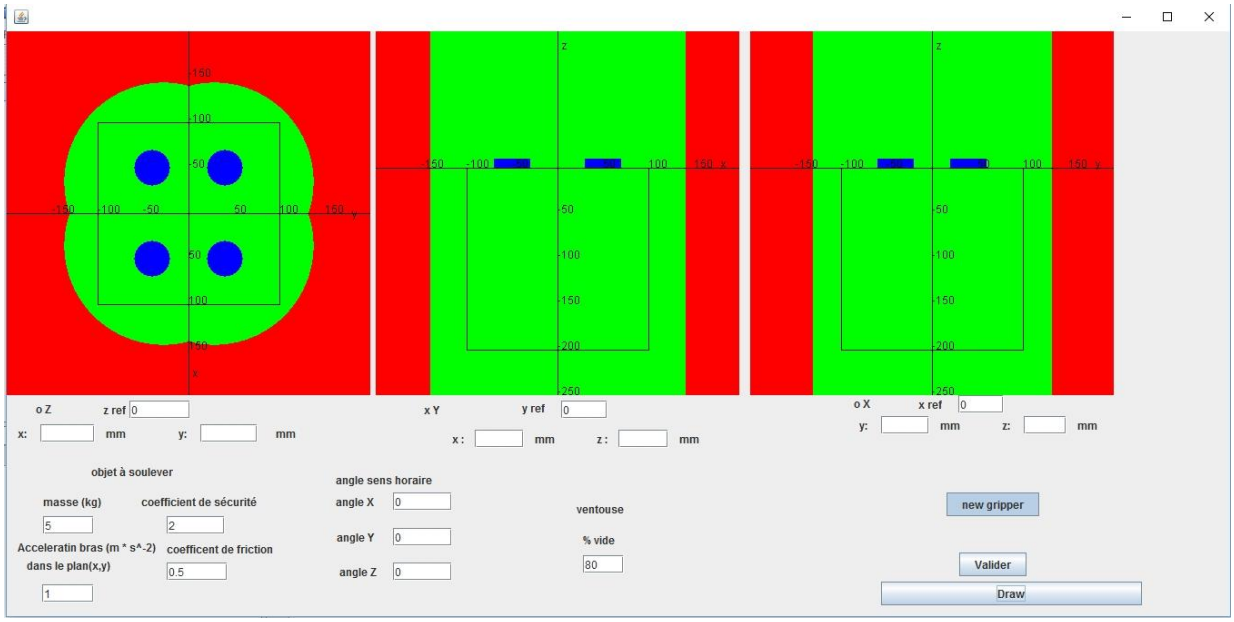


Figure 3.16 : Visualization of the Stable regions

Chapter 4

Localization Navigation Manipulation

Reflecting on the preeminent objective of the undertaken research project the global goal is to achieve autonomous pallet production. In this course the process of palletization seems as the only activity to be performed to pick products and stacking them to produce a homogenous or heterogenous pallet. In actuality the whole process comprises a pipeline of functional elements working in sequence to constitute the global framework of execution, leading to the process of picking. The task of picking can be viewed as a single activity of grasping with adequate articulation as a whole, but in fact arriving to this stage involves the execution of underlaying components functioning in perspective. Specifically, the situation concerns the stage prior to the commencement of picking and object retrieval. Keeping this aspect into perspective, the previous chapter focused on the key hardware components for the implementation of the functional framework whereas the work presented in this chapter focuses on the theoretical concepts and strategies required to facilitate the execution of all the elements in the pipeline, successively leading to the phase of commencement of the picking operation. However, an understanding of the requirement and execution of these elements in the first place is made more clear by detailing the following scenario.

4.1 Scenario

The warehouse manages and stores large volumes of SKUs. These SKUs are collected and packed in pallets and shipped to the consumer based on a specific order of the client. The order receival and packing activity is tracked by the ware house management system (WMS). When an order is received, a worker is assigned the task to pick up the items and the supervision to pick the exact items is provided by the WMS. The sequential and iterative picking of item leads to the production of a pallet for a specific volume. In the automation scenario, the human worker is replaced by a robot that is a mobile manipulator. In the same manner as before, the order is processed by the WMS, which now directs the robot to perform the picking operation. Therefore, for a successful operation the robot has to replicate all the actions of a human worker in successive steps. The operation of picking can be divided into two phases. The first phase consists of reaching to the pick location, while the second phase consists of actual picking. Considering the first phase the issue of concern is on how to get to the destination. This aspect involves where the robot needs to go. For this purpose, prior to

locomotion, the robot needs to have a sense of where it is at the present moment. It has to localize itself with respect to its surrounding environment to know its exact location (position) and then infer the right direction to go towards the destination. Therefore, the initial and fundamental crucial element in the pipeline of global execution framework is the localization element.

For the localization aspect the robot initially needs to have a sense or knowledge of the environment. This knowledge can be provided to the robot in the form of a map of the environment (warehouse) with some indicators in the environment. The robot will extract information from the environment via exteroceptive sensors. It will then localize itself by recognizing the indicators in the environment by using the information acquired and matching it to its own knowledgebase (map). This is how it will correctly infer its location (position) in the environment. The localization strategy and methodology adopted for the current scenario being discussed is presented in the coming sections. Once the robot has localized itself i.e. it correctly knows where it is at the present moment, it then has to plan its locomotion towards the goal (destination point). This locomotion or motion planning has to be done according to some specific strategy. Moving in random directions is not feasible, rather a correct direction to move is determined based on the initial localization estimate and a multi-criteria decision. This decision to choose the right direction combined with the initial estimate is based on the criterions such as a feasible path (obstacle free) and the shortest route possible. Therefore, the next element in the global execution framework is the navigation or motion planning from the start point to the goal point. During the motion the robot also has to keep track of its location and direction in a perpetual manner. It has to be situationally aware. The process of localization and navigation is cyclic with the two components systematically connected and one dependent on the other. The navigation aspect is also dynamic since it has to take into account the obstructions and obstacles present along the way in order to arrive successfully at the goal location.

Once the robot has arrived at the goal location, it has to ascertain its position once again to verify the arrival. After validation it has to proceed to the imperative goal of the task of picking. Comprehensively the task of picking requires the robot to first identify the correct location, navigate to it, place itself close enough for an affordable reach and commence picking with the help of the manipulator arm. Since in the warehouse the picking task involves picking inside the rack locations, therefore the correct placement at a designated point inside the rack locations involves development of a precise control of the platform i.e. a controlled movement or navigation strategy. The development of the control technique is not discussed here and deliberated to the subsequent sections. Once the robot has correctly placed itself, it has to identify the correct items to pick. After identifying the item, the robotic arm is used to

correctly place the gripper to grasp the package properly for lifting it up and placement onto the pallet.

Considering the second phase of the operation of picking, the task of picking may seem as a single activity in its entirety, in fact analogous to the previous phase this task is a complete process with underlaying elements involving execution in sequence. Once the robot has arrived at the designated location, the fundamental aspect for the commencement of the picking phase is the localization of the package or the item to pick. The robot has to localize or extract the coordinates of the package in the rack location for picking. The extraction of the coordinates is used in determining the pick point on the package. Once the package has been localized via a sensor, the next crucial step is to engage the arm to grasp it. The action of pick requires a motion planning of the arm complemented with adequate articulation for manipulation of the package. The motion planning is presented at the end of this chapter. Prior to generating the motion plan the robot has to ascertain that the pick point is within the reachable workspace of the arm or within the reach of the arm. After validating the pick point, the robot will generate a feasible motion plan that ensures the reach of the gripper to the pick point. Execution of this motion plan will move the arm to place the gripper on the pick point, and then lift the package to shift it to the stacking location on a pallet. If in the case the pick point is not within the feasible reach of the arm, the robot will readjust its pose to bring itself to a location to ensure that the pick location is within the reach of the arm. This pose adjustment and position validation can be cyclic to acquire the pose that is convenient for the reachability of the arm. Considering the elements of the pick process and their execution in sequence, the underlaying actions constitute to a certain degree a behavior of the system at hand. The modeling of this sub behavior and all other higher-level behaviors of the system (mobile manipulator) is presented and discussed in detail in chapter 5. The focus here is only on the elements of the first phase i.e. localization and navigation. A comprehensive overview of the methodologies and strategies adapted to materialize the execution of these elements are presented in the subsequent sections.

4.2 Localization

Given an environment the most important question is “where am I ?”. The human beings can answer this question easily by observing their environment and then find their way. When a robot has to move, localization becomes a fundamental requirement for the robot. The robot must be able to localize itself with respect to its environment and then accurately track its pose afterwards during motion. It can do so by the use of exteroceptive and proprioceptive sensors. Ambiguity in the initial localization of the robot will result in an incorrect arrival at a destination which is far from goal. Therefore localization prior to motion

is imperative and tracking the pose with respect to the environment is crucial during the motion to arrive at the goal destination. Based on the existing literature [87] and follow through in research pertaining to localization, the localization methods can be broadly classified into two main groups. The relative or local position estimate (also called dead reckoning) and the absolute or global position estimate (also called reference-based). The two groups are further divided into five general categories.

I. Relative Position Measurements (also called Dead-reckoning)
    1. Odometry
    2. Inertial Navigation
II. Absolute Position Measurements (Reference-based systems)
    3. Active Beacons
    4. Landmark localization
    5. Model Matching

4.2.1 Relative Positioning

In relative positioning methods the current position is calculated from previously determined position and it is advanced based upon estimated speeds over elapsed time and distance travelled. The position estimate is with respect to the system itself and no external reference. Of the first group in relative position measurement, 'Odometry' is the basic and fundamental localization technique. Odometry is based on proprioceptive sensors, such as wheel encoders providing incremental motion information. It is an effective localization technique for short term accuracy, but as the motion by the sensors is incremented over time, the unbounded systematic and non-systematic errors in odometry are also integrated causing large deviations in orientation and position of a vehicle. The odometrical errors increase proportionally to the distance travelled. The systematic errors are present due to the kinematic imperfections of the robot such as unequal wheel diameters, wheel alignment and ambiguity in the construction of the wheel base. The non-systematic errors are the ones due to interacting environment such as uneven floor, bumps and cracks and slippage of wheel on a slippery floor. The systematic errors can be measured quantitatively and can be catered for, while the non-systematic errors are difficult to measure and quantify since they depend strongly on the environment (floor) characteristics. In an extension of odometry, for inertial navigation systems the measurements are provided by accelerometers and gyroscopes to track the position and orientation of an object. However, since measurements are integrated once or twice to yield position, any small constant error also increases without bound after the integration, thus giving a wrong position at a specific instant of time.

4.2.2 Absolute Positioning

In the absolute positioning systems, the position is estimated with respect to an external reference. The most common type of external referencing system is Active beacon systems. These are signal transmission sources conveying their location readily detected by a receiver. The two main types of active beacons systems are visual indicators and powered signal sources. Common examples of visual indicators are the light houses or light towers and powered signal sources such as the radio beacons and radar transponders. Based on the method of measuring the position from these beacons, two types of systems exist i.e. trilateration and triangulation. In trilateration the vehicle's position is measured based on the "distance" from three or more beacons mounted at known locations in the environment. One of the common examples of active beacons based on trilateration is the Global Positioning System (GPS). The system requires a minimum of 24 satellites in earth's orbit. At the moment 31 are currently operational[17]. The receiver's or object's latitude, longitude and altitude on the earth reference is calculated by measuring the signal from the satellite and using trilateration to compute the position. In triangulation the vehicle's position is measured by measuring the angle from the transmitter beacons relative to the longitudinal axis of the vehicle. A minimum of three beacons is required. From three measurements the 'x' and 'y' coordinates and the vehicle's orientation can be computed.

An extension of the beacon-based localization is the landmark localization. In landmark referencing system the position of the vehicle is estimated based on the recognition of distinct features in the environment. These features can be geometric or may include additional information representing a distinct feature such as a barcode. Landmarks have a fixed and known position, relative to which a vehicle can localize itself and once the vehicle is able to detect enough landmarks, its position can be calculated with triangulation or trilateration. There are two main types of landmarks, "artificial" and "natural". The natural landmarks are features pre-existing in the environment, whereas artificial landmarks are specially designed objects such as markers, barcodes and passive reflectors, placed in the environment for localization. Natural landmarks such as vertical edges, doors, wall junctions and corner or an edge are extracted by the use of vision and range sensors. The selection of a feature determines the complexity in feature description, detection and matching for localization. A good feature selection increases the accuracy of localization. The same vision and range sensing are also employed for artificial landmark detection. A significant difference between the use of artificial and natural landmarks is that, the natural ones do not require modification of the environment whereas artificial ones require modification and incurring infrastructure costs. On the other hand, the detection of natural landmarks requires sufficient computational complexity whereas artificial ones can be detected easily since they already

17 https://en.wikipedia.org/wiki/List_of_GPS_satellites

contain encoded information for passive transmission for easy detection. The use of either type depends on the exact requirement and tradeoffs between cost and computation.

In model matching or map-based localization the vehicle uses the map of the environment to localize itself. The robot first scans the environment and builds up a map. Then for localization the robot matches its current instance of the local map with the global map. If the match is found then the robot can compute its position and orientation in the global reference. The generic map types are of two types, Geometric and Topological. Geometric maps are formed by capturing the representation of objects using geometric entities such as lines and polygons and represent the physical location of the objects without their physical attributes. These maps are further decomposed into occupancy grids to reduce the computational burden. The topological maps are a graph representation consisting of nodes and arcs, with the nodes representing the key points and the arcs giving the adjacency between the nodes. For the map-based positioning either the map is provided beforehand or the robot builds the map itself. The building process requires the exploration of the environment and it can be done by manually moving the robot or the robot itself explores the environment. The autonomous exploration is performed without any prior knowledge of the environment under a specific motion strategy that focuses on achieving the maximum amount of scanned area in the least amount of time. One of the caveats of map-based localization is that the map should be accurate enough for a matching with enough readily distinguishable features that can be used for matching. This can be done by having more enhanced features in the map by the fusion of multiple sensors data to acquire a robust and semantic mapping of the environment. The aspect of map matching that is establishing the correspondence between the local and global map, is quite challenging and requires the extraction of features from the environment and determine the correct correspondence between the current and model features. The accuracy of correspondence depends on the quality and consistency of the feature acquired. Yet another more complex problem is Simultaneous Localization and Mapping (SLAM) in which the robot simultaneously explores the environment and localizes itself with respect to the information already acquired during the mapping.

The basic two approaches to implement SLAM are given in [76]. Namely the Extended Kalman filter (EKF) and Markov based SLAM. The problem of estimating the pose of a robot based on the motion and sensor measurements is a non-linear least square problem [88]. If the noise present in the motion and measurement of sensors can be approximated as gaussians then EKF is used to estimate the pose of the robot. This is achieved by predicting the pose using process and sensor model, and then updating the pose by correction using an observation. In EKF-SLAM the motion model and sensor noise is represented as zero mean white gaussian distribution. The EKF is quite effective in estimating the pose of the robot after update from

an observation. It tracks the position of the robot from an initially known position. Robot localization is effectively solved using EKF when the environment contains clearly identifiable markers. The issue with using EKF for SLAM is that, at the end of the exploration, the reobservation of the landmark for loop closure is difficult and this is due to observing the landmark from a different viewpoint as compared to the previous pose of the first observation. Also, the utilization of the linearized models of the non-linear motion and observation models leads to inconsistencies in the produced maps. In the case when the position of the robot is completely unknown then the location of the robot is described by arbitrarily probability distributions. The method of representing the pose of the robot by a hypothesis is termed as 'Markov's localization'. In this paradigm the robot's belief state is represented by the discrete probability assignments for every possible pose in the map. The pose of the robot is assigned a probability P(A) that the robot is indeed at that position. Then Bayes law is used to compute the robot's new belief state (pose) as a function of the sensor's measurement and previous belief state. Both the motion model and observation models are represented as a set of samples of non-gaussian probability distributions. In probabilistic form the SLAM problem is to compute the conditional probability distribution $P(x_k, \mathrm{m} \mid \mathrm{Z}_{0:k}, U_{0:k}, x_0)$ for all times 'k'. This probability distribution [89] describes the *joint* posterior density of the land mark locations $'\mathrm{m}_i'$ and vehicle state $'x_k'$ (at time k) given the recorded observations and control inputs up to and including time k together with the initial state of the robot.

In Markov's localization, to update the probability of all positions within the whole state space at any time requires a discrete representation of the space which imposes a constraint on the required memory and computational power thus limiting the precision and the size of the map. The computational burden associated with markov's localization is proportional to the size of the environment and resolution of the discretization. To reduce the computational burden of the markov's localization employing fixed cell decompositions representations, another approach using the same probabilistic paradigm is employed and is termed as randomized sampling or alternatively known as particle filter method. This technique is also called 'Monte Carlo localization'. In this approach, instead of discretizing the space of robot locations to represent the complete belief state, an approximate belief state is constructed by representing only a subset of the complete set of possible locations to consider. A weighted set of the robot locations estimates, termed as particles is used to describe the probability distribution of the robot location. In this strategy the number of particles used determines the accuracy of the representation or the belief, where each particle represents an estimated pose of the robot location. The weighted sum of these particles describes the best estimated belief of the true location of the robot. In the particle filter method, the current belief is updated with a new sensor measurement. In the update step the weights of the

particles are changed based on the probability that the true robot position corresponds to the particle with the higher weight. Then resampling is done to select the particles with higher probabilities than the lower ones. This iterative update and resampling finally converges to the true estimated position of the robot.

4.2.3 Localization Indoor and Outdoor

The localization techniques discussed in the previous sections are applicable to both indoor and outdoor localization. The odometry is reliable for short term accurate localization, however these systems accumulate errors over long runs due to integration of noise and sensor biases. To overcome the uncertainty of drift, the basic localization methods are combined with other absolute referencing systems such as utilization of beacons and landmarks to estimate the position. For outdoors the localization is easily solved with the use of GPS. GPS provides an absolute referencing and the object/vehicle can be localized with a feasible range of centimeters in open environments. However, in the urban areas where there is a chance of signal blockage, the GPS is combined with inertial navigation systems (INS) to give a better estimate of position. Apart from the GPS the vehicle can localize itself with respect to pre-existing natural landmarks or beacons in the outdoor environment. For example, ships in the sea localize themselves with light towers and radar stations on the coast and correct their position. In the same sense the vehicles in the urban environments, estimate their position by observing buildings, houses, streets layouts and natural landmarks such as trees. In indoors the landmarks to detect can be natural features such as doors, walls corners, or artificial beacons such as RFID tags, markers and reflectors. The vehicle can effectively localize itself on detecting these beacons. For map-based localization in both indoors and outdoors, when the map is provided beforehand the robot localizes itself in the map by observing a landmark and making a correspondence between the current and local and the global map. In the case when the robot has to explore the environment either indoors or outdoors the map formation becomes complex. The mapping requires the registration of features either in the 2D or 3D map. The map is formulated in the form of an occupancy grid with each cell having the probability of being occupied or not. The extracted features such as obstacles are registered as occupied cells. In the 3D case the mapping becomes more complex since the size of the occupancy grows with time and this complexity increases more outdoors.

4.2.4 Indoor Environments

From the perspective of the intended research the localization aspect is focused on indoor environments. The indoors environments can be structured or semi structured with planar surfaces and walled compounds. These environments can be domestic such as houses or compounds such as office buildings, hospitals and museums with rooms and large hallways.

These environments are usually occupied with infrastructure offering lower dynamics with slower walking and driving speeds. The coverage area can be small or large but the size is almost incomparable to a vast outdoor area. Due to the compactness of space and concentration of material, there is a requirement of high precision and accuracy required for localization. Localization becomes more challenging in cluttered environments as compared to an empty room or a hallway. The environments are also dynamic with static and moving obstacles. GPS is not applicable in such environments. The environments that are non-structured, confined and narrow or restricted in space such as underground mines pose an acute challenge to localization. These environments are made of rough surfaces with irregular cross sections restricting the ability to move freely. Some areas consist of moist ground and puddles of mud causing wheel slippage and making odometry unreliable. The presence of darkness and humidity further complicates the scenario. Localization in such kinds of environments either structured or non-structured, is usually achieved by mapping the environment and localize in the map, or making use of indoor geolocalization system such as RFID or Wifi, or recognize natural features such as doors or hallways.

4.2.5 Localization in Industry

The indoor industrial environment is significantly different from the domestic office environments. The offices consist of rooms with furniture and narrow corridors whereas industrial zones such as factories and manufacturing units consists of large spaces and halls housing heavy infrastructure, machinery and equipment. These large spaces are open without having flat walls and corners and have high ceilings. At the same time there are areas consisting of cramped places with material offering limited space to move. The infrastructure present in these environments consist of metallic structures, assembly lines, material delivery conveyors, overhanging cranes, protruding structures of hardware, shelfs, racks and scaffolding with hoses, piping and cabling. The working environment is also sometimes contaminated with oil and liquid spills and debris on the floor. The lighting conditions also produce variation of luminosity and brightness throughout the facility. At the same time the environment is very dynamic with people and AGVs moving around different work zones. Considering the built up and layout of industrial environment with presence of infrastructure, transient obstacles and the working conditions, the localization of an object is not easy aspect to deal with. As already mentioned previously in the context of indoor localization, the issue of industrial localization is solved with the employment of different strategies such as placement of beacons, addition of visual symbols to the environment, use of markers or magnetic strips on floor [79], extraction of natural landmarks in the environment, localizing in a pre-existing 2D or 3D map, or using SLAM during navigation. A substantial amount of research has been done in the past on the localization of a vehicle inside an industrial environment. The scientific community

has dedicated a sufficient amount of work and focus on this issue. The research leverages the use of not a single strategy but the methodologies that have been proposed in the literature make use of a combination of different techniques. This combination of techniques and utilization of a composite architecture depends on the level of complexity of the environment and accuracy required in localization. Below is a brief overview of some of the efforts done in the scope of industrial localization. The preceding sections give a brief overview of the work done in the scope of vehicle localization in industrial environment. The vehicle in scope is either an AGV or autonomous fork lift. Most of the works utilize map-based localization while some make use of the artificial or natural landmarks by extracting the features of the environment.

4.2.6 PAN Robotics

In the context of localization, it is worthwhile to reference here a holistic effort towards the realization of a full-scale autonomy of AGVs in industrial operations. The Plug and Navigation Robots[18] (PAN) for smart factories was dedicated to the development of fully autonomous forklifts for material transportation to introduce higher flexibility, cost efficient logistics, low energy consumption, and enhanced work safety. The project commenced in 2012 and executed for 36 months with 6 European partners. Particularly the work focused on the autonomous onboard planning and navigation for material (pallet) transportation. This consisted of development of a framework of functional elements of localization, navigation, object detection, environment monitoring and fleet management for full autonomy. The localization and navigation framework were implemented by mapping the warehouse environment in 3D by plant exploration, and then defining optimal paths taking into consideration the obstacles and constraints of time and mission completion. 3D stereo vision and laser scanner data was fused to enable pallet detection, perceive overhanging obstacles and monitor environment for blind spots (rack intersections). In a nutshell the goal of PAN was to give the AGVs the perception capability to recognize the work zones and navigate autonomously to transport material from pick to drop off locations.

In the context of PAN the authors in their first effort [80] worked on the localization of an automated forklift in a warehouse based on contour localization. They extract the natural landmarks like posts, racks, corners and walls, by only using the range and angle information of scan data from a laser sensor SICK NAV350. The global localization is achieved by EKF using distance comparison between the landmarks in scan data and in digital map. In their next work [81] they combined the landmark self-localization with grid-based Monte Carlo localization and then implemented SLAM in [82] to account for the dynamic changes. The work consisted of 3D semantic mapping of the warehouse for routing and path planning. The goal was to

18 http://www.pan-robots.eu/

develop a 3D mapping system to overcome the requirement of having manual position measurements of common warehouse elements. A warehouse of the size 200m × 150m was scanned by a forklift truck with SICK NAV350 and SICK LMS500 laser scanners, where the first sensor was used to scan the pre-installed 285 reflectors in the warehouse and second sensor was used to scan the infrastructure to generate a 3D point cloud. The AGV operated automatically in the warehouse for scanning, and local motion was estimated using visual odometry by using Point-to-Line Iterative Closest Point (P-L-ICP). The mapping was acquired by using GMapping and then AMCL in combination with P-L-ICP was used on the map for localization. The installed reflector-based localization system was used as a reference and compared for accuracy of localization. The extension of SLAM in [82] was implemented using two approaches. The first approach was the Robust Pose Graph with the Sparse-ICP algorithm for scan matching and the second was Reflector Feature Graph using the reflector landmarks as feature points. The pipeline consisted of acquiring a 3D map of the environment by using the point cloud data of SICK laser scanner. The point cloud was processed to obtain a semantic map of the environment to record the main infrastructure elements (doors and racks etc). Two laser sensors are used. One fixed sensor for recording a 2D trajectory while the second moving sensor for constructing 3D scans. The 2D trajectory was estimated based on advanced form of ICP. The slam architecture was accomplished by two different graphs. A robust pose graph to represent the global map and another reflector feature graph to extract reflective 3D environment features by filtering 3D scan segments using RANSAC. The features are used for pose graph to obtain global scan matching and loop closure detection.

Apart from PAN robotics in another work, the authors in [83] have worked in the same manner on the global localization of an AGV forklift using artificial landmarks in the environment. The goal is to exploit the pre-existing infrastructure in the warehouse environment containing landmarks. The vehicle is equipped with a laser scanner and a digital map of the environment containing the placement of the landmarks. Global localization problem is solved by matching the detected landmark reflectors to the map and then acquiring the pose of the AGV by triangulation. The localization is done in two steps. First the detected landmarks (reflectors) are matched to the corresponding landmarks in the warehouse map and then the outliers are handled to enforce algorithm convergence. The landmark matching is done by computing the Euclidean distance between the detected landmark and the one in the map. The work concluded in [83] falls very close to the localization approach adopted in this thesis research. The robot localization problem was solved by using the same artificial landmarks detection strategy. The main difference is between the use of the sensor and the implementation of algorithm. A significant fact is that the landmarks detected in [83] are artificial reflectors and do not communicate any specific personal information, whereas in the

approach adopted for this thesis, the landmarks are qualified with specific IDs making them easy to detect and track. Therefore, there is no ambiguity of outliers and no requirement of outlier filtering in the detection and matching process. The robot position is computed with trilateration on detecting a specific number of landmarks. The approach and methodology are discussed in detail in the forthcoming sections.

In another relevant work the authors in [84] worked on the localization of an autonomous forklift in a manufacturing environment. The approach makes use of 2D environment contour-based localization using laser scanners of the forklift. The proposed method was based on an adaptive Monte Carlo Localization using three laser range scanners. Each scanner produced its own individual layer of map acquired during mapping by manual plant exploration. Localizations for each layer was done independently and mobile robot's location was estimated by fusion of the locations acquired by each layer, in order to achieve a more precise and robust location estimate. For mapping each layer GMapping approach was used and AMCL was used in conjunction to localize the vehicle within a layer and compared with marker-based localization as ground truth. The work in [85] presents a localization pipeline providing a sub-centimeter localization accuracy in industrial environments. The approach in the pipeline consisted of the use of AMCL, scan matching and Discrete Fourier Transform (DFT) for high precision indoor localization. The framework was implemented by first mapping the environment using Gmapping and then use AMCL to localize the vehicle in the map. To refine the estimates obtained by AMCL, Scan matching was done to acquire more precision by matching two-point clouds using Iterative Close Point (ICP). The results obtained with AMCL and ICP provide a precise estimate of the orientation of the vehicle. For more precision of the position the result was fed to the DFT. The accuracy in localization was verified in continuous docking operations against a state-of-the-art reflective marker-based system as ground truth. The authors evaluated efficient accuracy results for a travel time of three days of a full size fork lift.

In [86] Lilienthal et al. worked on the localization of an AGV in a warehouse. They proposed a localization method that is able to tolerate changes in the environment. The method makes use of two maps and Monte Carlo localization (particle filter) to track the pose in real time. To achieve localization, the DT-NDT-MCL algorithm makes use of a static map and a short-term map which is updated continuously using Normal Distributions Transform Occupancy (NDT). This approach was employed to use only the best timescale locally and not use the entire timescale map. The algorithm was tested and evaluated on an industrial AGV in use and the authors have shown that the algorithm maintained accuracy in extremely dynamic environments and outperformed commonly used SLAM algorithms. The work presented above mainly focus on the use of particle filters and SLAM combined with landmarks for

localization and mapping. Due to time constraints the mentioning of other approaches is being omitted in this thesis. The various works quoted precedingly report efficient results with good accuracy, however there is requirement of a comprehensive evaluation of the methods based on accuracy and long-term robustness of localization in industry. Although a substantiation of the methods can be acquired based on certain metrics of localization to standardize a method, yet one method cannot be qualified as optimum to satisfy every scenario, since the localization requirement in industrial setting varies with respect to the exact application and requirement. For instance, the global localization of an AGV to get to the next waypoint for navigation is different than the precise localization required for a forklift to align to a pallet location. The localization accuracy is also heavily dependent on the complexity of infrastructure present. In some cases, good accurate 2D localization can be achieved in the presence of a sparse infrastructure where as in other cases the environment needs to be mapped in 3D to extract the features from a dense infrastructure to reduce the ambiguity. The localization approaches are relative and one approach might not be qualified to be optimum over the other since one would prove to be effective for a specific scenario and the other one not. An optimal localization strategy befitting almost every scenario in industry can only be achieved by combining different frameworks but at the cost of computational complexity.

4.2.7 Warehouse environment

Since the research project presented in this thesis involves the localization of the robot inside an industrial warehouse environment, therefore it is imperative to describe the prevailing factors and conditions of such an industrial environment that pose a challenge to localization. The modern logistic warehouse environment is complex. The warehouses are large with a typical size of 6000m^2 with multiple sections (halls) containing dense infrastructure. The infrastructure consists of storage locations with racks, shelfs and containers. Each section contains multiple rows of static racks with a symmetric layout. These racks extend to heights of 25 or 36 feet with multiple levels of storage. Each level stores a specific number of pallets. Apart from static racks, some warehouses also have dynamic storage such as gravity flow and pushback racks. The static layout of the racks forms long aisles or rack corridors with a typical width of 3 to 4 meters. These aisles are used for moving in between the racks and material transportation. These racks are filled with pallets which are extracted and move through these aisles. The structured environment of racks yields perceptual aliasing, since all the racks are of the same type, i.e. dimensions, size and shape. Due to this symmetry of the rack layout, one aisle cannot be clearly distinguished from the other and the situation becomes worse when the racks are empty.

Apart from the racks the infrastructure also consists of overhanging cranes in automatic storage and retrieval systems, metallic structures for safety, conveyer and assembly lines for package transportation, packing and unpacking units, automated loaders and unloaders, sorting systems, trolleys for material delivery and transport and static robots. This density of this infrastructure poses obstructions to movement as well as localization. 3D mapping of dense geometric features results in computational burden and some of the landmark features used for map matching can be occluded by the presence of other materials. The warehouse has some areas which are empty with variations of light in different areas. These spaces consist of long corridors or walkways, big doors and small, docking stations for offloading and loading, and pallet pick up and drop of locations. During the mapping process the walls can only be extracted semantically by fitting vertical planes to the stacks of good and materials stored inside the racks.

The warehouse environment is also dynamic and not static. The warehouse environment configuration is continuously changing due to material transfer flow. If the warehouse is in operation during the mapping process for localization, it is also necessary to handle dynamic motion in the measurements. The dynamic motion of the entities is a consequence of the activities and operations in the warehouse. The activities involve movement of people and vehicles for material transportation. The typical activities involve basic functions of transportation, storage, loading and unloading, picking, handling, packaging, distribution processing, distribution, recycling, and information processing. With respect to the infrastructure the configuration is also continuously changing due to semi-static objects, such as moving cranes, robots, and trolleys that are temporarily placed into the storage area. For this type of environments localization cannot rely solely on a static map. Plant personnel, other manual forklifts or AGVs will interfere with the mapping process and corrupt the 3D scans. Since the environment is continuously changing the map formed at one instant of time needs to account for the changes evolved in the scene. The map needs to be consistent with the static and dynamic modifications. This aspect poses a significant challenge in SLAM and landmark association based localization.

4.2.8 Issue of SLAM

SLAM involves the map building and position extraction at the same time. Prior to the process start up the map and the vehicle location are unknown. The unknown environment is populated with artificial or natural landmarks. The robot updates both the map and position by detecting the landmarks from sensorial measurements. The SLAM is carried out by three main probabilistic techniques all of which are based on the Bayes rule, namely the Extended Kalman Filter (EKF), Particle Filter (PF) and the Expectation Maximization (EM). Although

the SLAM framework has advanced extensively in the last decade, considering the utilization of SLAM in a larger environment such as a warehouse, from algorithmic point of view there are still open issues discussed in detail in [90]. The warehouse environment is large. In order to carry out the mapping process, natural features on observation have to be extracted and stored as landmarks in the map. The EKF is slow in estimating high dimensional maps, since every single sensor measurement affects all the parameters of the gaussian distribution and the update of the map requires disproportionate time when dealing the environment having extensive landmarks. The number of landmarks will go on increasing during mapping and position estimation in a large warehouse, and eventually computational resources will not be sufficient enough to update the map in real time. This computational complexity is due to the fact that computation of landmarks (K) for the sensor updates is quadratic [91] in time. The number of elements in the covariance matrix of Kalman filters is $O(K^2)$ and all of them must be updated on every observation of a landmark. Since every landmark is correlated to others based on the observation measurement and this correlation between the landmarks needs to be maintained for the full duration of the mission. The same is the case with using particle filters (PF) since the state dimension increases as new landmarks are detected thus increasing the computational complexity. Particle filter methodology is effective for localization (detect vehicle pose) and not for mapping (detect landmark pose). Considering the use of EM, this method localizes the robot relative to the current map in a perpetual manner, by generating various correspondences on the possible robot position and then associating those correspondences to the map features to get the precise location of the robot and the most likely map. However, the requirement to process the data repeatedly to obtain the most likely map makes this method inefficient and unsuitable for real-time applications.

The map matching or data association problem which involves the correspondence of features with the observed ones is the most fundamental aspect in SLAM. This feature correspondence involves the correct identification of two features observed at different positions at different instants of time as of being from the same physical object in the world. If the landmark features cannot be uniquely identified, then the number of possible hypothesis grows exponentially which makes the SLAM solution intractable for large areas. The features correspondence is used to match successive scenes as well as to close the loop in SLAM. Loop closure involves the closing of a loop of long trajectory when the robot comes back to the starting point again. The loop-closure problem requires the successful identification of revisited landmarks to build a consistent map in large scale environments. The identification requires correct data association to uniquely validate the landmarks corresponding to previously seen ones, for detecting the loop closure. Considering the large environment of a warehouse, the dense features to map and the perceptual aliasing caused by the symmetry of

the rack's layout and corresponding aisles, loop closure becomes quite challenging to produce a precise and consistent map of the warehouse. Although not implausible, SLAM is still being pursued in the industry for 2D, 3D mapping and localization in large spaces. But the framework for capturing highly precise maps and accurate localization consists of not just one, but a combination of several techniques to achieve the desired result, eventually at the cost of extensive computation. Robots that use embedded systems and dedicated processors cannot use SLAM for covering very large areas. One idea is to use the computing resources utilizing the cloud system to store the maps acquired after exploration in a cloud server and then reuse it again online for map matching and update, but again this depends on the reliability of connection with the cloud and sustainability of the duplex communication. Due to hardware limitations, complexity of computation and limitation of time for the undertaken research, the SLAM framework was not realized and it was decided to adopt a rather simpler approach. This approach was based on the idea of shared intelligence and the process is explained in detail in the subsequent sections.

4.2.9 Localization Methodology

Bringing into perspective the scenario presented in section 4.1 which gives a global overview of the operation to be executed, for the first phase the robot needs to localize itself inside the warehouse prior to navigation. This localization process is cyclic and the robot needs to localize itself before the movement to have an absolute position reference, during the navigation to keep track of its position by localizing in regular intervals, and on reaching the goal point to correctly place itself at the operating point (close to the pallet). The localization required before navigation is global and is required to give the robot a general idea where it is placed inside the warehouse. This localization does not require to be very precise, whereas localization required at the operating point is local and is combined with the "Hough transform" navigation strategy to localize the robot precisely next to pallet. The "Hough transform" navigation strategy is explained in detail in the navigation section.

For the localization framework of this research, the concept of shared intelligence was exploited. The idea is based on the concept of intelligent environment or intelligent spaces [92] in which sensors and information systems are embedded into the environment to facilitate the users in both physical and informative way. A prime feature offered by these intelligent spaces is the ability to track the position of objects in space. For the tracking a variety of sensors and passive transponders are installed in the environment. Some common sensors are cameras, ultrasound or electromagnetic transmitters and receivers, pressure sensors, laser range finders and passive reflectors. The passive reflectors convey encoded information on detection. To employ the framework of shared intelligence the distribution of these sensors can be alternate.

For example, a laser scanner can be installed on the robot to track the passive reflectors in the environment and vice a versa. Considering the warehouse environment and presence of huge infrastructure, for localization the strategy to utilize artificial landmarks was employed while obeying the constraints of simplicity and minimum infrastructure change. Hence leveraging both the intelligence of the platform and the environment to localize the robot in the warehouse by detecting these artificial landmarks.

Considering the use of landmarks for localization, the strategy is not an uncommon one. Numerous works in the research have taken advantage of this strategy for indoor/outdoor vehicular localization. The authors in [93] have worked on the localization of a robot from natural landmarks given a global map of the environment. The map of the environment contains the position of the landmarks and the approach consists of two localization algorithms. The first strategy is based on triangulation where the position of the robot is computed by measuring the relative bearing and computing circle intersection followed distance measurement to the landmarks and solving a system of non-linear equations. To avoid solving non-linear equations, the second method is based on the complex number representation of landmarks to get a set of linear equations whose solution is a set of position estimates. Further the authors in [94] estimate the position of the robot by triangulation from the landmarks. The bearings from the landmarks are treated as a constraint on the position and orientation of the robot. Given three constraints (three angles) and using SVD the linear system is solved to get the solution giving the estimate of the robot's position. The accuracy is further improved by catering for the non-optimal solution acquired by SVD and applying several transformations to the original linear system and an optimization procedure to yield more accurate results. The authors in [95] have worked on the localization of the robot from three landmarks for a successful navigation of the robot. The method consists of the utilization of two control laws which use the bearing information from the landmarks to compute the resultant vector that drives the robot to the goal location. Based on geometric observations about the perceived angles, a hybrid system was built to combine the reachability sets of the two control laws. The execution of controls was managed by a state machine that selects the appropriate control to drive the robot to the goal. The work in [96] consists of the indoor localization method based on natural landmarks and computation of robot position by triangulation. The natural landmarks are extracted from panoramic images and the positions of the landmarks in the map are identified by comparing the input image and reference data set. The landmarks bearings and positions are extracted and robot position is triangulated.

Another work on vision-based localization [97] involving natural landmarks consists of using 3D landmarks in an a-priori obtained accurate 3D map of the environment. For localization the feature visibility is determined by associating the image features with the map

elements through the use of geometric relations between the 3D map and the camera pose in the matching process. The visibility of every map feature is modeled with respect to the camera pose via non-parametric distributions. These distributions are learnt during the 3D reconstruction process and are used to predict the visibility of features during localization. The map features with the highest visibility score are used in the matching process for a qualitative localization. In [98] the authors have worked on a robust indoor localization system for mobile robots. Their approach is based on the triangulation method by measuring the bearing from artificially placed retroreflective landmarks with a laser sensor. The overall localization method involves the resolution of problems which result in the failure of triangulation in general. This involved the calibration of the sensors to produce unbiased readings, landmarks calibration for faulty positions, efficient detection of misidentified landmarks by using RANSACK, and localization during motion of the robot. The calibration of landmarks consisted of calculating the positions of landmarks minimizing the MSE over a set of measurements at different locations and orientations of the robot, while the effects of translation and rotation on sensor measurement were modelled to achieve accurate localization during motion. The framework was implemented and tested on a real robot equipped with a laser scanner.

The authors in [99] estimate a non-holonomic robot's speed and position with only one landmark by measuring the position and relative angle of the landmark. They treat the localization as a partial observability problem of the state of the robot and compare the results of localization with the traditional EKF method. They have reported a variation of effectiveness of the proposed method with respect to the number of landmarks and computation time. The work in [100] presents the use of magnetic sensors as artificial landmarks for the localization of an AGV. The magnetic sensors are automatically able to connect to the passing mobile robots and exchange information for localization in the form of Transducer Electronic Data Sheet (TEDS) stored on their memory. The magnetic sensors consist of two parts, with a transmitter on the mobile robot platform and a receiver on the artificial landmark. The description and location of the landmarks is stored as metadata (TEDS) and landmark localization is handled by Robot Operating System (ROS), in which the controller on board the robot calculates its own absolute position taking int account the uncertainty of the sensor. The work in [101] gives the optimal placement of the landmarks based on machine learning producing an optimal layout of landmarks for arbitrary environments. Similarly, the work in [102] gives the optimal placement of the landmarks based on finding the distribution of landmarks which guarantee a bound on the maximum deviation of the robot from a desired trajectory.

4.2.9.1 Artoolkit Library

The artificial landmarks are installed throughout the warehouse environment. The entity used as artificial landmarks are vision based Augmented Reality (AR) markers. These markers can be reproduced in paper printed form and can be installed in the environment easily in a short time. The detection and pose estimation of the markers is performed by the open source library ARtoolkit[19]. ARtoolkit is a C/C++ software library used for developing augmented reality applications. It uses computer vision algorithms to compute the camera position and orientation relative to the marker and overlay virtual information over the detected marker. The toolkit is based on the corner detection approach for pose estimation. Each image from the video frame is thresholded (converted to binary). Then a search for contours is done. The contours are combined to search for edges and corners. If a square is formed by these edges, then the sub image within this square is compared by template matching with a pattern stored beforehand to get the marker's 3D position and orientation. The position and pose of the marker are calculated from the transformation from marker's coordinates to camera coordinates. The pattern of the marker corresponds to a specific ID. Since the complete functional framework of this research has been implemented using Robot Operating System (ROS), a more advanced version of ARtoolkit called ALVAR[20] and its extension 'ar_track_alvar[21]' specifically developed for ROS, was used for the detection of markers. On detection the virtual marker is overlaid at the same position as the detected marker in the ROS visualization environment.

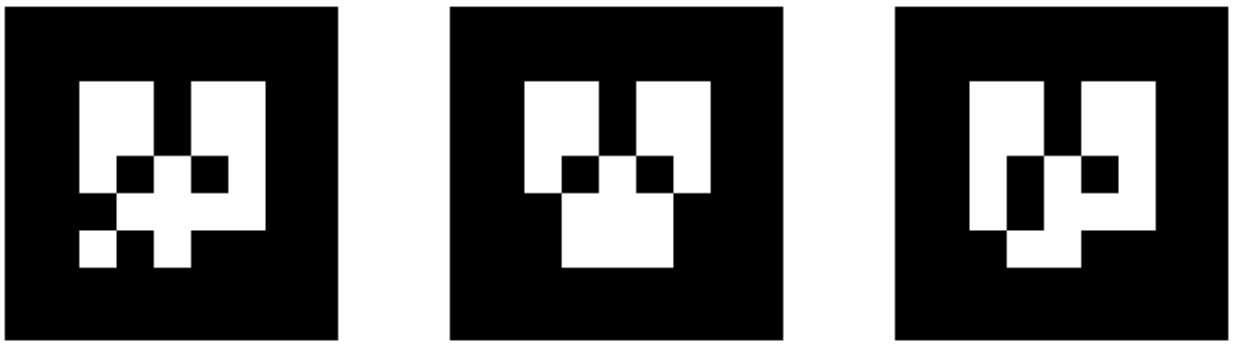


Figure 4.1 : Augmented Reality Markers

With the ar_track_alvar different AR tags/markers can be generated with variable size, resolution and encoded data ID. The range of the size is from 5-20 cm and the user ID can be a string or a number from 0-255. The marker can be detected using both a 3D depth camera and RGB camera. There is also another alternative extension 'april_tag[21]' for ROS which can be used to generate tags with a higher number than 255.

19 http://www.hitl.washington.edu/artoolkit/

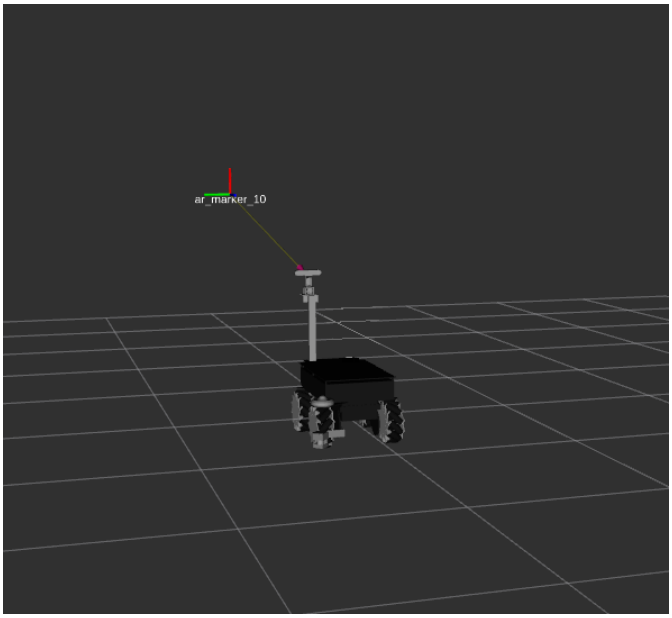


Figure 4.2 : Marker Detection in ROS

4.2.9.2 Vision Sensor Selection

The marker can be detected by both the 3D depth camera and the RGB camera. The robot is equipped with both the types and is able to detect and extract its position with respect to the marker in the robot frame. For the localization it is required to have an idea of the placement of the markers. The markers cannot be installed in a random fashion inside the warehouse. The placement has to follow an appropriate strategy. A specific approach would be to place the marker at equal distances at designated places. But this distance threshold has to be under the detection range of the vision sensors to be used.

4.2.9.2.1 Experiment 1

To find out this threshold i.e. the maximum detection range of any of the vision sensors used on the robot, a simple experiment was performed. The experiment yielded both the maximum detection range and quality of detection of the RGB camera (Axis) and 3D depth camera (Ptu). The experiment was executed as follows.

Setup : The robot was placed at a distance of 1 meter while facing the wall. The distance was measured from the wall with the Lidar from the center of the robot. This distance was taken as the ground truth for comparison and quantitative analysis. A marker was installed on the wall at a height of 1.45 meter from the ground. The robot was aligned with the center of the marker.

20 http://virtual.vtt.fi/virtual/proj2/multimedia/index.html

21 http://wiki.ros.org/ar_track_alvar

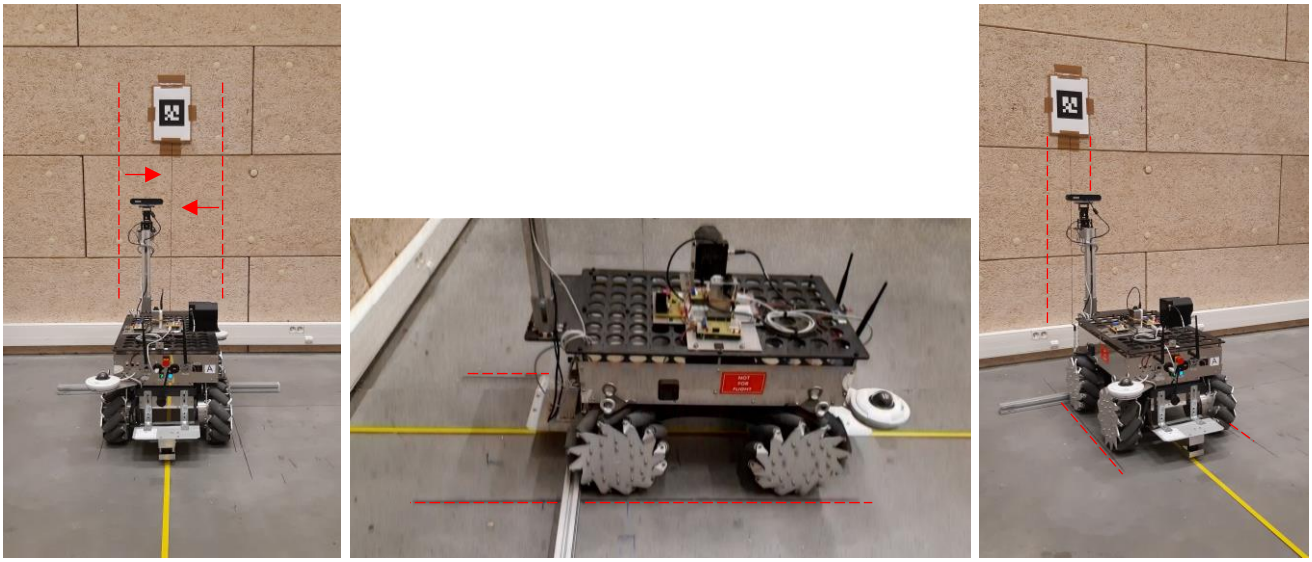


Figure 4.3 : Robot Placement for Experimental Setup

Camera Parameters

Camera	Tilt Angle (Deg)	Pan Angle (Deg)	Zoom	Marker ID
PTU Camera	0	0	0	10
Axis Camera	25	-25	0	10

Table 2 : Camera Parameters Set for the Experiment 1

Robot Parameters

Speed (m/s)	Start Distance from wall (m)	Stop Distance (m)
0.1	1	10
0.3	1	10
0.5	1	10
0.7	1	10
1.0	1	10

Table 3 : Robot Parameters Set for the Experiment 1

Figure above shows the centering of the robot with respect to the marker manually. The tables '2' and '3' show the parameters set for the experiment. The cameras were calibrated and the positions were fixed to face the marker while taking into view the rectified image.

Procedure : The detection for both the cameras with the 'ar_track_alvar' was enabled and the robot was moved in reverse with constant velocity. The robot was moved for a distance of 10 meters at constant speeds of 0.1 m/s to 1.0 m/s. The distance threshold of 10 meters was specified based on the available space in the testing facility. The cameras detected the marker during the continuous motion and the data was recorded. The detection of the marker by the cameras gives pose coordinates of the robot relative to the marker. These coordinates give the

relative 3D distance of the robot from the marker which were compared with the ground truth to extract the maximum range. The data for all the speeds in the table was recorded, but due to the brevity of time and space only the results at the speeds of 0.3 m/s and 1.0 m/s are given here. The speed 0.3 m/s is the standard speed for all the navigation tests whereas 1.0 m/s was tested to validate the robustness of detection at high speeds, even though the robot never moves at this speed for generic navigation.

Conclusion : As shown in the below figure 'D_Lidar' is the distance given by the Laser in the center of the robot. From the height 'H' and 'DLidar' the 3D Lidar distance or the ray 'D_Lidar_3D' is computed and the distance for the RGB and 3D cameras is computed as

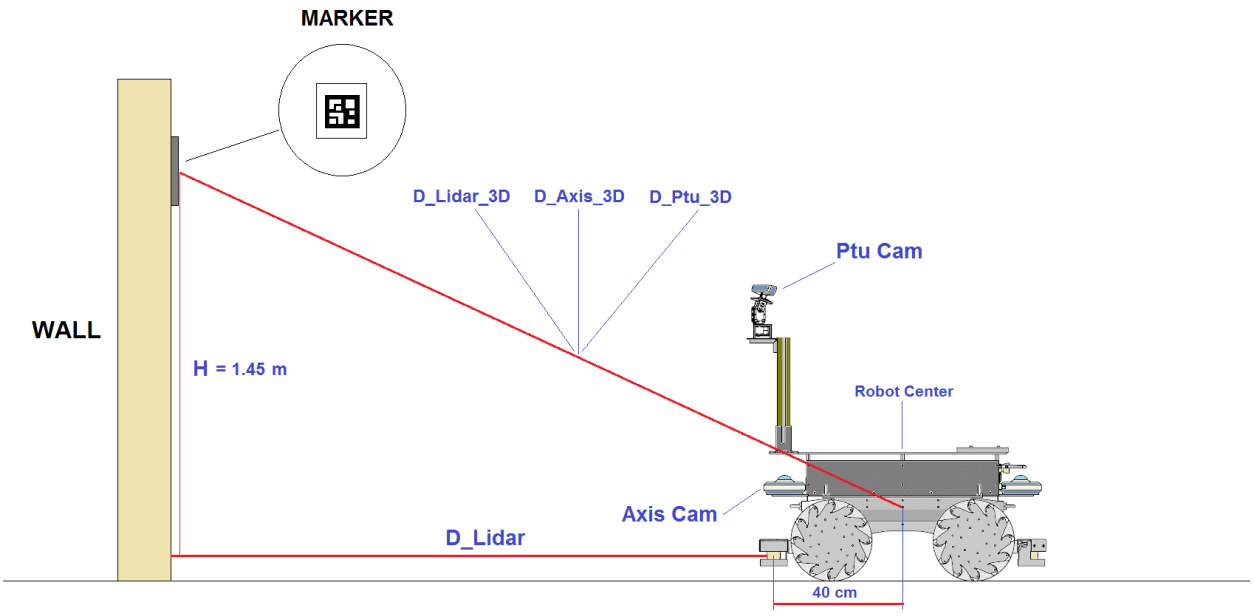


Figure 4.4 : Robot Lidar Distances

$$D\_Lidar\_3D = \sqrt{H^2 + D\_Lidar^2} \tag{4.1}$$

$$D\_Axis\_3D = \sqrt{pos\_mrkr\_x^2 + pos\_mrkr\_y^2 + pos\_mrkr\_z^2} \tag{4.2}$$

$$D\_Ptu\_3D = \sqrt{pos\_mrkr\_x^2 + pos\_mrkr\_y^2 + pos\_mrkr\_z^2} \tag{4.3}$$

where the variables $'pos\_mrkr\_x, pos\_mrkr\_y, pos\_mrkr\_z'$ give the pose coordinates of the robot from the marker expressed in the center of the robot frame. The robot frame coincides with the 'Robot Center' as shown in the figure above. After computing the distances, the error for each camera was computed by subtracting the distances from the ground truth as

$$Error\_Axis = abs(D\_Lidar\_3D - D\_Axis\_3D) \quad (4.5)$$

$$Error\_Ptu = abs(D\_Lidar\_3D - D\_Ptu\_3D) \quad (4.6)$$

The abbreviations 'Axis' and 'Ptu' refer to the RGB and 3D depth camera respectively. The recorded data was extracted in Matlab and the respective distances were plotted to quantify the error and extract the maximum range.

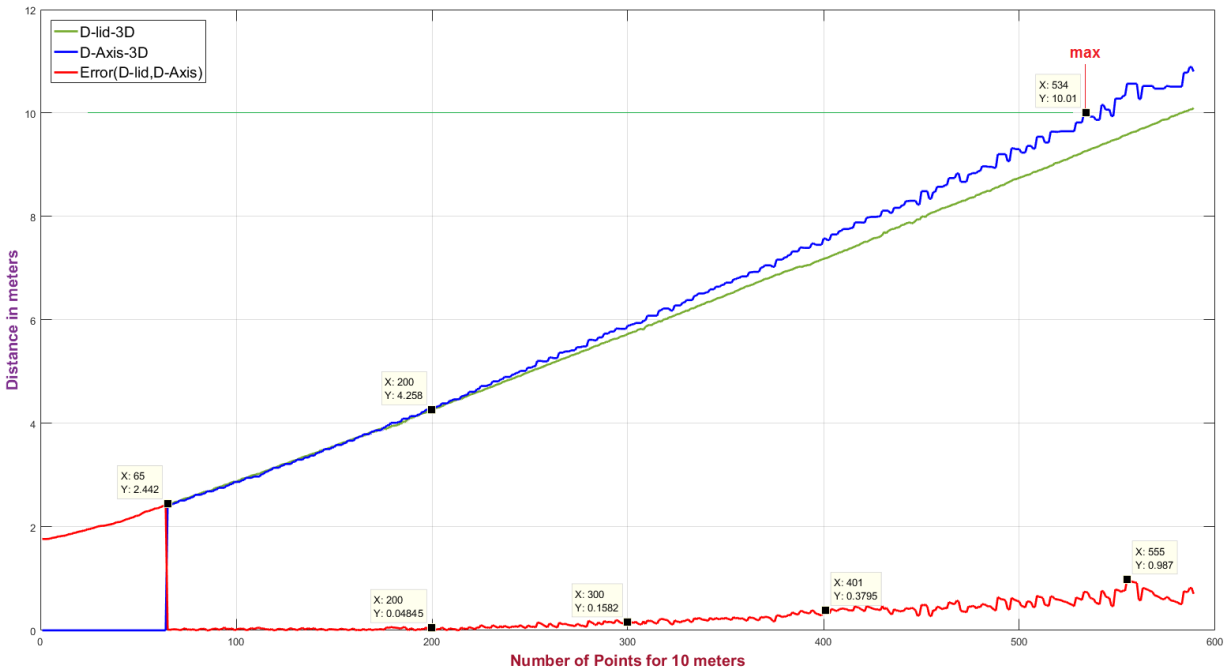


Figure 4.5 : Marker Detection Axis Cam at Speed of 0.3 m/s

The above figure gives an over view of the detection of marker by the RGB Axis camera. It can be seen that the error (in red) starts to increase initially till a distance of '2.4' meters and the detection (detected distance in blue) is zero. This is the time when the robot starts to move away from the wall and the marker is not in the view of the camera. As soon as the marker comes into perspective at 2.4 meters, the camera is able to detect it and the error reduces to zero since the detected distance by the marker is almost equal to the ground truth (Lidar distance). This is evident by the overlapping of the green and blue curves till the data point 200. As the robot moves away further, the blue curve deviates from the ground truth and the error also increases. The error increases due to the increase in distance from the marker and also due to other uncertainties not accounted for in the whole setup. Despite the error, it can be seen that the camera is still able to detect the marker beyond the distance of 10 meters (max). The maximum but logical error after this range is almost 1 meter. In the figure below, showing the detection of the marker by the 'Ptu', the detection also starts at 2 meters when the marker comes into perspective. It is evident from the plot that the 3D camera is able to

detect this marker up to the range of 6.5 meters. Beyond this range the blue curves becomes constant due to the fact that after this distance, the camera is not detecting new coordinates for the pose of the robot, but giving out previous coordinates detected at 6.5 meters. This is the inherent feature of 'ar_track_alvar'. After this distance the error also starts to increase due to difference between the ground truth and the constant curve.

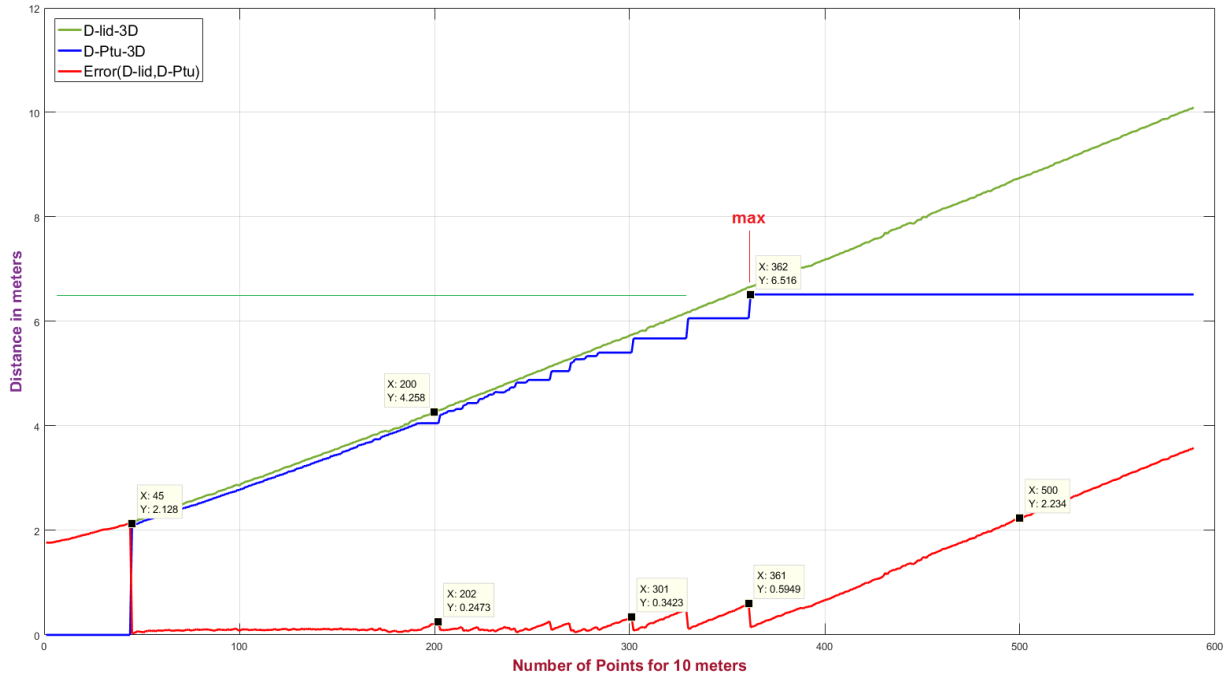


Figure 4.6 : Marker Detection Ptu Cam at Speed of 0.3 m/s

The figure (4.7) below shows a comparison of the two errors of the RGB and 3D camera. The errors are well below 30 cm for a distance of 5.5 meters. The error for the RGB axis camera (red curve) remains bounded below 30 cm and the maximum errors is close to 1 meter. Both the cameras are able to give a good pose of the robot till a distance of almost 4 meters which is visible in the plot at data point of 200. The figures (4.8) and (4.9) below is for the RGB and 3D depth camera for the speed of 1.0 m/s. It is visible from the figure (4.8) that the ground truth(green) is not consistent. This is due to slight vibrations induced in the lidar housing by the meccanum wheels when the robot moves at high speed. The RGB camera is still able to detect the marker but the detection stops at some intervals. This can be seen from the red interval after data point 122 (fig 4.8). This is due to the fact that at high speeds the marker is moving too quick out of perspective of the camera than its frame rate.

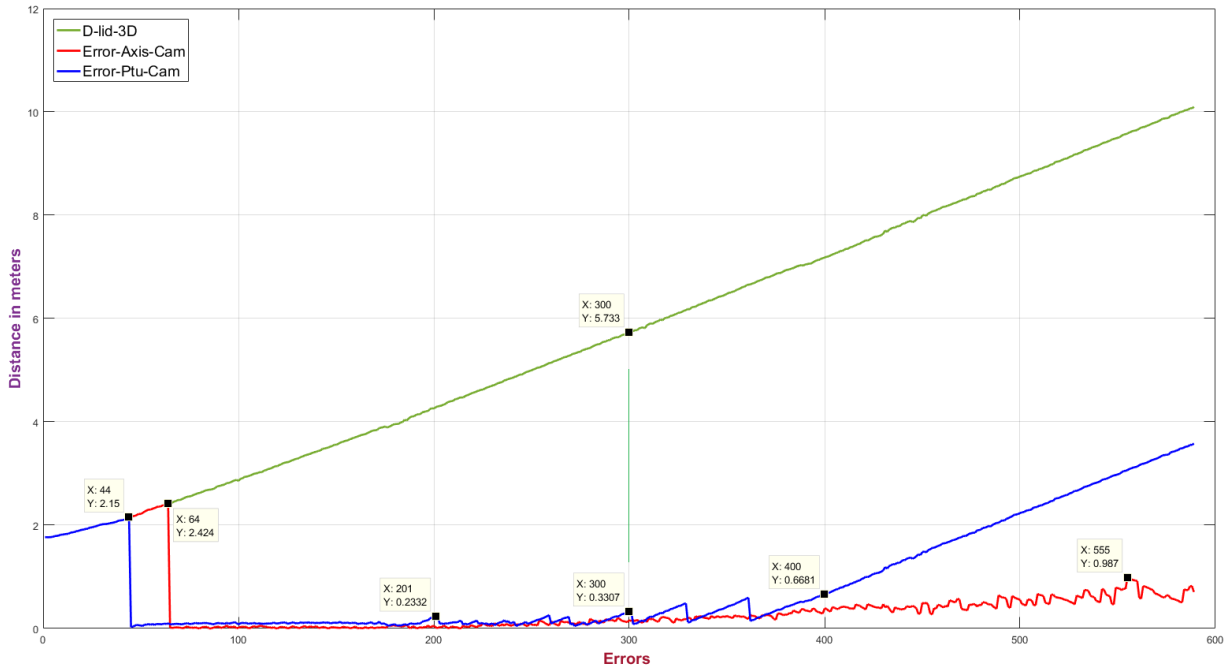


Figure 4.7 : Errors for Axis Cam (RGB) and 3D Depth Cam

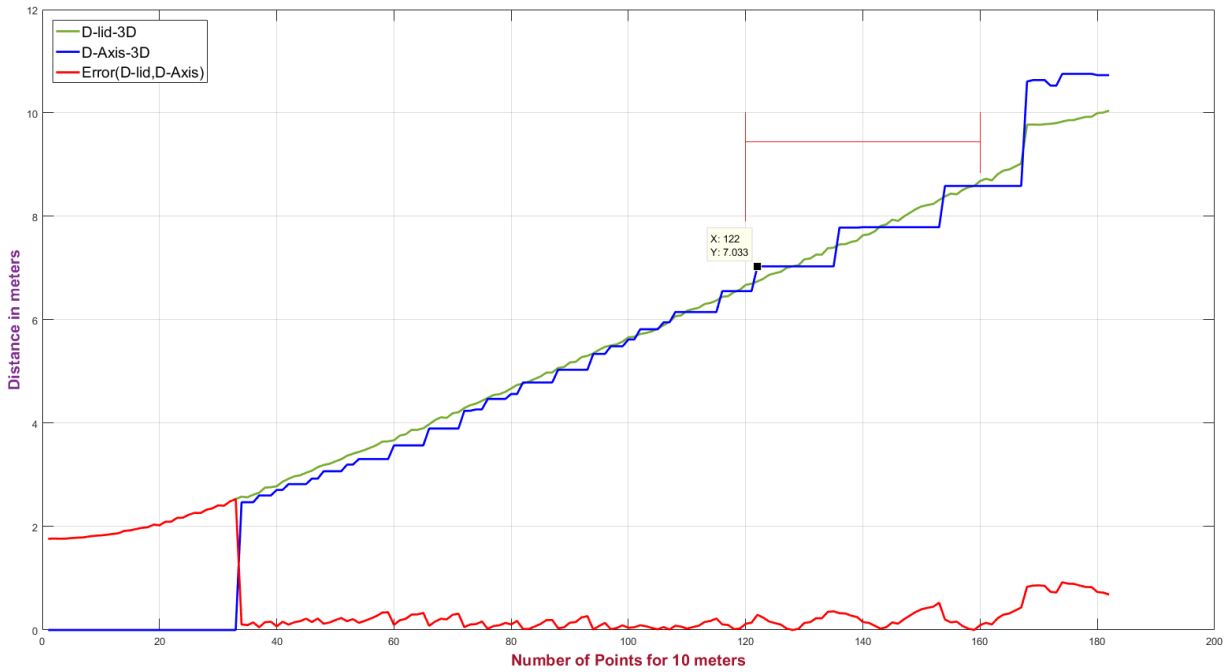


Figure 4.8 : Marker Detection by Axis Camera at Speed of 1.0 m/s

The situation is the same for the 'Ptu' camera above. The curve levels off at a distance of 3.4 meters which shows the detection stopped after 3 meters. The error of the Axis camera remains bounded initially for almost 5 meters while even moving at high speeds. From the errors plots and detection performance of both the cameras it was concluded that the Axis camera

performed better in detection of the marker than the 'Ptu' for a range of 10 meters. From the sensor selection test, the Axis camera was selected for the detection of the markers. The camera was able to detect the markers at a distance of 10 meters. This distance was selected as the threshold for the placement of the markers. For localization at larger distances outside the racks the inter-marker distance was specified as 10 meters whereas inside the racks the markers were placed at a distance of 3.55 meters.

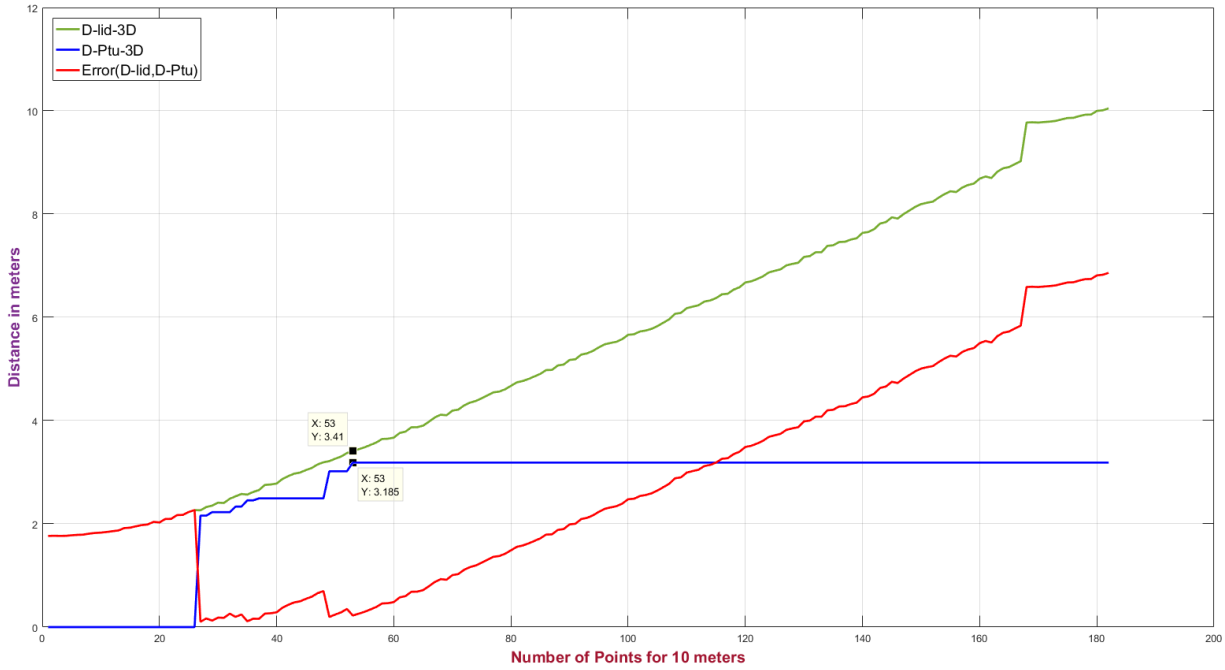


Figure 4.9 : Marker Detection by Ptu at Speed 1.0 m/s

4.2.9.3 Landmarks Placement

For picking operations the robot needs to know the pallet location. This requires localization of the robot with respect to the pallet locations inside the racks. The robot requires to ascertain its current location to go to the right picking location. This is accomplished by localization with the help of markers installed inside the racks at specified distances determined from the previous experiment. For the efficacy of localization process, it is necessary to validate the detection of markers during the stationary and mobile modes of the robot.

4.2.9.3.1 Experiment 2

Setup : For localization inside the racks another test was carried out in the facility environment. A mockup of warehouse environment was setup and markers were placed on the pillars of the racks at a distance of 3.5 meters as seen in the figure below. The infrastructure

arrangement consisted of a standard conventional storage rack with pallets and boxes, whereas 3 synthetic pillars were placed on opposite side of the rack pillars to simulate an opposite rack. The width of the corridor formed by the rack pillars for navigation was 3.0 meters. The robot is equipped with two RGB cameras (Axis) on the front and rear sides as shown in the figure. Based on the previous experiment these cameras were used for the detection of the markers.

Figure 4.10 : Robot Placement for Experiment Setup

Camera Parameters

Camera	Tilt Angle (Deg)	Pan Angle (Deg)	Zoom	Marker ID
Axis Camera Front	30	25	0	11,12,26
Axis Camera Rear	30	25	0	3,4,5

Table 4 : Camera Parameters Set for Experiment 2

Robot Parameters

Speed (m/s)	Start Distance from wall (m)	Side Distance from Rack (m)
0.1	10	1.0 ,1.5, 2.0
0.3	10	1.0, 1.5, 2.0
0.5	10	1.0, 1.5, 2.0
0.7	10	1.0, 1.5, 2.0
1.0	10	1.0, 1.5, 2.0

Table 5 : Robot Parameters Set for Experiment 2

Procedure : The robot was placed at a distance of 10 meters from a wall at front and at distances of 1.0,1.5 and 2.0 meters from the side of the rack. The distances from the wall and side of the rack were manually measured and verified by the lidar (laser) from the robot, and then marked on the floor. For acquiring the camera view angles for a good marker detection, the robot was placed in the center of two pillars diagonally at equal distances to view the markers in the center of the image acquired by the cameras. The detection was enabled and the camera view angles were fixed. The robot was then placed each time at the start of rack at the specified distance from the side of the rack and moved for a distance of 10 meters at speeds

of 0.1 m/s-1.0 m/s. The pose of the marker was detected in the robot frame giving the distance between the robot and the marker and the coordinates were saved. The lidar distance from the front wall and the side of the rack was measured and recorded during the movement of the robot.

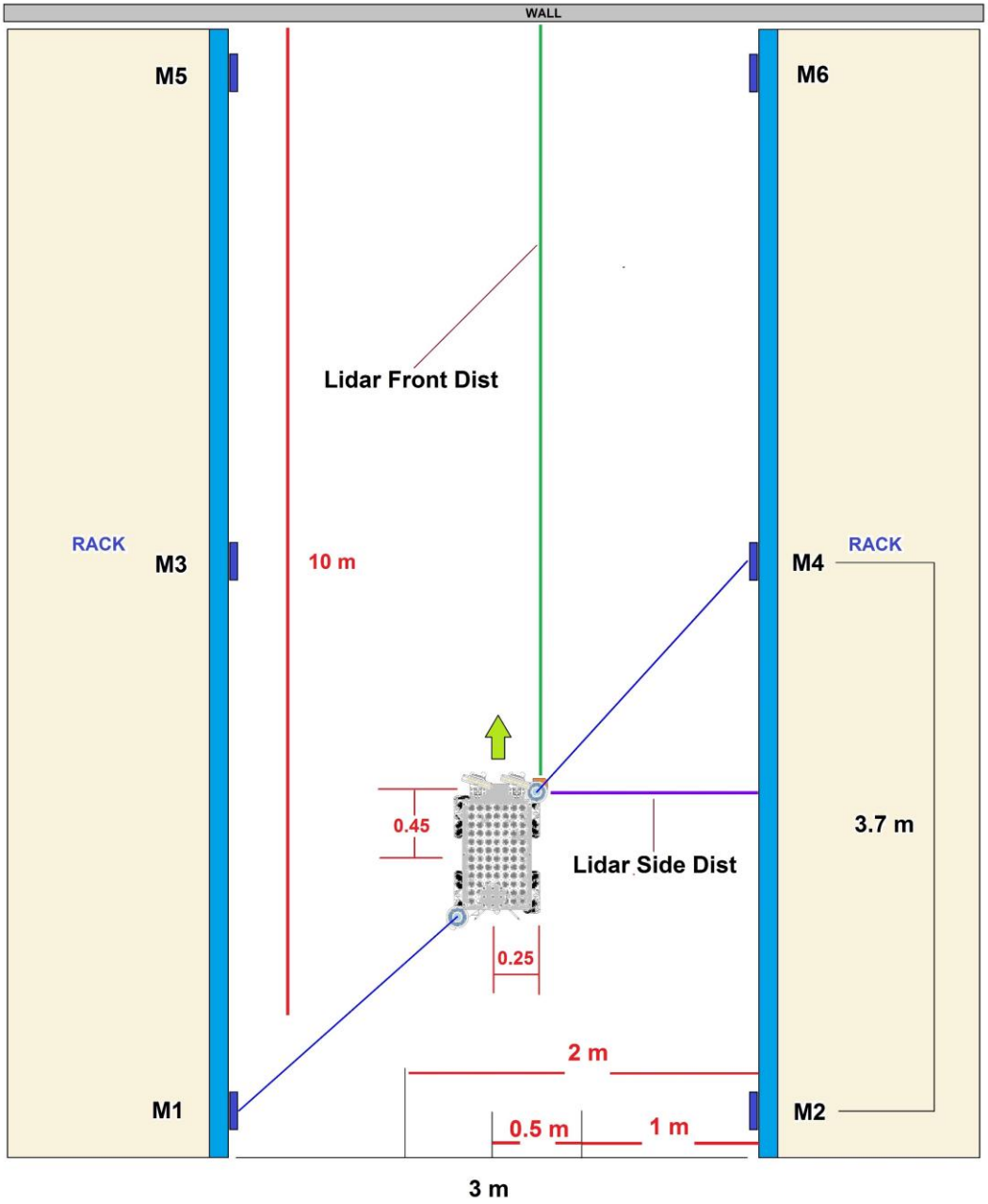


Figure 4.11 : Robot Top View Placement

Conclusion : As shown in the figure below 'D_lid_wall' is the distance given by the Laser from the wall in the center of the robot. The distance 'D_lid_rack' is the right-side laser distance from the rack when robot is facing the wall in fig (4.11). From the height 'H' and 'D_lid_rack' the 3D Lidar distance or the ray 'D_3D_rack' is computed. This distance is taken

as the ground truth. The distance from the pose of the marker is given by 'D_marker' and is computed as

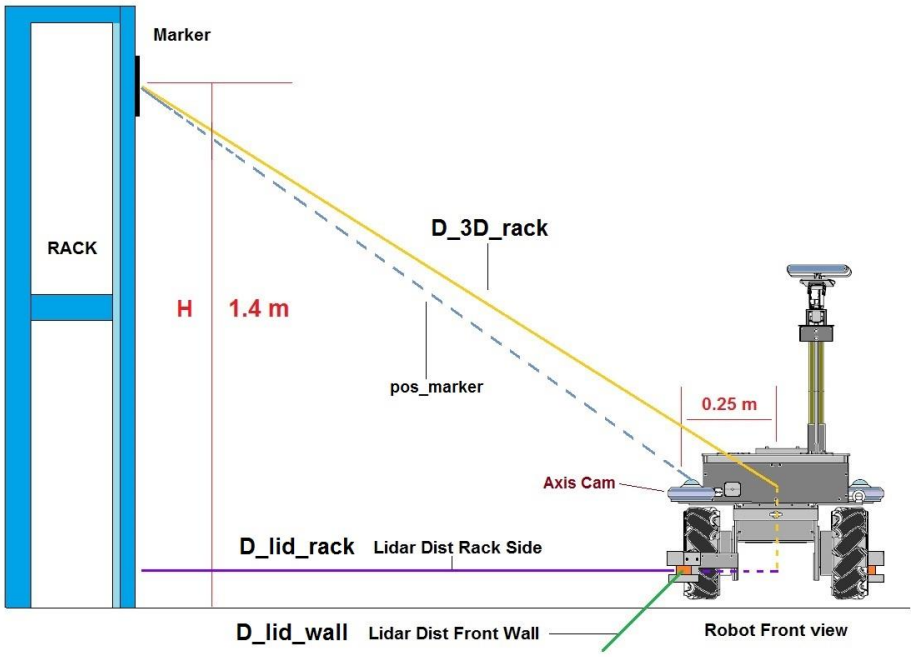


Figure 4.12 : Robot Front View Placement

$$D\_3D\_rack = \sqrt{H^2 + D\_lid\_wall^2} \tag{4.7}$$

$$D\_marker = \sqrt{pos\_mrkr\_x^2 + pos\_mrkr\_y^2 + pos\_mrkr\_z^2} \tag{4.8}$$

where the variables $'pos\_mrkr\_x, pos\_mrkr\_y, pos\_mrkr\_z'$ give the pose coordinates of the robot from the marker expressed in the center of the robot frame. The robot frame coincides with the 'Robot Center' as shown in the figure above. After computing the distances, the error between the 3D rack distance 'D_3D_rack' and the 3D distance given by the marker is computed

$$Error\_3D = abs(D\_3D\_rack - D\_marker) \tag{4.9}$$

The data acquired from the lidar and marker detection was saved and plots were acquired in Matlab. The detection of the marker was observed from the plots. The figure below presents the data for marker detection by the front Axis camera at the speed of 0.5 m/s when the robot was placed at a distance of 1.5 m from the rack (in the center of the corridor).

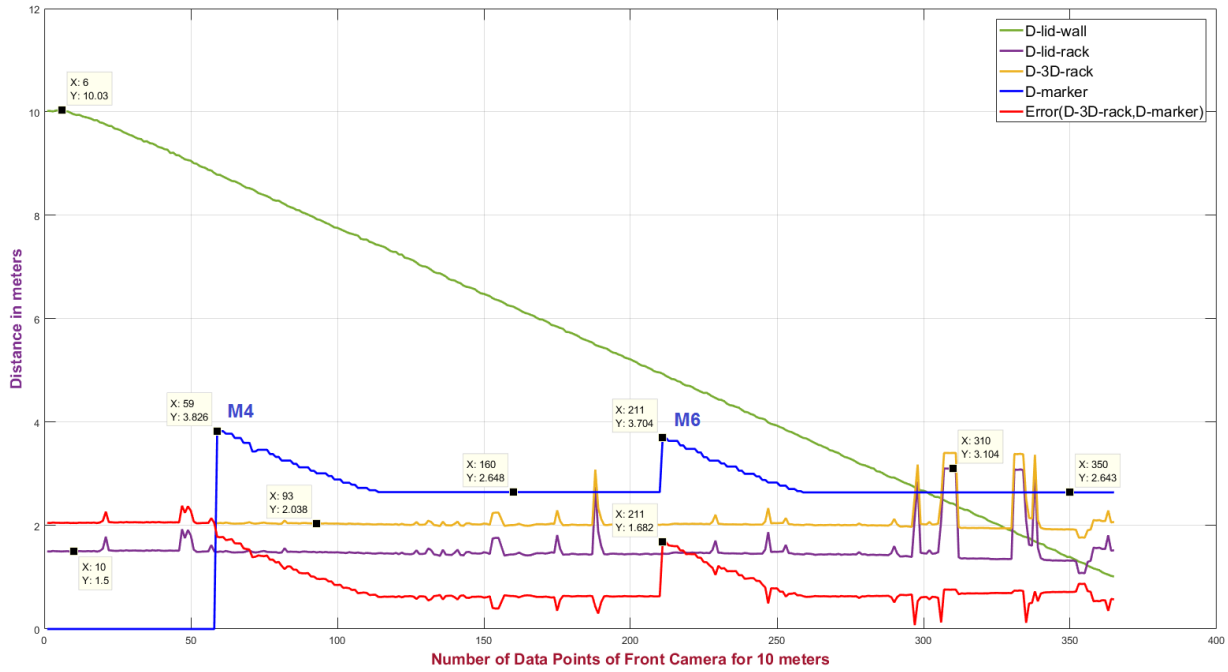


Figure 4.13 : Marker Detection By Front Axis Cam at Speed 0.5 m/s

In the figure the x-axis presents the number of data points for the trial run and the y-axis gives the distance in meters. The green curve gives the lidar distance to the wall. The blue curve is the detected marker distance from the robot. The orange curve is the 3D lidar distance (ground truth) as shown in the figure. The magenta curve is the side distance of the robot from the rack. In the plot it is visible that the first marker 'M4' is detected after data point '50'. The blue curve abruptly goes up showing an increase of distance from zero. This is the point when the coordinates of the marker are detected and distance between the robot and marker is computed. This distance gradually decreases as the robot approaches close to the marker. This can be seen clearly as the blue curve descends and then becomes constant. The constant curve shows the cessation of marker detection, and the coordinates saved are the last ones from the previous detection. This is an inherent feature of the 'ar_alvar_track' library. The second marker 'M6' is detected after data point '200'. The detection of the marker is also evident from the error between the ground truth and the detected distance. Initially the error and the lidar 3D distance are equal but this error decreases as soon as the marker is detected and becomes closer to the robot during motion. The green curve gradually descends showing the robot approaching the wall and stops at 1 meter. The sharp peaks in the lidar side distance (magenta) and the lidar 3D (orange) distance correspond to the instances, when the beam of the lidar goes through the hollow sections of the pallets and open spaces and hits the wall behind the rack.

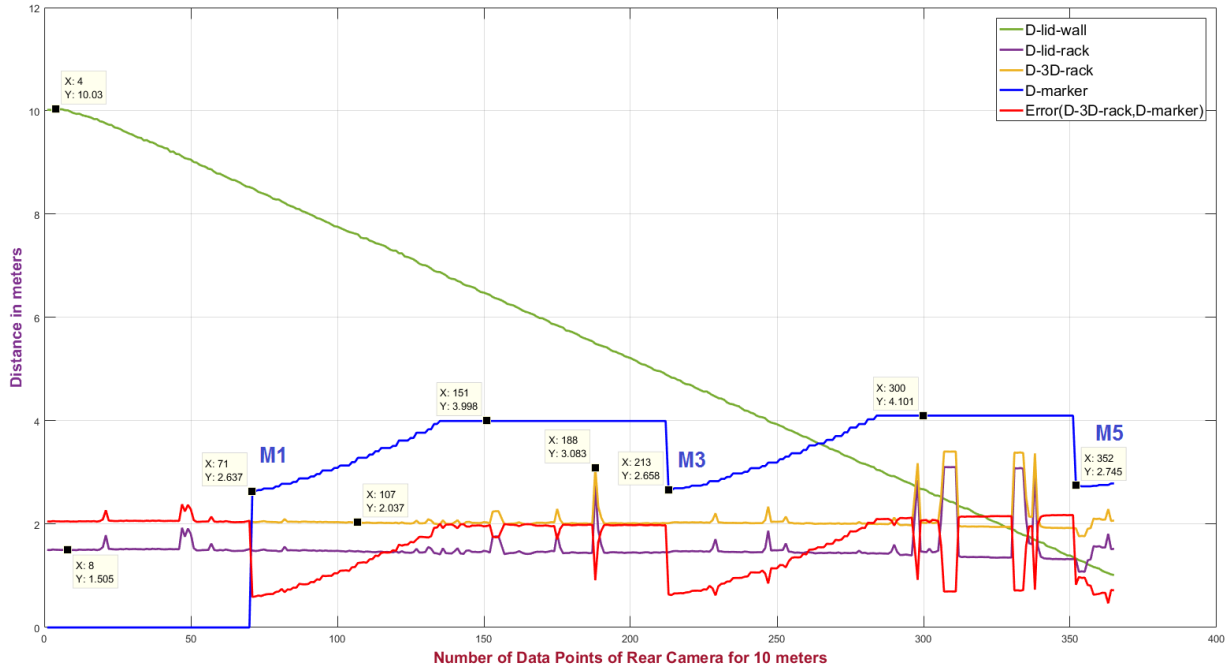


Figure 4.14 : Marker Detection By Rear Axis Cam at Speed 0.5 m/s

A similar picture can be seen in the plot above for the rear camera. The marker 'M1' is detected after data point 50 and is at a distance of 2.023 meters from the robot center. This distance starts to increase gradually since the robot is moving away from the marker. This can be seen from data points '71-135' (x-axis). The curve becomes constant afterwards showing the cessation of marker detection since the marker goes out of the perspective of the camera. The second marker 'M3' is detected at data point '213'. The curve falls downwards showing the detection of the marker based on the computed distance between the robot and the marker. Here a fact to note is that the error curve follows the behavior of the detection (blue) curve. The third marker 'M5' is detected when the robot just stops close to the wall. It can be seen from both the plots, that for the front camera the detected markers give a distance of 3.7-3.8 meters and the for the rear camera the detected distance is 2.6-2.7 meters. This shows the consistency of the robot position from the side of markers during movement in a straight line towards the wall.

The figure below shows the plots for the three speeds of the robot at 0.1 m/s, 0.5 ms/s and 1.0 m/s at a distance of 1.5 meters from the rack. The detection is very smooth at a speed of 0.1 m/s however the cameras are able to detect the markers even at a high speed of 1.0 m/s.

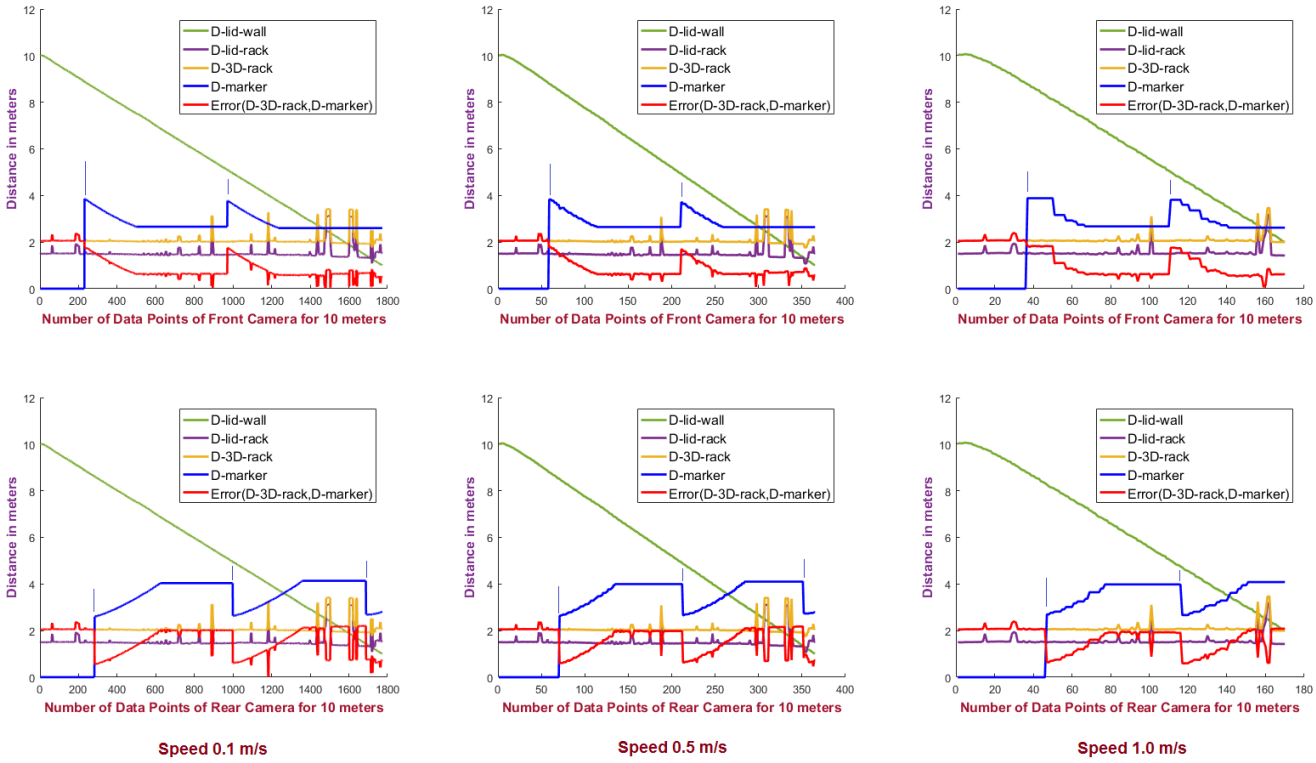


Figure 4.15 : Marker Detection by Front and Rear Cameras at 1.5 meters

The figure below shows the plots for the three speeds of the robot at 0.1 m/s, 0.5 ms/s and 1.0 m/s at a distance of 1.0 meters from the rack. The robot is quite close to the rack in this case, the front camera is able to detect the marker but the variation in distance is very small as seen in the upper left and middle plots. The marker detection is shown by tick marks, and the very small peaks correspond to the fact that since the robot is close to the rack, for the front camera the angle of view is fixed and the marker comes into perspective for a very short time. Same is the case at the speed of 1.0 m/s, the front camera is able to detect only one marker. The rear camera is able to detect all the markers on its side (opposite side of the rack) which can be seen by the significant variation of distance (high peaks). The situation is similar in the below figure for the plots of different speeds at a distance of 2 meters from the rack. The distance of markers from the front camera is 2 meters where as the distance for the rear camera is 1 meter. The plots for the rear camera (bottom figure) show a similar situation to the plots for the front camear in figure (4.16). From the plots of the front and rear cameras of figure (4.16) and (4.17), it is clear that the cameras are able to detect the markers ever at high speed, inspite of having a short time exposure to the marker. This observation leads to the conclusion that when the robot is quite close to the rack, a much higher tilt angle of the camera can result in better detection of the markers, even at high speeds.

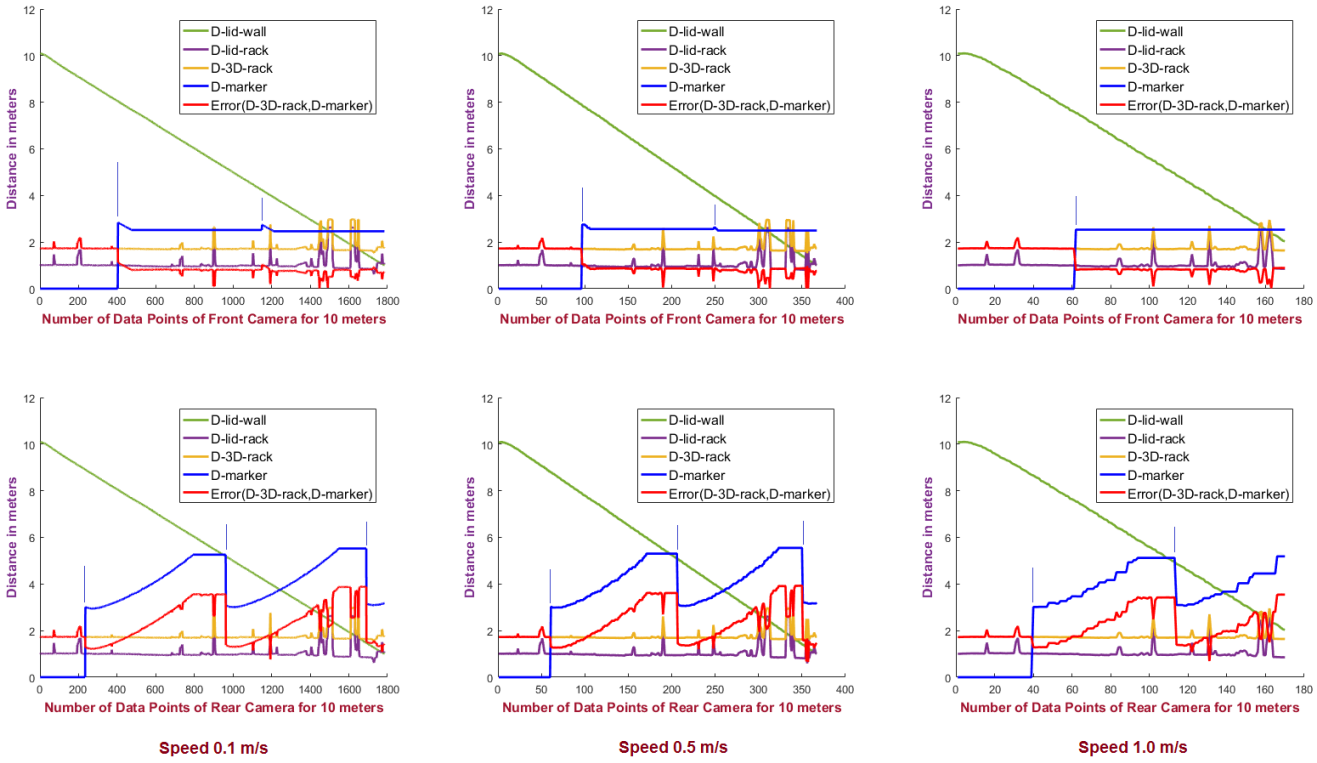


Figure 4.16 : Marker Detection by Front and Rear Cameras at 1.0 meters

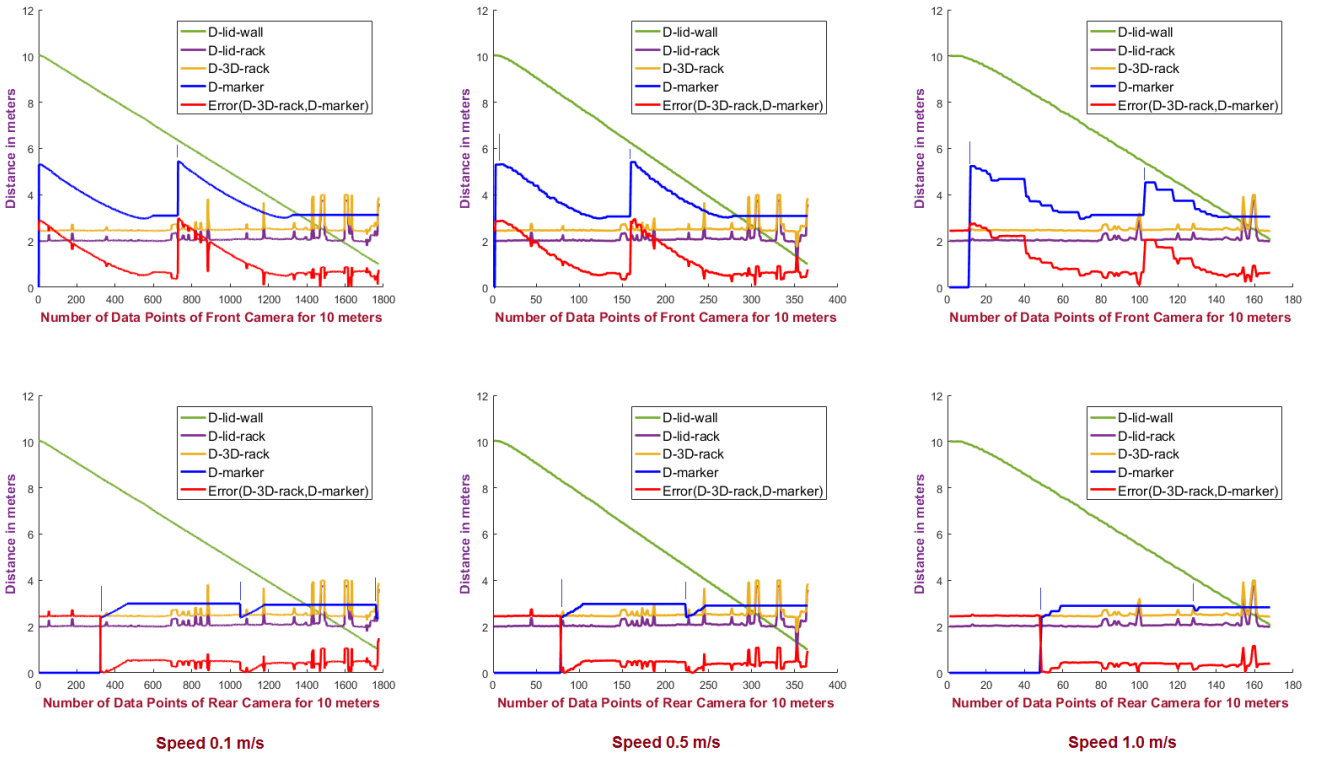


Figure 4.17 : Maker Detection by Front and Rear Cameras at 2.0 Meters

While keeping the camera angles fixed the marker detection test inside the racks provided a successful detection of markers. This detection can further be improved by 'scanning' the environment for a valid detection. From the results of sensors selection test and the marker detection test it was concluded that the use of landmarks for localization with the help of a dynamic framework such as 'ar_track_alvar' was sufficient for localizing the robot in the industrial environment under consideration.

4.3 Navigation

Referring back to the scenario 4.1 and considering the first phase of the global framework of operation, the next element of execution is 'navigation' after the robot has resolved its localization. The robot requires to plan and execute its locomotion towards the goal. This navigation aspect is global and involves the research and development of path planning and development of a control to make the robot reach the goal. The path planner will produce a feasible path and the control will make the robot follow this path to bring it to the goal. The forthcoming sections give a detailed overview of the path planning strategy used for the scenario 4.1 in the light of constraints explained in sections 4.2.7 and 4.2.8. When considering navigation as a whole, there are fundamentally two aspects involved which are path planning and motion planning. Path planning refers to the determination of action sequences that will make the robot in the environment eventually converge to the goal. The general path planning problems involves the computation of a feasible path that would bring the robot from the start point to the goal. The planning process is inherently dependent on the environment representation and complexity. The environment can be fully known, partially known or completely unknown. At the same time, it can be fully or partially unstructured or completely unstructured. Mostly the environment is partially known prior to the movement and the knowledge about the structural distribution and built up is acquired via exteroceptive sensors. The presence of structures in the environment creates obstacles to the movement and the planning has to be done under a strategy to avoid it. The obstacles can be static or dynamic. Based on the type and nature of the obstacles, the path planning can be either global or local.

4.3.1 Global Navigation

Global path planning is performed when the environment only consists of static obstacles and the planner produces a path from the start to the goal while avoiding obstacles. When the environment is not static and consists of dynamic obstacles then 'local' or reactive planning is required in which the robot acquires the information around itself at the current time to generate a new path around the obstacles. Further based on the environment representation the path planning can be model based or model free approach. In model-based approach all the information is provided beforehand consisting of description and geometric models of the objects and their layout. In model free approach the information about the environment is partially known and residual details are acquired by the sensors to have an updated knowledge and awareness. In model-based approach the path planning is done utilizing the concept of configuration space, which is defined as in [76] "the configuration 'q' of the robot is the number of parameters 'n' required to determine its position in 3D space". For a mobile robot, the configuration space is represented by three parameters, i.e. (x, y, θ).

But for path planning to accommodate both the holonomic and non-holonomic mobile robots, this configuration space is generally reduced to a 2D representation given by x and y coordinates by considering the robot as point.

The configuration space of the robot is segregated into free space 'CS_{free}' and occupied space '$CS_{obstacle}$'. The objective of the path planner is to find a path in 'CS_{free}' from the start to the goal point. In order to find a path in 'CS_{free}' the environment has to be transformed into a discrete representation. The common methodologies to transform the environment into a suitable representation are the Road maps, Cell decomposition and Potential field methods. In the road map approach, the robot's configuration space is decomposed based on the obstacle geometry. The concept in 'Road maps' is to construct a set of roads or path segments that would enable the robot to reach any position in the free space 'CS_{free}'. The road maps consist of a graph network consisting of collision free paths i.e. two nodes in the graph are connected by an edge if the robot is able to move in between them. A feasible path is produced by using the graph search algorithm to search the network for the connection between the start node and the goal node. The two commonly used approaches in road maps are the 'Visibility graphs' and 'Voronoi diagrams. In the 'Potential Field' method, the robot is treated as a point [75] under the influence of an artificial potential field created across the robot's map. This strategy acts as a control law for the robot movement since the robot motion is governed by the influence of the field. The robot moves by following the field analogous to a ball rolling downhill. The goal acts as an attractive force while the obstacles act as a repulsive force. The robot moves under the influence of the superposition of all the forces in which the potential field acts as a control for the smooth movement of the robot towards the goal, while avoiding the obstacles at the same time.

In the 'Cell decomposition' method the 'CS_{free}' of the robot is transformed into discrete number of cells. The cells are used to distinguish between free and occupied regions. The path planning finds a path by constructing a connectivity graph between free and adjacent cells from the start to the goal cell. This is achieved by finding the cells in which the initial and goal configurations lie and then searching for the path in the connectivity graph to join the initial and the goal cell. The cell decomposition further consists of two types. It can either be 'Exact' or 'Approximate' cell decomposition. The 'Exact' cell decomposition is a lossless decomposition as the boundaries are defined as a function of the environment structure based on the geometric criticality. The path planning in the network is complete since the resulting cells are either completely free or occupied. In the 'Approximate' cell decomposition the environment approximation is based on the cell occupancy or a resolution threshold of a grid representation. A common approach of approximation is the occupancy grid methodology. Occupancy grid decomposition is acquired by producing cells or tessellation based on the

probability of cell's occupation. Considering fixed grid size or cell decomposition, the conventional path planning algorithms are the 'A*', 'Dijkstra' and NF1 or 'Distance transform'.

A* is a graph traversal algorithm and a variant of the best first search algorithm which explores a weighted graph by expanding the most promising node according to a specific criterion. It finds the path starting from a specific node to the goal node having the smallest cost or least travelled distance. It does this[22] by maintaining a tree of paths originating at the start node and extending those paths one edge at a time until its termination criterion is satisfied. While expanding the nodes A* uses both the actual distance from the start node and the estimated distance to the goal. At each iteration the algorithm determines which of the paths to extend based on the cost of the path traversed and an estimate of the cost required to extend the path to the goal. Specifically, A* selects the path that minimizes the function

$$F(n) = G(n) + H(n)$$

where $G(n)$ is the cost of the path from the start node to node $'n'$ and $H(n)$ is the heuristic function that estimates the cost of the cheapest path from $'n'$ to the goal. The heuristic function is problem specific and the type of heuristic used effects the scanning of the nodes as all nodes might not be expanded at all. If the heuristic is admissible meaning it never overestimates the actual cost to get to the goal, then A* is guaranteed to return a least cost path from the start to goal. The success and performance of A* heavily depends on the admissibility of the heuristic function. The choice of heuristic function may or may not yield an optimal solution. Therefore, another prominent algorithm used to overcome this caveat is the Dijkstra's path planning algorithm.

Dijkstra's like the A* is also a best first search algorithm. It is applied both to directed and undirected weighted graphs for finding the shortest possible path. This graph search algorithm finds the shortest path between the start node and goal node in the graph. It starts from the source node and goes to every other node while building a set of nodes with minimum distance from the source node according to the weights of the edges. The algorithm expands outwards from the source considering every node that is closer in terms of shortest path until it reaches the goal. Dijkstra is essentially A* without the heuristics. The heuristic is always zero which makes the algorithm to expand in all directions to analyze each node, thus finding the shortest path between the source and every other node, eventually leading to the shortest one between the source to the goal. While exploring the nodes the algorithm calculates the shortest path by finding and maintaining the set of nodes that have the minimal distance from the source node.

Similar to A* and Dijkstra, one of the efficient and simple to implement technique for finding routes in fixed size array cells is the 'Grassfire' or 'Distance transform' algorithm. This

22 https://en.wikipedia.org/wiki/A*_search_algorithm

algorithm employs a wavefront expansion from the goal cell spreading outwards while assigning each cell, its distance from the goal. This is a breadth-first search algorithm implemented in the constrained space of an adjacency array.

4.3.1.1 Distance Transform

For the global path planning the path planner implementation makes use of the Distance transform method (DTM). This method considers the task of path planning from the 'goal' location to the start. The distance transform was first used in image processing for describing the shape of the blobs [103]. Then in 1985 Jarvis [104] extended the procedure to develop DT's throughout the free space of the robot's environment from a goal cell. The method was transformed to propagate distances from the goal cell filling all of the free space, flowing around the obstacles, using a raster scan method requiring multiple passes to guarantee total coverage in complex environments. Once all the distance values have been assigned then the shortest path to the goal is found by walking along the steepest descent path. The distance transform methods are propagation schemes based on a certain metrics of propagation, in which Euclidean distance is the most common one. The distance transforms are computed based on the 'order' in which the cells are processed. The basic three methods [105] are the Order propagation, the raster scanning and Independent scanning. In the order propagation the smallest distance information is computed starting from the goal cell with '0' distance and then progressively transmitting to other cells in order of increasing distances. The raster scan method uses 2D masks to guide the processing of cells, line by line from top to bottom and then bottom to top. Independent scanning method processes each row of the map separately (independent of other) and then each column is processed afterwards. Summing up, all the three methods in essence perform propagation.

The significance of DT is that it does not require knowledge of a start location and generates a distance map only once based on the goal location since all the distance values propagate outwards from the goal location. This is in contrast to the most of the global path planners where the knowledge of the start and goal location is required initially to search for a path, making use of the connectivity of the adjacent nodes in a particular graph or a map. The successful execution of these methods depends on the initial knowledge of the start and goal location. Using DT the robot pursues a steepest descent path, while at the same time it can be disturbed pursuing this path to the goal, unlike other global path planners where the re-planning is required when the robot is removed from the original path. The DT is implemented utilizing the map of an initially known environment with the knowledge of the static obstacles and the goal location. It can also be used to compute the map of an unknown environment, where in that case the robot will have to build the map on the move and

simultaneously compute the distance map and the path based on the local information acquired via sensors. However, this will be excessively computational and the robot will have to move at a relatively slower speed to accommodate for the map building and the distance transform computation. To account for the dynamic obstacles in the static map, DT can be used employing a path planning strategy of a global and local path planner. The global path planner will compute the map of the already known environment and the path to the goal. The local path planner will track the global path while employing a local window maintaining a local map to recompute the local path to avoid the obstacles and converge back to the original path.

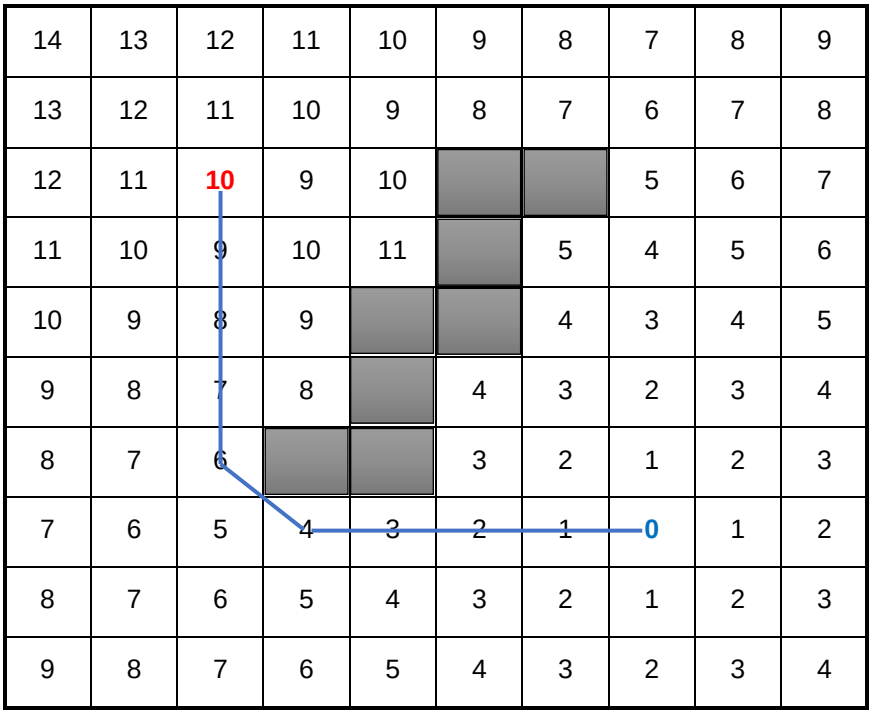


Figure 4.18 : A Distance Transform Map with Path

Considering the order propagation distance transform, it is based on the grass fire analogy and is also called wavefront or bushfire method. Being one of the simplest and effective technique to execute path planning, starting from the boundary cells the distance of cells from the closest to farthest is computed. 'The processing [105] is performed only around the narrow band of cells (fire front) where a change in the current stored distance can occur". This propagation algorithm is almost similar to Dijkstra's algorithm. In the initial stage the workspace of the environment is transformed into an occupancy-based grid representation. DT uses this occupancy-based grid map to compute the distance map. The goal cells are assigned the minimum value, the obstacles cells are assigned very high values (infinity) and the rest of the free cells are assigned values that represent its distance from the goal. The

distance values are propagated flowing around the obstacles (in grey) as can be seen in figure (4.18). The order propagation algorithm executes in iterations until all the values of the free cells have been assigned and stabilized. During the value assignment scans, the obstacles are skipped over. The resulting distance transform calculated by the propagation algorithm is independent of any start point and represents a distance potential field with no local minima. This enables to find a globally minimal distance path from any start point in free space to the goal following a steepest descent path. Figure (4.18) represents a grid map with the distance values calculated with the distance transform. The goal cell value is '0' and the start point value is marked red. The obstacle is in the center of the map and the steepest descent path is shown in blue.

4.3.1.2 Algorithm

For the undertaken research project, the distance transform strategy has been adopted owing to the fact, that CAD map of most of the warehouses is readily available, and the layout is almost similar. The CAD map can easily be converted to an occupancy grid map, representing the occupancy of the static structures (racks) in the map. In the first step the map with its given dimensions is transformed into a 2D occupancy grid tessellated representation. Then the occupied cells and free cells are assigned the respective values. In the algorithm used for this thesis, obstacle cells and free cells are assigned alternate negative integer values. Any specific integer value can be used but the negative value is assigned since the distance values incremented in the free cells by the order propagation are positive integer values. The goal cell is assigned a '0' distance value and then the order propagation increments the distance of the free cells using a 2D mask. The 2D mask can use either all 8 cells or just 4 cells with respect to the center cell at the mask window. For the algorithm given below, a mask of 4 cells has been used to increment the values of the neighborhood cells outwards starting from the goal.

	+1	
+1	C	+1
	+1	

$$d(x,y) = \begin{cases} d(x,y+1)+1 & if\ \ I(x,y+1) = -1 \\ d(x,y-1)+1 & if\ \ I(x,y-1) = -1 \\ d(x+1,y)+1 & if\ \ I(x+1,y) = -1 \\ d(x-1,y)+1 & if\ \ I(x-1,y) = -1 \\ d(x,y) & if\ \ I(x,y) = -10 \end{cases}$$

where $'I(x,y)'$ represents the map matrix and $'d(x,y)'$ gives the distance matrix values to increment in the neighborhood cell of the center cell $'C(x,y)'$ using the mask as shown above. The order propagation algorithm runs in a loop with each pass of the map employing a local neighborhood operation in order to increment the distance by '1' with respect to the current cell $'C(x,y)'$ at the center of the mask window.

Algorithm 1 : Distance Transform

Input: $I(x, y)$ – 2D map initialized with obstacles and free cell values
Goal (i , j) – goal cell coordinates
Output: d(x,y) – distance transform map

```
Begin
  fill_map=true;      map(i)(j)=0;    increment=0;  //goal value initially
  while fill_map ! = null  do          //as long as map has default values keep filling it
      fill_map=false
    for  rows=0 to rowsmax   do
      for colms=0 to colmsmax  do
        if map(rows)(colms) = -1
           fill_map=true
        elseif map(rows)(colms) = increment
          if rows-1>= rowsmin      //check previous vertical neighbor of mask
            if map(rows-1)(colms) = -1 || map(rows-1)(colms)>increment+1
              map(rows-1)(colms) = increment+1
          if rows+1< rowsmax       //check next vertical neighbor of mask
            if map(rows+1)(colms) = -1 || map(rows+1)(colms)>increment+1
              map(rows+1)(colms) = increment+1
          if colms+1< colmsmax     //check next horizontal neighbor of mask
            if map(rows)(colms+1) = -1 || map(rows)(colms+1)>increment+1
              map(rows)(colms+1) = increment+1
          if colms-1>= colsmin      //check previous horizontal neighbor of mask
            if map(rows)(colms-1) = -1 || map(rows)(colms-1)>increment+1
              map(rows)(colms-1) = increment+1
    increment++
```

In the above algorithm the goal value is initialized with '0'. The algorithm runs in a loop and the mask starts iteration from the goal value at the center cell of the mask window. The mask keeps scanning and incrementing as long as the map has remaining default (-1) values. In each iteration the mask scans horizontally and updates (increments) the values of the four neighbors of the mask center cell i.e. horizontal and vertical. Then in the next iteration the mask updates four neighbors of the center cell which was incremented in the previous iteration as shown below in 2D mask. The blue square shows the center cell of the mask. At the end of the loop the complete map is transformed into a distance map with each cell having the value of its distance from the goal cell.

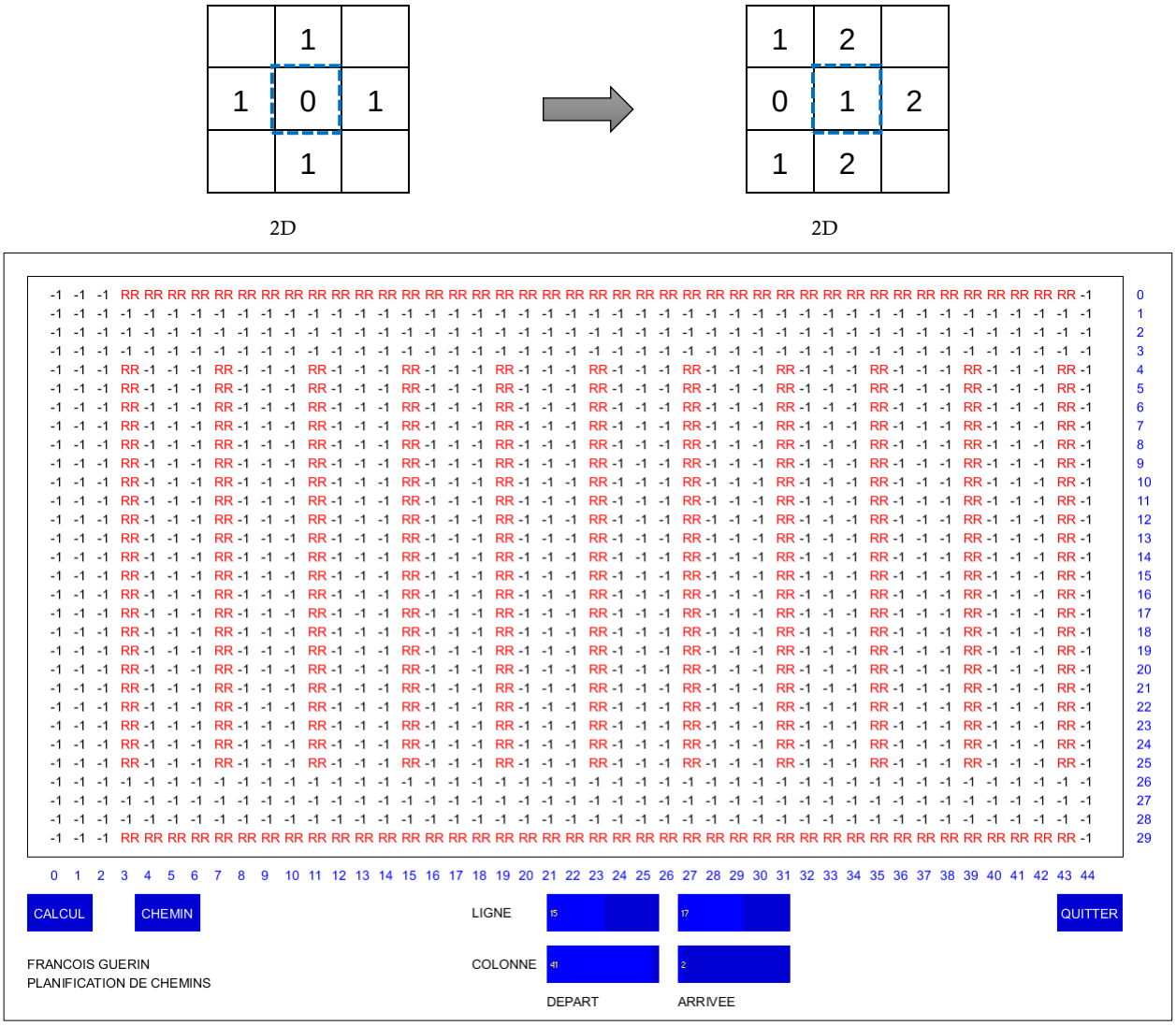


Figure 4.19 : DT Map with Initialized Values

The figure (4.19) above shows a map of a warehouse initialized with the default values of '-1' corresponding to free cells and the obstacles (racks) or no-go area is shown by 'RR'. The same map is transformed into a distance map with all the distance values with respect to the goal. The goal (in red) and the start point (in blue circle) is shown in the map in figure (4.20). Utilizing the distance map, the path is calculated by following the steepest descent path, or scanning and connecting the cells with smallest distance from the start point back to the goal. The algorithm for the path calculation is given below and the map showing the extracted path is given in figure (4.21).

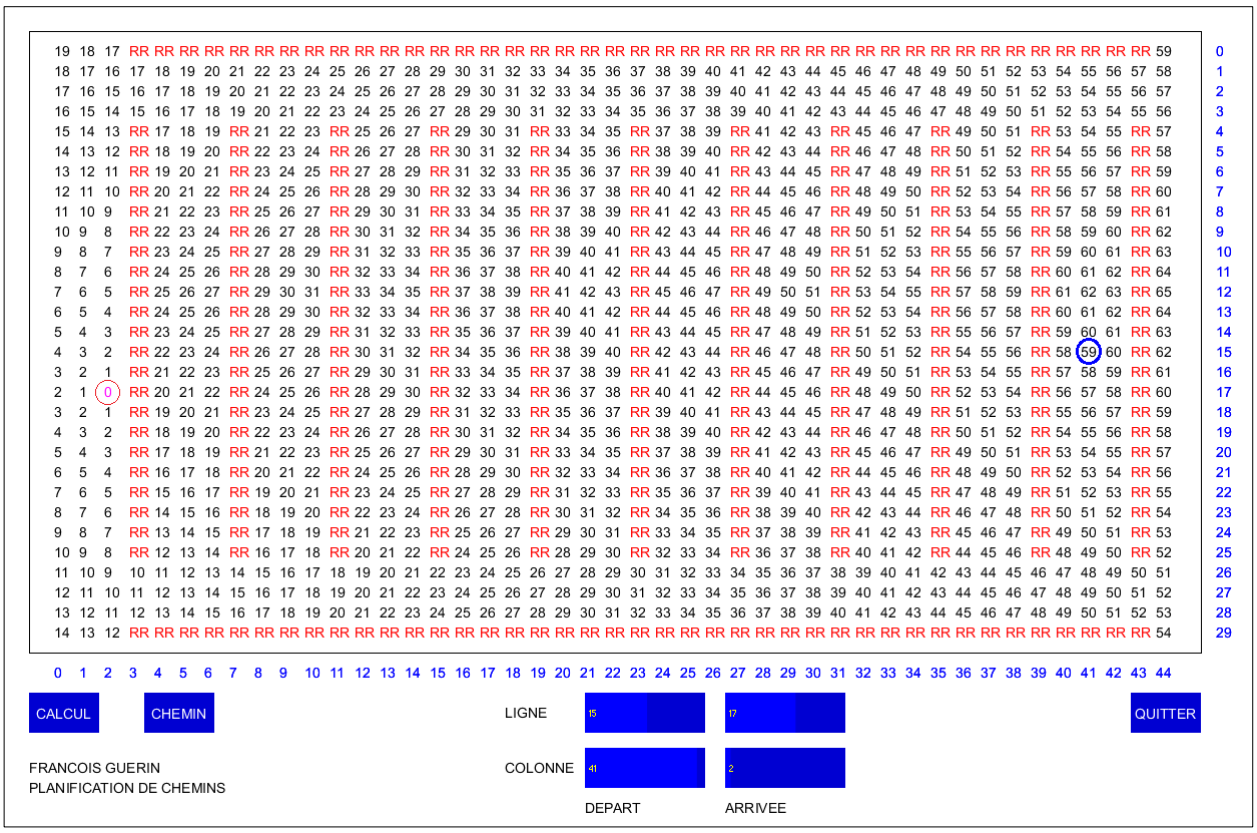


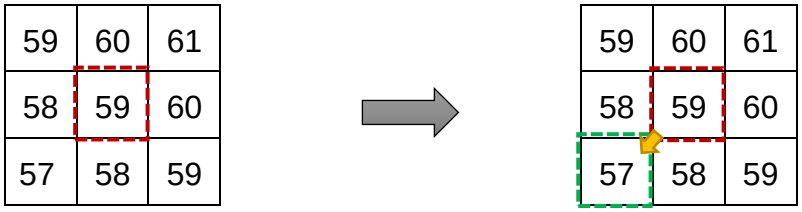
Figure 4.20 : Distance Map with Distance Values from Goal

Algorithm 2 : Path Calculation

Input: $d(x, y)$ – 2D distance map calculated with Algorithm 1
Start (i , j) – Start point cell coordinates
Output: $P(x_i,y_i)$ – and i ∈ 0……n where 'n'= number of path coordinates
Begin

```
R=i; C=j;       //start point value
while compare_val ! = 0  do                    //while it is not the goal point
   min_row_val=0, min_col_val=0, min_val=compare_val
   for  rows=-1 to 1   do
      for colms=-1 to 1  do
         if (R+rows>=rowsmin & R+rows<rowsmax )
                  & (C+colms<colmsmax & C+colms>=colmsmin) //Check the map limit
         if map(R+rows)(C+colms)<min & map(R+rows)(C+colms)>=0   //extracting the minimum values
                  & map(R+rows)(C+colms) ! = obstacle_val )        //check around each cell of the
                                                                   // current pivot cell
            min_val= map(R+rows)(C+colms)
            min_row_val=rows        min_col_val=colms

   R=R+min_row_val          Path(xi)=R       //storing the row for path
   C=C+min_col_val          Path(yi)=C       //storing the colms for path      Compare_val=min
```



In the 'Algorithm 2' the steepest descent path is extracted by scanning a 2D mask in the distance map acquired by algorithm 1. The 2D mask is initialized from the start position (start point of path) and in the first iteration the cell with the minimum value is selected. The mask extracts the cell with the lowest value in each iteration by scanning all 8 neighbors of center cell. The path is constructed by connecting all the minimum value cells extracted in each iteration of the loop. The cells thus produce a steepest descent path from the start to the goal. A path for the map given in figure (4.20) above is shown in figure (4.21) below.

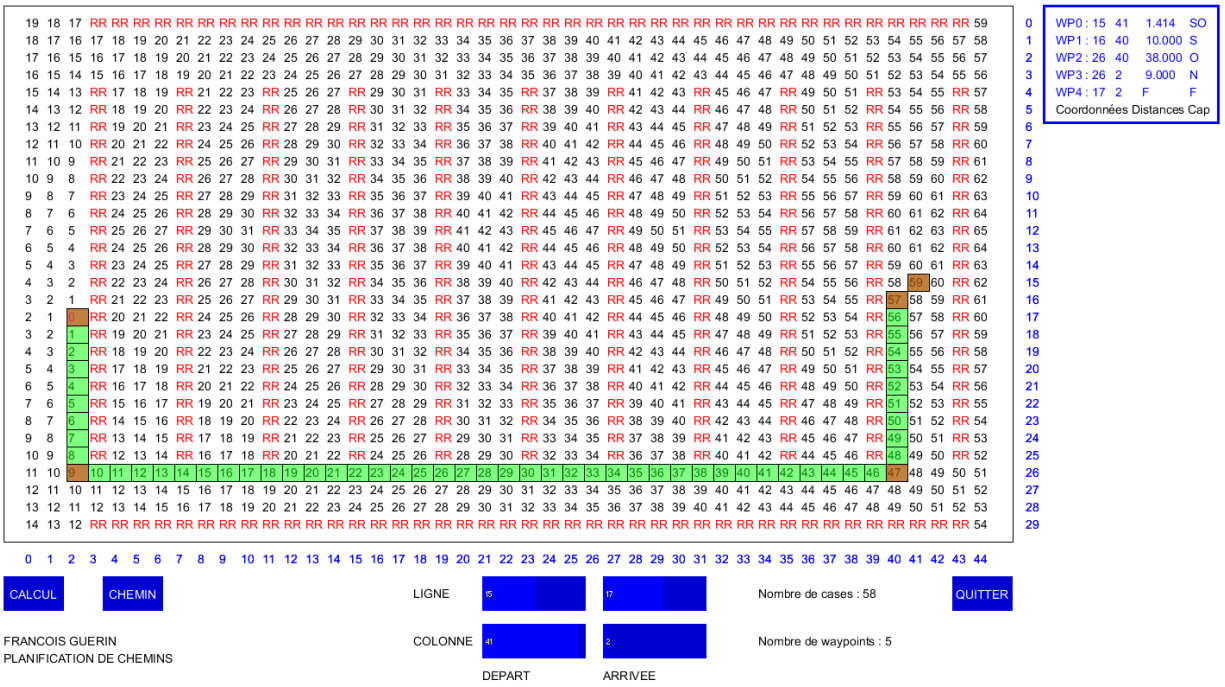


Figure 4.21 : Shortest Path Extracted in the Distance Map

The algorithm produces a shortest path between the start and goal positions which can be verified by checking the cell values. The waypoints produced and the direction to move is given in the top right window with a total distance between each waypoint. The resolution of the map in the above figure is 1 meter and the total length of the path is thus 59.141 meters. The map in the figure above consist of symmetric layout of racks in one hall of the warehouse. The path planner and the effective distance propagation in the presence of an unsymmetrical layout of obstacles is given more clearly in the figure (4.22) below.

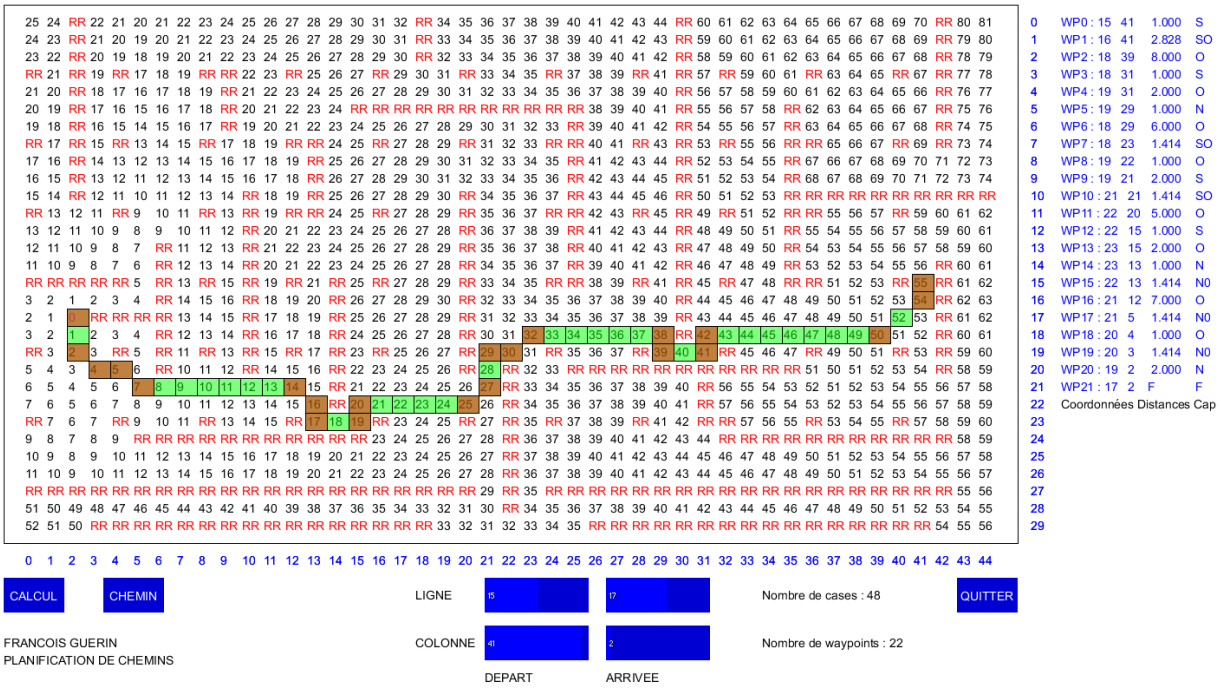


Figure 4.22 : Path Generated in DT Map with Dense Obstacles

All of the above simulations were developed using "processing" open source library[23]. The real-time implementation of the algorithm was done in C++ utilizing the map of the existing research facility. A 2D CAD map of one of the testing arenas was transformed into a 2D tessellated grid representation whose resolution could be changed on demand. This metric map is tessellated regularly into a fixed decomposition grid with each cell occupying 5 to 50 cm in different instantiations of the resolution.

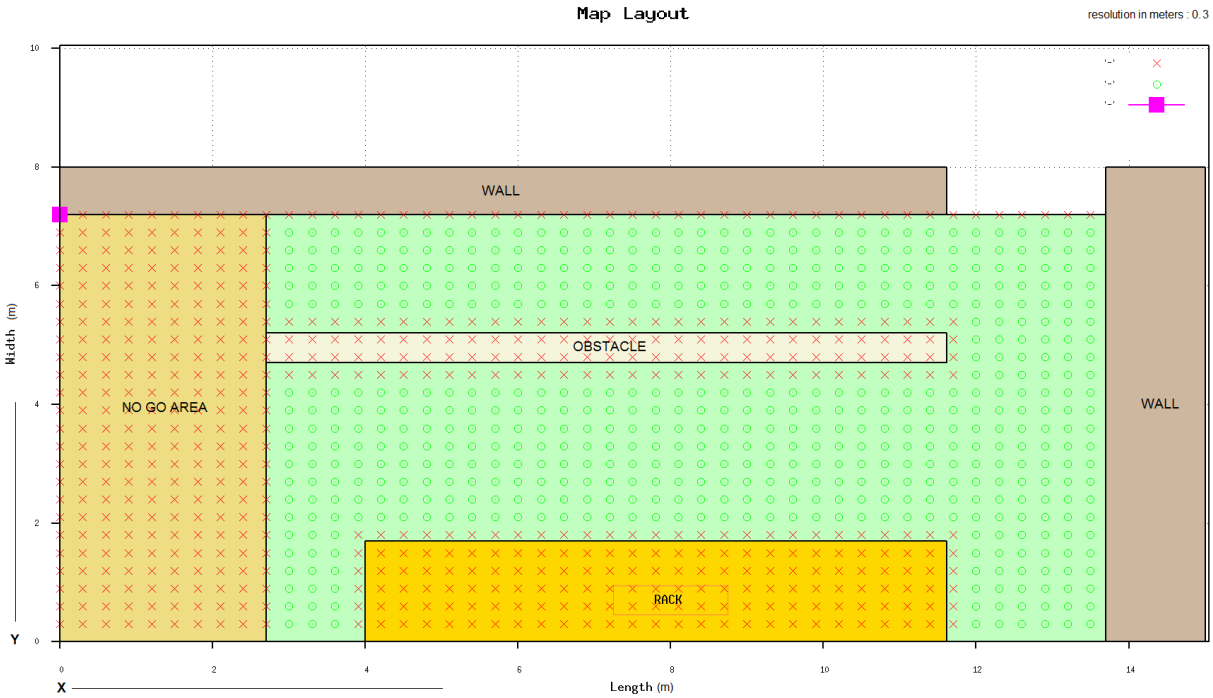


Figure 4.23 : Tessellated Map of Testing Arena at 30 cm resolution

The figures (4.23) above and figure (4.24) below show a map with a resolution of 30 cm and 10 cm (inter cell distance). The area in green with green circles, i.e. figure (4.23) represents the free cells, whereas the area marked with red crosses gives the obstacles cells. Each cross or a circle corresponds to a cell. The areas in figure (4.24) are transformed to a fine resolution of the cells in the map below. In order to avoid the robot from coming too close to obstacles, the obstacles are inflated by a certain threshold so the resulting path is not too close to the obstacles. This is visible by the overlapping of the red regions (crosses) around the obstacles.

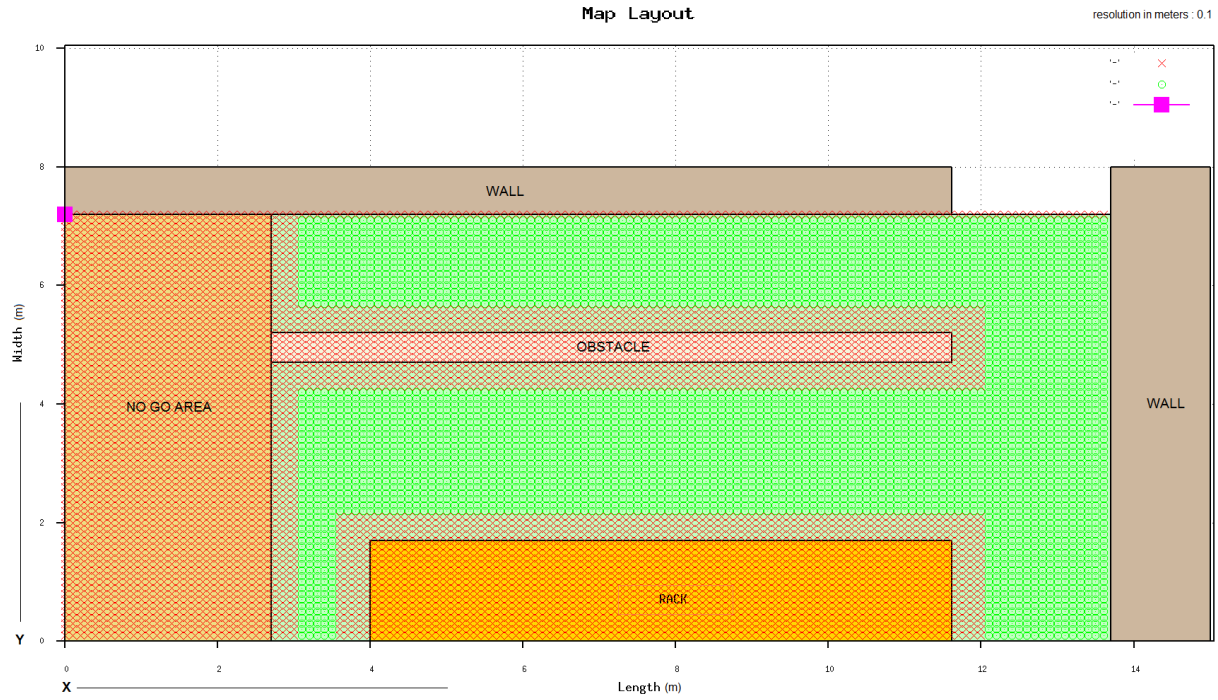


Figure 4.24 : Tessellated Map of Testing Arena at 10 cm resolution

4.3.2 Motion Planning

After the generation of a feasible path the robot has to move under an effective control strategy to follow that path accurately. The locomotion aspect of the robot involves as to how and when to move. The how aspect is concerned with the realization of the sequences of valid configurations that enable the robot to reach from one state to the other. The answer to this aspect leads to the development and implementation of a control architecture to produce a motion of the robot that meets a certain requirement, for instance the tracking of path coordinates precisely for path following. In the previous section the aspect of path planning has been explained in detail. This section focuses on the development of the control strategy to follow the path produced by the path planner.

23 https://processing.org/

4.3.2.1 Control

The goal of the control is to track the waypoints produced by the path planner. This can be done by producing the respective velocities of the robot to track the set point. The set point in this case is the path waypoint and the control law will generate the velocities that should reduce the error between the current position and the desired. Consider the figure (4.25) below :

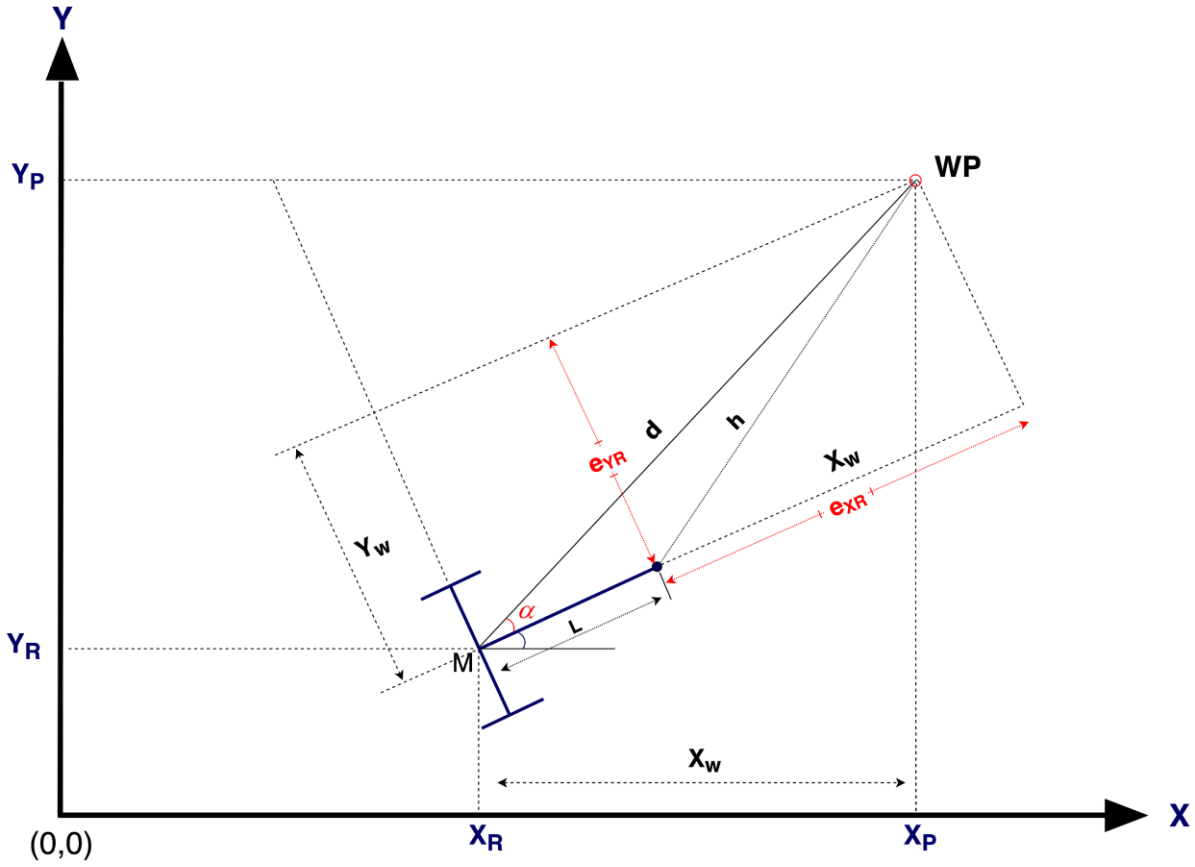


Figure 4.25 : Geometric Model of Robot For Control

In the above figure the notations are :

(X,Y) → Global coordinate frame

(X_P,Y_p) → Waypoint coordinates in the global coordinate frame

(X_R,Y_R) → Robot coordinates in the global coordinate frame

(X_{WR},Y_{WR}) → Waypoint coordinates in the robot frame

The waypoint is transformed in the robot frame by :

$$\begin{bmatrix} X_{WR} \\ Y_{WR} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \cdot \begin{bmatrix} X_p - X_R \\ Y_p - Y_R \end{bmatrix} \tag{4.10}$$

and distance '*d*' is : $d = \sqrt{X_W{}^2 + Y_W{}^2}$ and angle 'α' is : $\alpha = \tan^{-1}\left(\frac{Y_W}{X_W}\right)$

The robot with its center 'M' is placed in the global coordinate system. It has to reach waypoint 'WP' of the path. To achieve this the control objective is to regulate the distance *'d'* and the steering angle 'α' as follows:

$$\lim_{t\to\infty} d(t) = \mathrm{L} \qquad \lim_{t\to\infty} \alpha(t) = 0 \tag{4.11}$$

The above condition eventually results in the situation that at the end of the robot movement the point 'P' of the robot should coincide with the waypoint coordinates. This means that the distance errors on both the 'X' and 'Y' axis should converge to zero. The distance errors in robot coordinates are given by

$$e_{X_R} = d\cos(\alpha) - \mathrm{L} \quad \text{and} \quad e_{Y_R} = d\sin(\alpha) \tag{4.12}$$

The state evolution of the system has to be such that the distance error between the current and desired position has to converge to zero, if the robot has to reach the waypoint. To fulfill the control objective (4.11), let us consider the following error dynamic equations:

$$\dot{e}_{X_R} = f(\mathrm{e}_{XR}, U_M, D_{X_M}) = -(u + \tau_u.\dot{u}) \quad \text{and} \quad \dot{e}_{Y_R} = -K_Y.e_{Y_R} = -(r + \tau_r.\dot{r}) \tag{4.13}$$

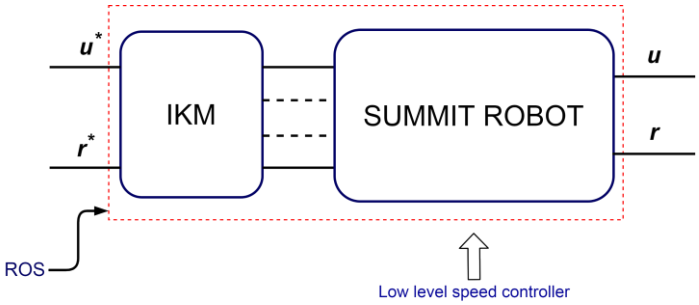


Figure 4.26 : Robot Level Control Block

Considering the control scheme above, the control input to the robot is 'u' from the control law. The summit robot includes the Inverse Kinematic Model (IKM) given in (3.4) and the lower level control of the electrical drives. The control architecture as shown in the above image is a smooth PID control and was developed by the robot manufacture. The robot block was considered as a black box since it was not possible to access the low-level control architecture of the actual robotic platform due to the limitations put by the manufacturer. However, an attempt was made to evaluate the system and analyze the response for various step inputs for system characterization.

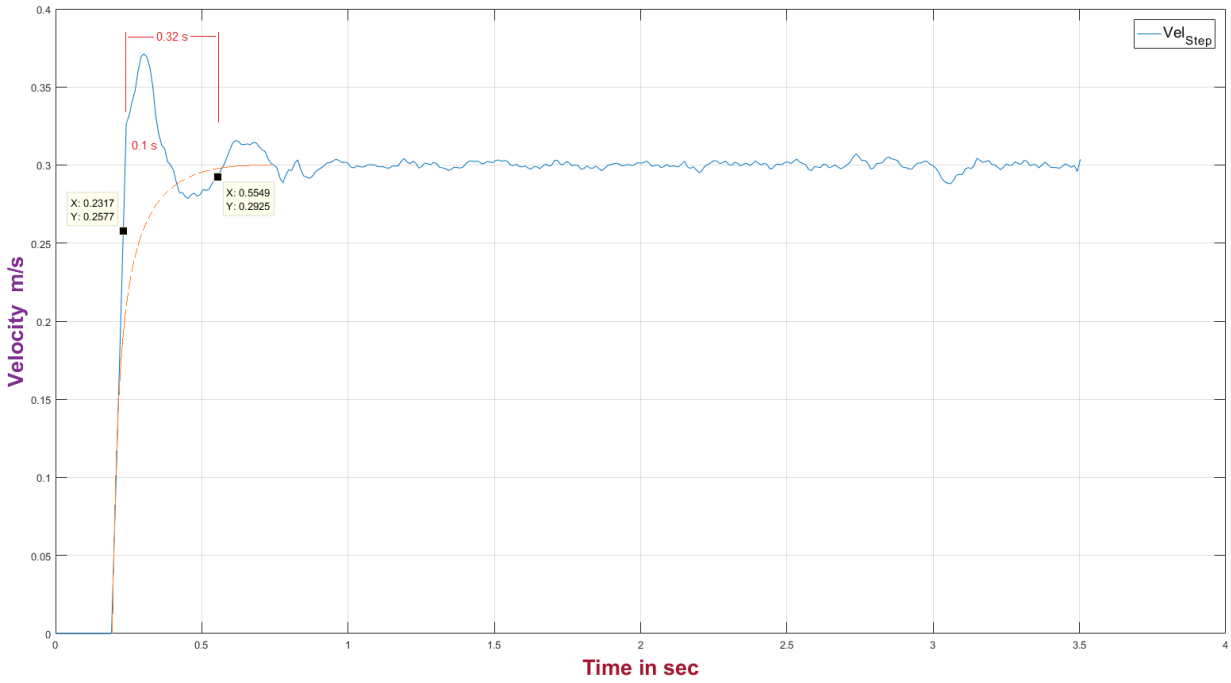


Figure 4.27 : Open Loop Step Input System Response

Figures (4.27) shows the response of the robot to a step input command of 0.3 m/s. From the output response of the system it can be observed that the system looks like a second order system without static or steady state errors. The response stays within the ±2 % of the final value. The response reaches the final value in almost 0.3 seconds. This is evident in the figure. However, it is difficult to conclude about the damping or extract the damping ratio since the frequency of the oscillations is high as compared to the system. Since the output response was recorded from the sensor giving out the odometry values of the robot, it is reasonable to conclude that the oscillations are from unfiltered sensor output i.e the output response is not a filtered one and contains frequency components of both the sensor noise and the control outputs. In conclusion damping is not very important because the initial overshoot duration of '0.1s' is quite small. Over all the performance of the controller was found to be very smooth and well-tuned.

4.3.2.1.1 Linear velocity controller

In discrete time (Euler) the control input 'u' is then derived from as follows

$$u_n + \tau_u\left(\frac{u_n - u_{n-1}}{T}\right) = f(e_{XR}, U_M, D_{X_M}) \;\Rightarrow\; u_n = \frac{\tau_u}{T+\tau_u}.u_{n-1} + \frac{T}{T+\tau_u}.f(e_{XR}, U_M, D_{X_M}) \quad (4.14)$$

where 'T' is the sampling period, 'τ_u' is the time constant which has to be tuned to provide smooth acceleration. The function '$f(e_{XR}, U_M, D_{X_M})$' is a saturation function defined as follows:

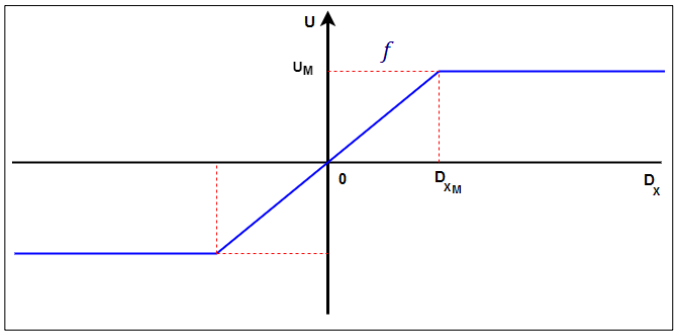


Figure 4.28 : Saturation limits for longitudinal velocity

where 'U_M' is the maximum allowed velocity, and 'D_{X_M}' is the distance starting from the robot deceleration. Consequently, the control law for the linear velocity is given as:

$$\begin{aligned} u_n &= \frac{\tau_u}{T+\tau_u}.u_{n-1} + \frac{T}{T+\tau_u}.f(e_{XR}, U_M, D_{X_M}) \\ u_n &= \frac{\tau_u}{T+\tau_u}.u_{n-1} + \frac{T}{T+\tau_u}.f(e_{XR}, U_M, D_{X_M}).\left|1-\sin(\alpha)\right| \end{aligned} \quad (4.15)$$

A term '$\left|1-\sin(\alpha)\right|$' in the above control law is added as a scale factor. It is maximum when $\alpha = 0$ and it is reduced by an amount '$\sin(\alpha)$' since in that condition the robot's motion is more influenced by the rotational velocity for turning rather than going straight.

4.3.2.1.2 Angular velocity controller

For the angular velocity control, again in discrete time (Euler) the control input 'r' is then derived from as follows

$$\omega_n = \frac{\tau_u}{T+\tau_r}.\omega_{n-1} + \frac{T}{T+\tau_r}.K_Y.d.\sin(\alpha) \quad (4.16)$$

To be independent of the distance 'd', the term '$K_Y.d$' has been replaced by a maximum allowed angular velocity. Figure (4.29) shows the 'closed loop' response of the system. The top plot exhibits the linear velocity response for a distance move of 2 meters. The saturation 'U_M' is kept at 1 m/s to extract a full view of the response. Since the distance to move is small, the velocity quickly converges towards zero after the initial rise. The lower plot is for the distance move of 6 meters. In this case the error is high and therefore the velocity is limited by the saturation for the initial rise period, but as soon as the robot gets closer to the goal the velocity starts to decrease.

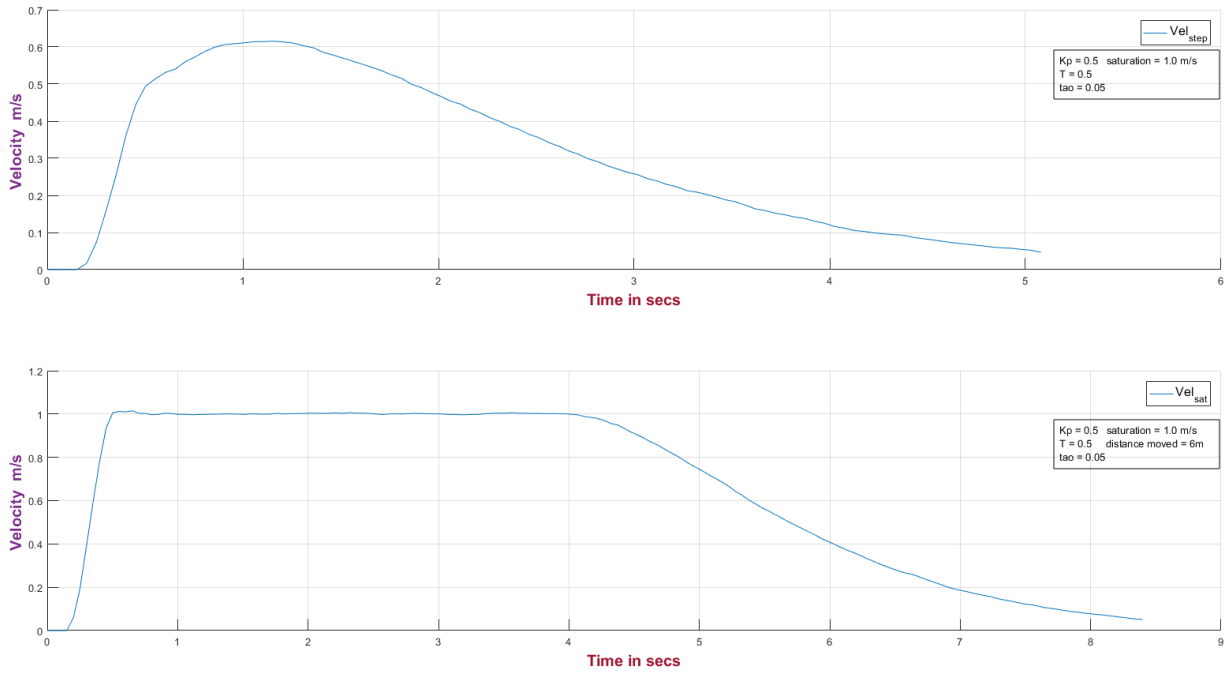


Figure 4.29 : Closed Loop Response of First Order System

4.3.2.2 Path Tracking

The path tracking is performed by using the control law developed above. The tracking strategy involves the tracking of specific waypoints. The path planner produces a complete path in the global (x,y) coordinates of the map. This path is further segregated into a set of waypoints for tracking. The waypoints are chosen at specific turns and the goal position of the path. These waypoints are given to the robot as local goal points to reach. The robot then traverses the whole path by reaching these waypoints sequentially. Figure (4.30) below shows the path tracking evolution. The resolution of the map is 10 cm and the start point is 'SP' is at x=3.7 m and y=6.2 m, with the goal point 'GP' at x=6 m and y=3 m in front of the rack. The

global path produced by the path planner is in blue whereas the path traced by the robot is given in red. The robot path is recorded during the motion from the odometry values and the coordinates are plotted onto the map directly.

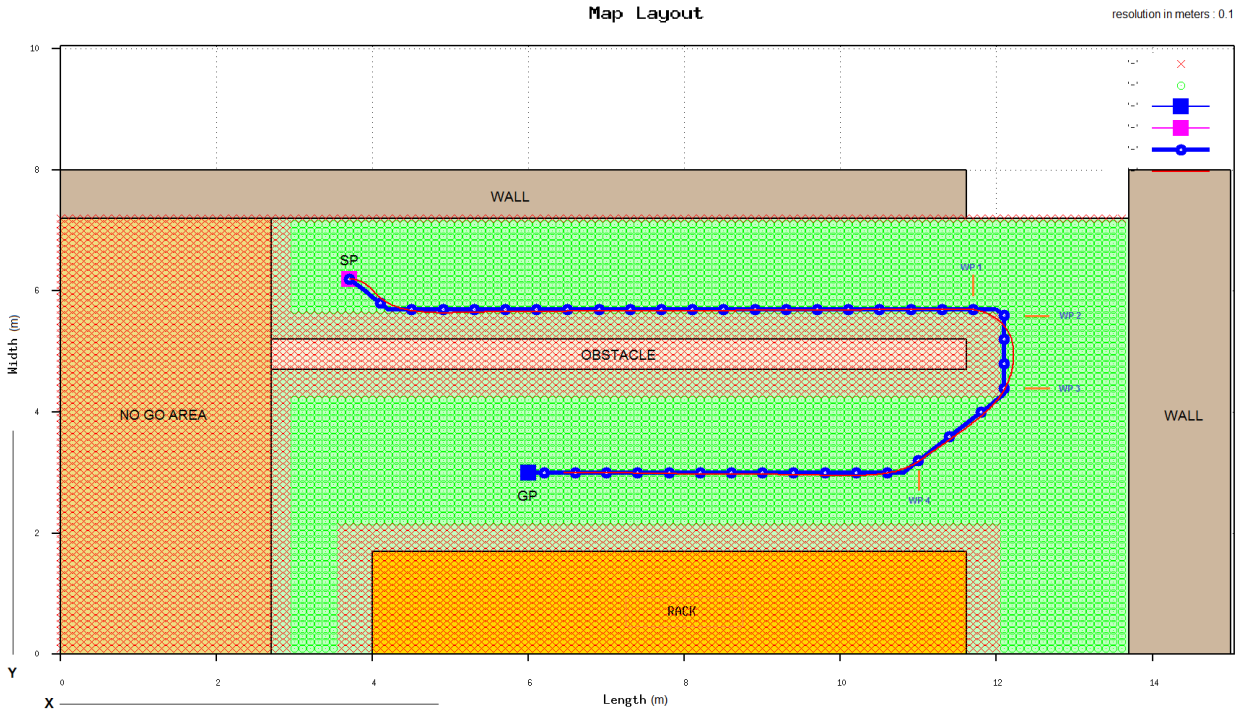


Figure 4.30 : Generated Path Tracked By Robot at 10 cm Map Resolution

The robot traverses[24] the path around the obstacle to reach to goal. Here an observation has to be made with regard to the obstacle avoidance. The current path planner produces a global path for the robot around static obstacles, and does not take into account the dynamic obstacle avoidance. Likewise, the robot just follows the path without sensing any obstacles. The development of a obstacles avoidance strategy for navigation was not pursued in this thesis due to constraints of time and scope of the research. However, there is substantial room for improvement in the current path planner for incorporating obstacle avoidance. Figure (4.31) below shows the time evolution of the path tracked by the robot, along with the speed and distance errors Ge(x,y) for each waypoint. The distance error 'Ge' is the remaining distance to reach a specific waypoint and is given by

$$Ge = \sqrt{(e_{X_R})^2 + (e_{Y_R})^2} \tag{4.17}$$

The data was extracted in MATLAB from the recorded data during robot motion. The speed of the robot corresponds to the tracking of waypoint distance. For smaller distances the speed of the robot is minimal whereas as for larger distance errors in bottom figure (4.31), the maximum speed of the robot is limited by the saturation visible in figure (4.31) middle. The

24 https://vimeo.com/389303036

distance errors gradually converge to zero as the robot reaches closer to the local waypoint goal, along with the decrease in speed. This is visible from the plots in middle and bottom plots.

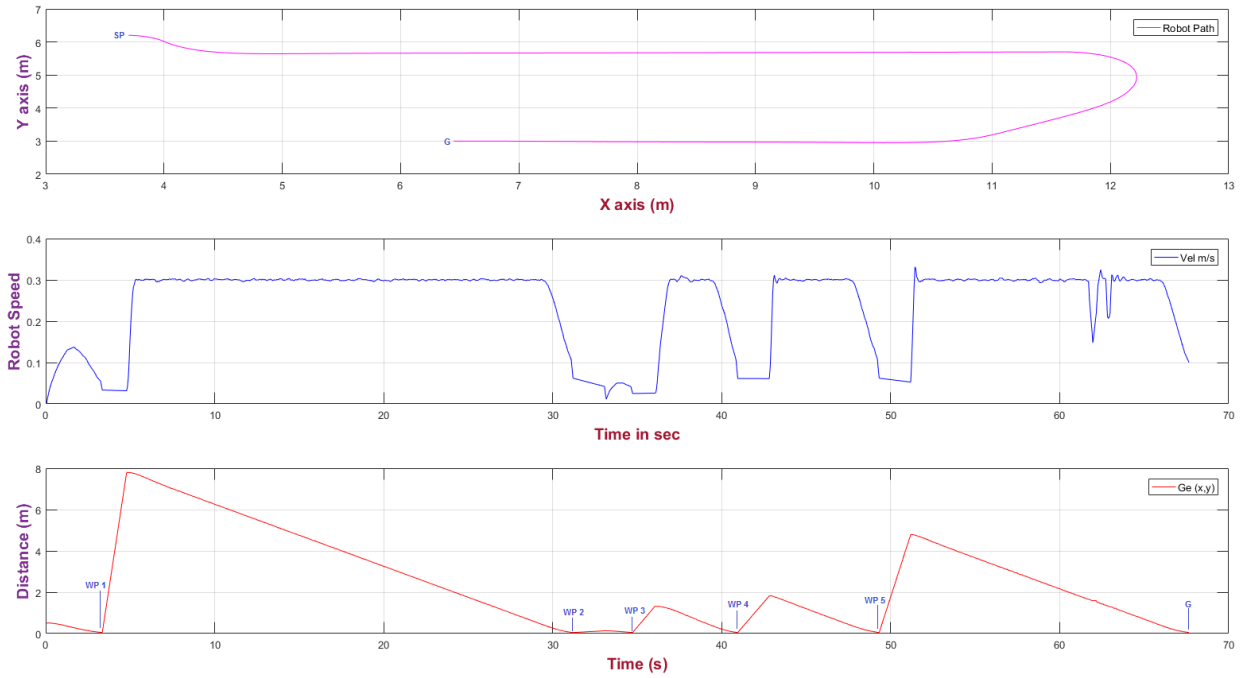


Figure 4.31 : Time Evolution For Speed and Waypoints Following

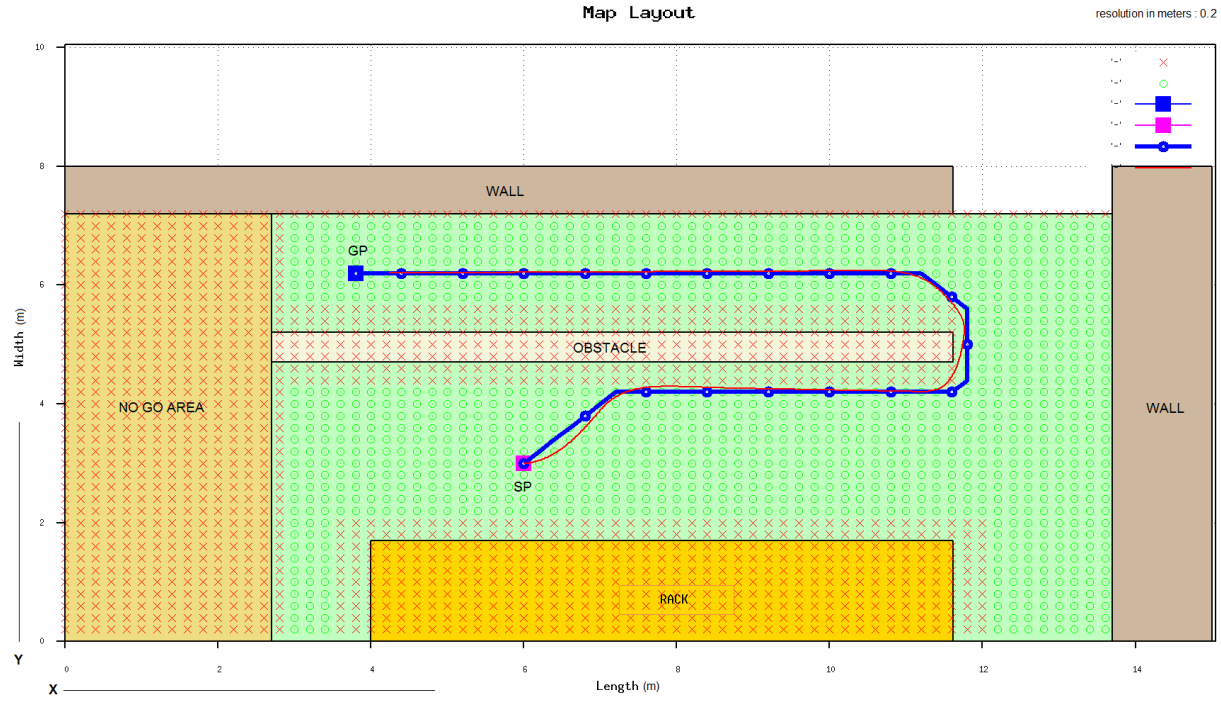


Figure 4.32 : Generated Path Tracked By Robot at 20 cm Map Resolution

Figure (4.32) above shows the same map with a resolution of 20 cm and the path of the robot to go back to the same position as in figure (4.30). However, the start point and goal point are reversed. The resolution of the map is distinguishable from the small green circles. The time evolution of the distance and velocity in figure (4.33) is similar to the figure (4.31). The speeds and distance errors increase and decrease as the robot moves from one waypoint to the other along the path.

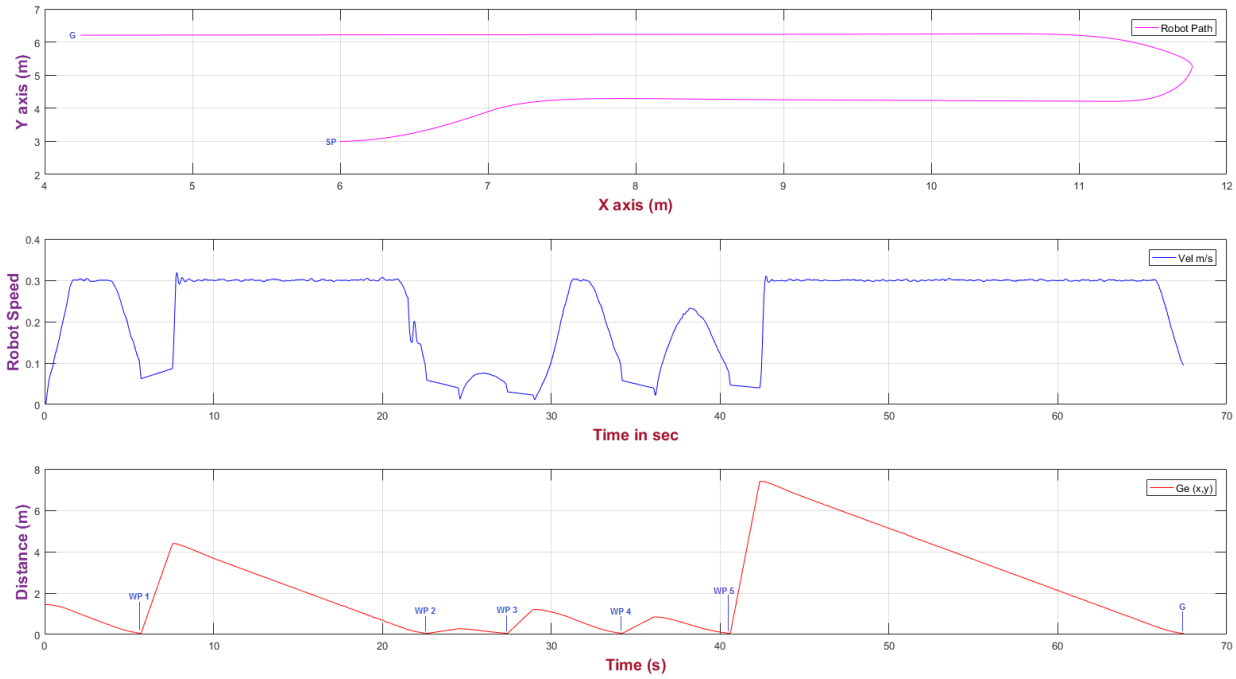


Figure 4.33 : Time Evolution For Speed and Waypoints Following

4.3.3 Rack Navigation

Going back to scenario 4.1 and reflecting on the global objective of picking, comprehensively the first phase consists of reaching to the pick location. This involves the successful materialization of two elements in the phase, namely localization and navigation. The navigation itself consists of two elements. The global framework to make the robot come to the goal location and navigation inside the racks to move for picking operation. Once the robot has reached a rack location using the architecture explained in the preceding sections, subsequently for the picking activity it has to utilize the rack navigation framework. The objective of rack navigation framework is to enable the robot move under precise navigation requirement in the presence of racks. The answer for the requirement of a specific navigation framework inside the racks is provided by giving an overview of the requirements pertaining to the picking operation and more specifically to move inside the racks.

4.3.3.1 Navigation requirement

To find a solution to the issue of motion inside the racks, the situation can be compared to the corridor following problem, where the movement or navigation of the robot is facilitated by a reference of the either sides of the corridor. Similar to path tracking for global navigation, this reference for motion control is readily available from the sides. Thus, the navigation problem becomes easy in a corridor following case. The robot executes a linear motion while keeping a fix offset from the sides avoiding any collisions. Analogously the racks can be considered as virtual corridors when stocked with the pallets containing material. A reference can be acquired from the sides using a sensor (lidar) for virtual wall (side) following. In addition to the implementation of this framework a requirement was specified by the enterprise (company) to utilize a navigation strategy that facilitates the movement of at least two vehicles (robots) in the same rack. Therefore, the corridor motion realization framework facilitates this requirement, since each robot can move on either side while maintaining a fixed offset from the reference of the rack side. Considering the availability of space inside the rack aisles, the typical width of the alley formed by two racks is 3 to 3.5 meters wide, providing limited room to maneuver and execute turns if required. This leads to the infeasibility of collision avoidance with another robot in the same zone (alley) operating simultaneously. To fulfill this requirement a rectilinear motion is more effective than a curvilinear or zigzag motion strategy in a constrained space.

Another requirement put forth by the enterprise is the inventory management of the stock inside the racks to be achieved by the robot during picking activity. This requires the robot to move linearly at constant velocity at the center of the rack alley, scanning both the sides of the racks while maintaining a fixed distance to have a symmetric view of both sides simultaneously. To achieve this the corridor following motion strategy establishes as the ideal choice of framework, since references from both the sides can be used to keep the robot in the center or any required offset from either side. At the same time for picking operations the parallel motion to the racks facilitates a trivial detection of the pallets. Yet another additional demand scenario corresponds to large order volume. This is the case when the frequency of the picking activity is high to meet the demand of multiple customers in time. When the number of items to be picked in a specified zone is more, it is required to deploy multiple order pickers (robot units in the current case) to meet the picking demand and maintain the productivity level. To achieve this the deployment of multiple picking robots requires a coordinated and effective navigation strategy. The robots in this scenario are deployed to specific racks and act as loaders, picking and placing the products on the transporter units. An effective motion strategy to coordinate their motion is the parallel navigation along the racks. This lower level navigation strategy will be combined with a higher-level control and

coordination scheme that would enable the robots to engage and disengage from the pallets for picking operations. The system would thus exhibit the functionality of a finite state automaton.

4.3.3.2 Strategy

Based on the analysis of requirements and a conclusive navigation strategy of corridor following, to implement the framework the most obvious choice is to utilize a reference extraction and tracking algorithm since the reference provided by a wall or corridor corresponds to a consistent linear edge or a line. These reference or line tracking algorithms provide the reference input for the motion control of the robot to make it move in straight line. The research community has dedicated a considerable amount of effort to the development of line (edge) extracting algorithms, and numerous variants have been used for robot motion control. Their requirement is significant in the wall or corridor following scenarios, or whenever the motion controller requires a virtual edge as reference input. Lines and edges were first extracted in image processing [106], where edge detection algorithms were applied to digital images to extract a valid edge for image segmentation. These algorithms produced edges in the forms of lines and curves [107]. The algorithms made use of either gradient based techniques or a nonlinear mathematical filter [108]. The common technique of these methods involved the use of a convolutional kernel or operator on the image to detect the presence of an edge. In addition to the convolutional kernels, other methods [109] made use of the parametric representation of the lines or curves to extract the edges. One of these methods is the 'Hough transform'.

Line extraction has been realized not only using Hough transform but also other line extraction algorithms. In robotics line (edge) extraction is important for control, navigation and localization. The employment of line-based reference control for navigation was in effect from the early development of mobile root navigation. Line extraction methodologies were employed both for control and mapping. The maps built up from extracted lines were either static or dynamic and used for robot localization and navigation. A line segment map development is given in [110]. Based on the scanned points from the radial laser scanner, a line segment map is built directly from the 2D range data. The methodology consists of short line segments to approximate any kind of environment. Similarly, in [111] the authors have used the 2D range data and line segments to build a dynamic map for robot navigation. The map is composed of line segments and circles, and the methodology consists of building lines by 'regression' from the measured points. The lines are computed by first computing a point slope form and then converting it into respective polar form (rho,theta). The uncertainties of the measured points and final parameters (rho,teta) is also taken into account by computing the

covariances. The authors in [112] have used the recursive line splitting method to extract lines to build a close connected (CLS) map of the environment for robot localization. Likewise, in [113] the authors have presented an overview of three-line extraction algorithms. The three techniques consist of an online incremental technique, an offline method to extract linear approximations from data based on split and merge, and a technique consisting of producing expected maximization (EM) of the most likelihood linear models, where the probability of the most likely measurement to the model is computed in the expectation step and based on that, the parameters of the model are calculated in the maximization step. Correspondingly in [114] the authors have thoroughly evaluated 6 commonly used lines extraction algorithms in computer vision and robotics. These algorithms are used to extract line features to produce maps for robot localization. The algorithms are evaluated on the basis of speed, complexity and precision based on two laser scan data sets acquired from two different robotic platforms and testing sites. The performance of these line algorithms is further evaluated by utilizing them to generate maps in Orthogonal SLAM (OrthoSLAM). The six algorithms presented for evaluation are the 'Split and Merge', 'Incremental', 'Line regression', 'Random Sample Consensus (RANSAC)', 'Expectation-Maximization (EM)', and the 'Hough Transform'.

4.3.3.2.1 Hough transform

The classic 'Hough' transform was invented by Paul Hough in 1962 [115] to plot the tracks of subatomic particles in bubble chamber photographs. The method proposed by Hough involved transforming the points in the figure (picture) into a straight line in parameter space. This parameter space is defined by the parametric representation of the slope intercept form. The parametric space as proposed by Hough was a two-dimensional slope intercept plane, and had a performance limitation of the unboundedness of both the slope and the intercept. This limitation was overcome and an improved version of Hough transform (HT) was introduced in 1972 by Duda and Hart [116]. This consisted of a polar representation of the parametric space. This transform was put to use in computer vision to detect the edges of shapes by extracting lines in digital images. This transform is used till date with its further variations and improvements in image processing [117] and [118]. A detailed overview of the basic Hough transform and its variations is presented in [119]. The important aspects presented are the early development, improvements to cope with limitations and detect other shapes, novel techniques for efficient accumulator distribution, performance quantization based on variation of parameters, application of Hough to detect complex shapes, patterns, motion estimation from images, 3D shapes from range data and use of asynchronous parallel architecture for reduced computational load. For extracting lines in digital images, the authors in [120] introduced an improved version of Hough transform to have the specific advantages over the conventional HT, of infinite parameter space, high resolution, small storage and high

computational speed. The method consisted of picking image pixel points at random based on equal probability of selection, and computing their respective parameters of the line model in parametric space. The lines are extracted by finding the maximum of the accumulated cells in the parametric space.

One of the earliest applications of Hough transform in robotics [121] was to detect a docking work station for a robot using images from a video camera. A series of HT's was applied to detect edges in the image filtered by a 'Sobel' operator, while taking into account the constraints of position and orientation of the object to recognize in the image. The edges extracted by the HT enabled to recover the position of the dock within the image and guide the robot to it. In image processing the HT is applied to points or pixels to extract corresponding lines. This application requires the point data form in which lines and curves can be extracted by the employment of suitable parameterizations. This highlights the fact that if the data acquired from a sensor can be transformed to a representation of points, then HT can readily be applied to extract the lines or edges in the data. This approach has been used to extract lines or curves from range data acquired by a sonar or a 2D laser range data. A significant work based on control, consisted of the use of HT inside the feedback loop of navigation controller of a mobile robot. These works [122] and [123] focused on the navigation in cluttered rooms using a laser sensor, and the 'Range weighted Hough transform' (RWHT) was applied to extract distances and orientations to the walls, to estimate the location of the robot. This location was given by the Extended Kalman Filter (EKF) which was used to estimate both the position of the robot and the size of the room simultaneously. The peaks produced by the HT were used in the feedback loop of the navigation controller to represent the walls for control reference. The robot was navigated successfully for a corridor following and door passing problem. The same framework was further used in [124] for mapping the environment for navigation. Another work in [125] based on the combination of angle histogram and 'RWHT' consisted of absolute localization, by computing and tracking the pose of the robot with respect to the walls inside a rectangular room.

A similar work of localization and mapping [126] consisted of map building and localizing within, using geometric primitives of lines. The map is built using the range data from a sonar sensor and is based on the primitives of lines and probabilistic framework of occupancy grid. The range data is stored in occupancy grids and line features are extracted by applying HT to the stored data, and then clustering to group them in similar classes (walls) to produce the map. The robot was localized by using 'EKF' to combine the perceived local world model with odometry and sonar data in the localization scheme. Similarly, the authors in [127] worked on the position tracking of a mobile robot equipped with 7 sonar sensors. The method is based on the HT and probability grids and the localization is achieved by correlating objects

such as walls, corners and edges in the sonar data with a reference map of the environment. The HT is applied on the data acquired from the sonar sensor and the localization method consisted of computing a two-dimensional feature space of lines (landmarks) acquired by Hough transform, and then performing a template matching in this feature space. The correlation counts obtained in this matching were used to update the position probability grid. The sonar readings and the world map are matched directly in the Hough space and the pose of the robot was tracked by the use of the Kalman filter which made use of the template matching in the Hough space to do the correction and update the probability distribution about robot's position.

The authors in [128] worked on the 3D mapping of the environment using a conventional 2D laser sensor with the objective of safe navigation. The 2D scans were collected in the online phase and a line matching and Hough transform algorithm was applied to extract the lines. The sequence of lines in successive 2D scans approximated the shape of a surface and shape reconstruction was done by concatenating the lines in 3D to a single surface. Finally, the full 3D map was constructed by offline object segmentation and detection. Another work [129] presents the real time localization of a robot in a polygonal environment. The approach called the Hough localization made use of geometric representation of the reference map (line and circles) and was based on the map matching in the Hough domain (parameter space) and not in Cartesian space of the robot sensor reference system. The overall method consisted of matching the sensor data acquired by a vision-based range sensor with the model (environment map) and integrate that information with odometry using EKF to acquire an updated probability distribution of the absolute pose of the robot in the map. The method was tested in a real time competitive environment by comparing the trajectories from an external vision system and Hough transform.

A significant work on mapping based on HT is given in [130]. The authors have presented a map construction algorithm consisting of fitting line segments to uncertain data points in the 2D range data using a likelihood formalism. Hough transform is applied to extract subsets of collinear points from the range data. Then lines are fitted to sets of uncertain points by a maximum likelihood formalism by weighing each point's influence on the fit according to its uncertainty i.e. by minimizing the errors over the set of measurements. The contribution of each virtual error for the data point is weighed according to its modelled uncertainty. The final map is formed by merging all the optimal lines under a threshold of combined uncertainty. In [131] the authors have worked on the mapping frame work to generate 2D line maps from laser range data using clustering in the spatial and Hough domain. The methodology consists of first producing density-based clusters in the scan data by applying 'IEPF' and then noise is removed from those clusters with the application of morphological kernels of erosion

and dilation. Then ‘k’ means or fuzzy ‘C’ means clustering is applied to compute the centroids of the noise free data, and then HT or SVD is applied to join the centroids into straight line segments map. The method was tested in a real cluttered environment and results were compared to the EKF-SLAM to conclude that the proposed methodology produced more single line maps much faster than standard EKF-SLAM for large data sets. Lu et al in [132] worked on the application of HT to detect curbs(edges) in the stereo range data. The work was focused on the navigation of a mobile robot indoors requiring precise localization to negotiate the stairs. Stair detection was acquired by having the perceptive information through a stereo camera to detect the curbs. The method consisted of first extracting the ground plane to filter out incorrect curb lines and compute a suitable curvature index (CI). Then candidate curb lines were extracted from image edge segments by applying HT to the 3D points weighted by the CI and closer to the ground plane. The 3D lines were then regressed from the back projection of points in the detected image lines to produce final 3D curbs onto the 3D points in the scene. A set of final concave and convex line segments was used to localize the stairs.

Perhaps the work closest to this thesis is the one in [133] which presents a method to detect crop rows using Hough transform and effectively guide an agricultural machine such as an interrow cultivator or a mobile robot. The rows of crops are recognized by applying HT to gray scale intensity images acquired by a focused camera. Taking into context both the crop and weed data in the scene, a set of parallel adjacent lines is extracted to produce a rectangular box to model a plant row. A heading angle and row offset is calculated for each row in right and left and then fused together to get a global angle and offset. This information is given as input to a PID controller to guide a cultivator. The system was tested for different kinds of rows and the position accuracy was reported to be within a deviation 2.7 to 2.3 cm relative to a row. Another work [134] focused on the use of HT to control a wheel chair passing through a door. The door was estimated from the discontinuity of the laser range scans and Hough transform was applied in a divide and conquer approach to extract the lines from the given scan. The door posts were calculated from the intersection of Hough lines, and the parameters of these lines were used by a Kalman filter to track the pose during navigation through the door, i.e. a controller used the Kalman filter estimates to make pose corrections of the vehicle.

A work dedicated to localization [135] made use of the line feature-based SLAM. The framework was applied to a differential drive service robot equipped with noisy and sparse IR sensors. The methodology was based on a geometrically constrained framework, utilizing EKF to impose geometric constraints of orthogonality and parallelism between the lines in the measurement model. The line extraction algorithm was based on a constrained HT, whose voting space was restricted within the vicinity of initial line parameters, which were acquired by applying the least squares line fitting technique, to each cluster of scan data obtained by the

IEPF clustering algorithm. A work concerning the performance evaluation of HT with other line extraction algorithms was carried out in [136], where the authors utilized a sequential application of two algorithms to extract lines in the laser scan data. The methodology consisted of first detecting clusters of data by applying a 'Distance based Convolutional Clustering' (DCC) to detect a break point to cluster the data points. Then a reduced HT with a reduced accumulator array was applied to the clusters to extract line segments. This was done to reduce the processing time to detect the local maxima in Hough space. The algorithm was applied to scan data acquired by a laser sensor on a mobile robot in different environments, and the performance of DCC-RHT was compared with other clustering and line extraction algorithms separately. The evaluation was based on two correctness measures and was found to be more efficient for both clustering and line extraction while using DCC-RHT.

Last but not least, in a recent work in the context of precision farming operations such as weeding or fertilizing, Wolfram et al. [137] have worked on the navigation of a robot utilizing the pattern Hough transform to detect crop rows. The objective is to localize a robot with respect to a crop row and the method consists of detecting a pattern of parallel equidistant lines and estimating the angle, lateral offset and crop row spacing in a single step. The approach makes use of application of HT to the extracted plant features from vision and 3D point cloud laser range data. The extracted features from the RGB images and point cloud are merged together in a probabilistic 2D grid map of features. Then a single line HT is applied to extract a pregiven spacing crop row pattern, followed by applying a dual line HT to estimate the actual row spacing by extracting a second parallel line. The algorithm was tested successfully for detecting crop rows while traversing the field during navigation and the pattern HT was compared with a RANSAC pattern extractor for robustness of localization and navigation.

4.3.3.2.2 Theoretical Framework

As explained before Hough transform is applied to extract specific features in the 2D image or range data. The features to be extracted are generally lines and elliptical curves such as circles. This is acquired by a transformation of the input data to a parametric representation to extract the parameters of the hypersurface (feature) to locate in the given data. More generally the space defined by the input space is termed as the feature space and the transformed space is known as the parametric space. The feature space is typically 2D whereas the dimension of the parametric space depends on the parameters required to produce the feature in the feature space. For example, in the case of a line, the parametric space will be 2D whereas in the case of a circle it will be 3D. The standard equation of line is given by in the

$$y = mx + c \tag{4.18}$$

(x, y) space. For a number of points (x_i, y_i) lying on this line the corresponding parametric space will be the (m, c) slope-intercept space. Computation of the parameters of the line in the parametric space will correspond to a single point. Every line in the (x, y) space will map into a point in the (m, c) space. Conversely every line in the (m, c) space will map into a point in the (x, y) space. This is a reciprocal transformation of mapping points to lines and vice a versa. The caveat with the basic linear representation (4.18) is that discrete computation of (m, c) parameters requires memory, and for a vertical line in the (x, y) space the slope 'm' becomes infinite which will require infinite memory for computation. This limitation is overcome by the use of the 'Hesse normal form' of the line i.e.

$$\rho = x\cos(\theta) + y\sin(\theta) \tag{4.19}$$

where $'\rho'$ is the distance from the origin along the normal to the line and 'θ' is the angle between the x-axis and the normal. Using this parametrization all points of a line conform to the equation (4.19). This is the standard representation of the 'Hough transform' as proposed by Duda et al. [116] and used till date. Now using this representation, each point in the (x, y) space will transform to a sinusoid in the parameter space as shown in figure (4.34). This parameter space will be initialized as a discrete space whose size will depend on the bounds of (ρ, θ). An accumulator array will be formed based on the discretization of (ρ, θ) in which 'θ' can vary from $'0 \rightarrow \pi'$ and the values of $'\rho'$ can be positive or negative depending on 'θ'.

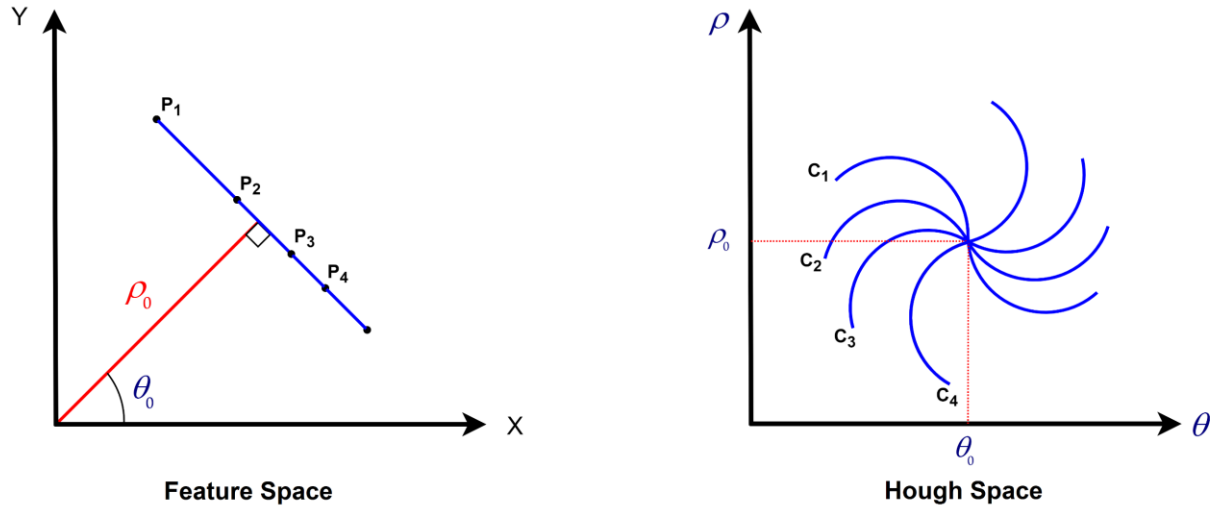


Figure 4.34 : Transformation of Points into Curves From Feature Space to Hough Space

The resolution of the accumulator depends on the size of the accumulator. Each cell (ρ_i, θ_i) in the accumulator array is incremented once for each curve (sinusoid) that passes through it. The cells where these curves (sinusoids) intersect will give the maximum accumulations or

votes and correspond to local peaks in the Hough space. Each local peak corresponds to a specific feature, and its location gives the parameters of that feature. That feature can be found by back projecting those parameters in the feature space. The significance of this approach is that the detection of peaks in the parameter space is more straight forward and computationally less expensive, than matching spatially extended patterns of points in the feature space.

4.3.3.3 Implementation

The implementation of the Hough transform requires the availability of the given data in a point form structure. Considering its application to image processing, the image data is readily available in pixels. These pixels are used in the Hough space to detect the peaks corresponding to those specific pixels representing particular regions in the image. Analogously Hough transform is applied to 'range' data taking advantage of the point structural form of the data distribution. The range data is provided by Ultrasonic or Laser range finders as distance measurements in the form of spatially extended distribution of points. The distances are extracted from the objects such as walls and corners in the environment. The data points are transformed to Hough space to extract local peaks, to detect specific features in the range data such a lines and curves corresponding to walls and corners. Considering the preference of the sensors to extract the data, the ultrasonic sensors produce more noisy data due to reduced range and crosstalk thus resulting in extraction of more ambiguous features. This shortcoming was overcome by the use of LIDARS since they provide more crisp and precise data with higher resolution in the form of spatial structure of points. A fixed planar Lidar takes measurements in one plane producing a 2D map and the features extracted using the Lidar data are less ambiguous leading to consistent maps.

4.3.3.3.1 Sensor and Data Extraction

The implementation of Hough transform for this thesis was done using a Lidar. The Lidar was a standard Hokuyo UST-10LX, with a range of 10 meters. The characteristics of the sensors are given in the table (6) below. The range data was acquired by two Lidars mounted at the front and back of the robot separately.

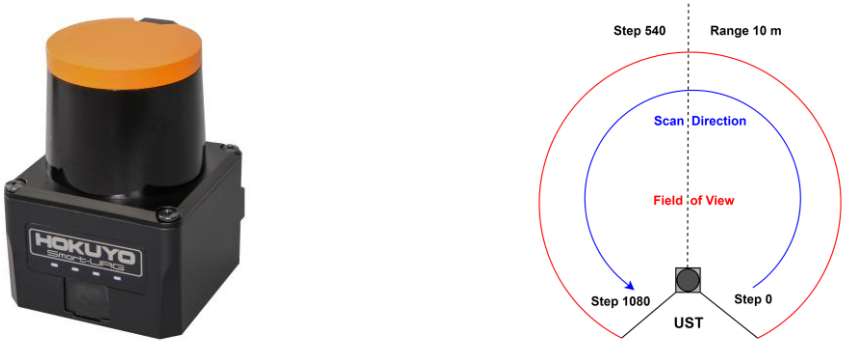


Figure 4.35 : Hokuyo UST-10LX Laser Range Sensor

Laser Range Sensor	Hokuyo UST-10LX
Scanning Range	0.02 m - 10 m
Scanning Angle	270°
Angular Resolution	0.25° (360°/1400 steps)
Accuracy	±40 mm
Scanning Frequency	40 Hz (2400 rpm)
Response	25 ms
Measurement Steps	1081

Table 6 : Sensor Characteristics

Each lidar gives the range measurement in the form of radial distance 'd' and radial angle 'θ'. The sensor has a '270°' of field of view. The positions of the object are calculated with a step angle of 'θ' and radial distance 'd'. The resolution of the sensor is '0.25°' giving out four readings per degree measurement. The robot moves in a two-dimensional Cartesian space 'R^2'. Lidar data is presented as a scan of 2D points in this space. Each scan point is given by its (x_s, y_s) coordinates. Figure (4.36) shows the two lidars on the robot producing a scan of two different environments. The scan from the front lidar (upper green point in figure) is shown in blue data points and the rear lidar (lower green point) in red data points.

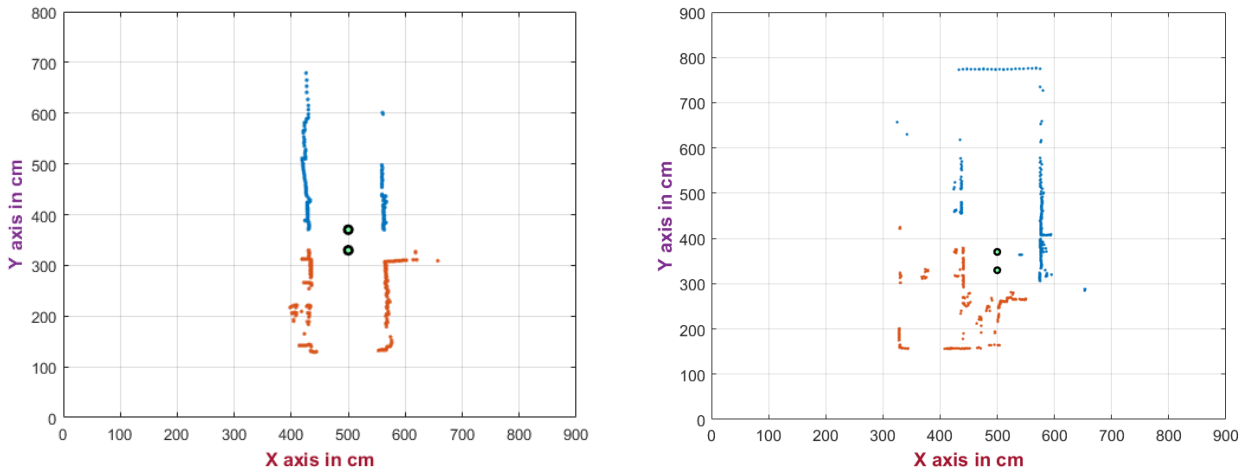


Figure 4.36 : Lidar Scans of Two different Environments

The scans are acquired in the sensor frame and consist of a number of points as

$$\begin{pmatrix} X_{lid(i)} \\ Y_{lid(i)} \end{pmatrix} = \begin{pmatrix} d_i \cos(\theta_i) \\ d_i \sin(\theta_i) \end{pmatrix} \quad i = 1..2\ldots n \qquad (4.20)$$

where 'n' is the number of points, 'd_i' is the radial distance and 'θ_i' is the radial angle for the 'i_{th}' point. The scans are aligned to the robot orientation and plotted with the use of equations (4.20) and (4.21). Consider the figure (4.37), the coordinates of the 'i_{th}' scan data point with

respect to the robot in the range image (X,Y) are given as $(x_{\text{fr}}, y_{\text{fr}})$ for the front lidar and $(x_{\text{r}}, y_{\text{r}})$ for the rear lidar respectively.

$$\begin{pmatrix} x_{\text{fr}(i)} \\ y_{\text{fr}(i)} \end{pmatrix} = \begin{pmatrix} \frac{w}{2} + (x_{Flid(i)} + rob\_wid \times scale) \\ \frac{h}{2} - (y_{Flid(i)} + rob\_len \times scale) \end{pmatrix} \qquad \begin{pmatrix} x_{\text{r}(i)} \\ y_{\text{r}(i)} \end{pmatrix} = \begin{pmatrix} \frac{w}{2} - (x_{Rlid(i)} + rob\_wid \times scale) \\ \frac{h}{2} + (y_{Rlid(i)} + rob\_len \times scale) \end{pmatrix}$$

(4.21)

Where 'h' is the height of the map and 'w' is the width. After the transformation the range scan is mapped with the robot orientation in the range image with the robot at the center.

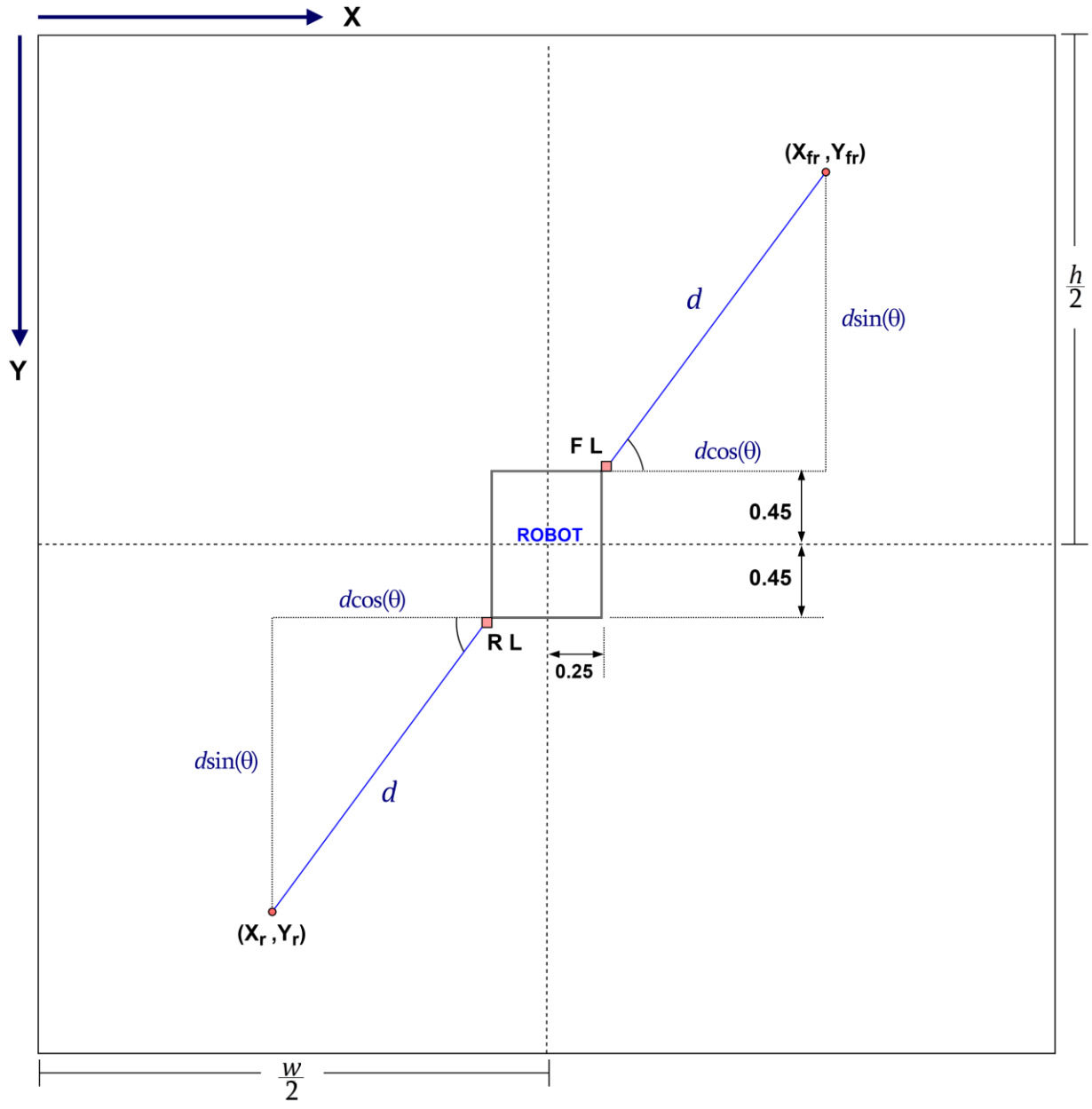


Figure 4.37 : 2D Representation of the Virtual Range Image

4.3.3.3.2 Algorithm

After the acquisition of the virtual 2D range image, Hough transform is applied to the scans to extract line features. The input to the Hough space is the feature space i.e. the 2D range image. The information (data) in this feature space produces peaks in the Hough space. These peaks are projected back in the feature space to generate the corresponding lines. Figure (4.38) shows the application of HT to a range image (a) with the corresponding lines of interest in (b).

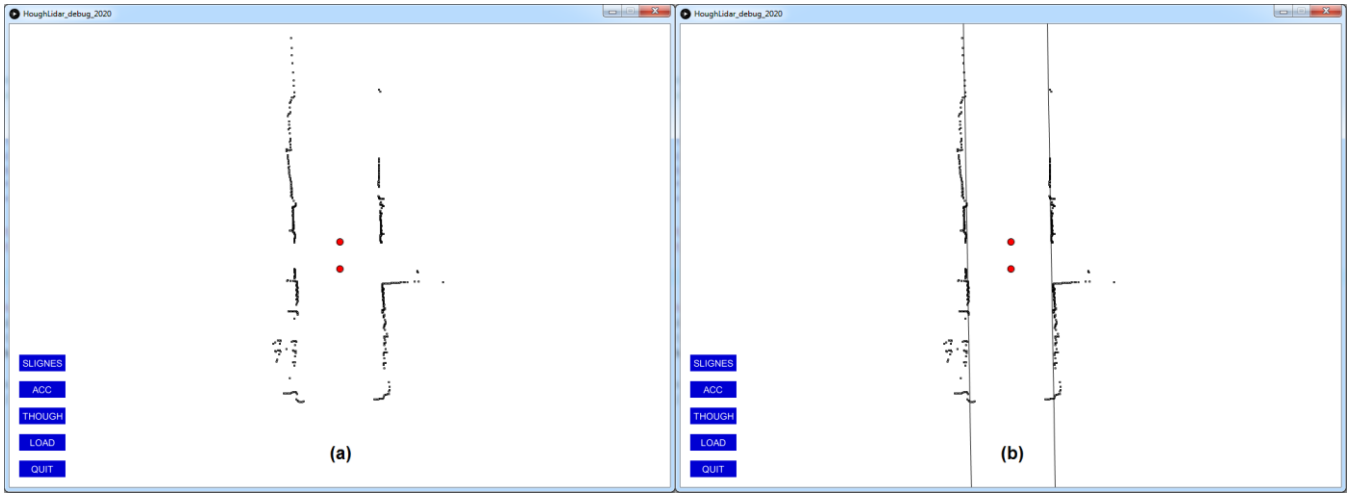


Figure 4.38 : Application of Hough Transform

The above plots were generated by the application of Hough transform to the acquired data in the open source platform 'Processing'. The above figure shows two lines on the sides of the lidars (red). These two lines act as the navigation reference for the robot. It is imperative to reflect upon the fact that the processing of Hough transform produces numerous lines in the feature space with different orientation and alignment. From this accumulation, only those lines have to be extracted which are suitable to act as navigation references, therefore a filtration process is required which is based on certain requirements.

- Lines are parallel (collinear)
- Lines fall within a certain width
- Lines fall under a certain angle threshold within the robot orientation

To conform to these requirements, the filtration is carried out by the algorithm which selects the optimal lines based on a minimization criterion. The filtration process executes sequentially and converges on this minimization criterion to select only those lines which fulfill this criterion.

Algorithm 3 : Hough Transform

Input: $Lidar(x, y)$ – A 2D feature space of Lidar data points

Output: HT(ρ, θ) – A 2D parametric space of sinusoids

Begin

//initialize the feature space and Hough space

1. measureLidar[][]; accumulator[][] ; cntrlines=0; cptPoints=0; w=const h=const
2. for pnts= 0 : 1 : 720 //Transform the Lidar points (generate virtual range image)
3. $X_{lid(i)} = d_i \cos(\theta_i)$, $x_{(i)} = \frac{w}{2} \pm (x_{lid(i)} \pm rob\_wid \times scale)$
4. $Y_{lid(i)} = d_i \sin(\theta_i)$, $y_{(i)} = \frac{h}{2} \pm (y_{lid(i)} \pm rob\_len \times scale)$
5. measureLidar[0][pnts]=x; measureLidar[1][pnts]=y; pnts++ //Store the points
6. end
7. for pnts = 0 : 1 : 720 //Start the voting in Hough space
8. x=measureLidar[0][pnts]; y=measureLidar[1][pnts];
9. if (x > 0 && x < w && y > 0 && y < h)//check the points if in the map then start voting the points
10. for θ_i = 0 : 1 : 180
11. $\rho = x\cos(\theta_i) + y\sin(\theta_i)$ //compute rho
12. $\rho_i = <\text{int}> (\frac{1}{2} + (\frac{\rho}{\rho_{\max}} + \frac{1}{2}) \times \rho_{\max index})$ //compute rho index
13. accumulator[θ_i][ρ_i]++ //increment the accumulator…i.e….increase votes
14. end
15. end
16. for $\rho_i = 0 : 1 : \rho_{\max index}$
17. for $\theta_i = 0 : 1 : 360$
18. if (accumulator[θ_i][ρ_i]>limit) //Check if peak detected in Hough space
19. ρ_i , θ_i //Extract the ρ and θ at this peak
20. $\alpha = -\frac{\cos(\theta_i)}{\sin(\theta_i)}$, $b = \frac{\rho}{\sin(\theta_i)} + \frac{h}{2} - \alpha.\frac{w}{2}$ //compute the parameters of line
21. $X_1 = -\frac{b}{\alpha}$, $X_2 = \frac{h-b}{\alpha}$, $X_m = \frac{X_1 + X_2}{2}$, $\theta_{line} = \tan^{-1}\left(\frac{X_2 - X_1}{h}\right)$ //compute line coordinates
22. saveData[0][cntrlines] = X_1 ; saveData[1][cntrlines] = X_2 ; saveData[2][cntrlines] = X_m ;

saveData[4][cntrlines] = θ_{line}, saveData[3][cntrlines]= accumulator[θ_i][ρ_i], cntrlines ++;

end //save all the lines

end

for line_1= 0 : 1 : cntrlines-1 //start comparing one pair of lines at each iteration

for line_2 = line_1+1 : 1 : cntrlines // X_m is …. saveData[2][cntrlines]

condition_1 = (X_m [line_1] - $\frac{w}{2}$) < 0 && (X_m [line_2] - $\frac{w}{2}$) > 0 //line_1 is on left, line_2 is right

condition_2 = (X_m [line_1] - $\frac{w}{2}$) > 0 && (X_m [line_2] - $\frac{w}{2}$) < 0 //line_1 is on right, line_2 on left

condition_3= (X_m [line_1] - X_m [line_2]) < +10% *(ref_width) //the lines are +10% of width apart

condition_4= (X_m [line_1] - X_m [line_2]) < -10% *(ref_width) //the lines are -10% of width apart

if ((condition_1 || condition_2) && condition_3 && condtion_4) // if either condition is true

determinant = (X_1 [line_1]- X_2 [line_1])-(X_1 [line_2]- X_2 [line_2]) // X_i is …. saveData[i][cntrlines]

$$\alpha_{line\_1} = \frac{h}{X_{2_{line\_1}} - X_{1_{line\_1}}}, \alpha_{line\_2} = \frac{h}{X_{2_{line\_2}} - X_{1_{line\_2}}}$$ //compute the slopes

$$b_{line\_1} = \frac{h}{X_{2_{line\_1}} - X_{1_{line\_1}}}.X_{1_{line\_1}}, b_{line\_2} = \frac{h}{X_{2_{line\_2}} - X_{1_{line\_2}}}.X_{1_{line\_2}}$$ //compute the intercepts

for pnts = 0 : 1 : 720

x = measureLidar[0][pnts]; y = measureLidar[1][pnts];

test_1 = $(x > \frac{y - b_{line\_2}}{\alpha_{line\_2}})$ && $(x < \frac{y - b_{line\_1}}{\alpha_{line\_1}})$ //if the lidar points are outside two lines

test_2 = $(x > \frac{y - b_{line\_1}}{\alpha_{line\_1}})$ && $(x < \frac{y - b_{line\_2}}{\alpha_{line\_2}})$ //if the lidar points are inside two lines

If (test_1 or test_2) cptPoints++; //if either test then increment points found

end

$$\text{criterion} = \frac{(1+\text{cptPoints}) \times (1+\text{fabs}(\text{determinant}))}{\text{peak}_{\text{line\_1}} + \text{peak}_{\text{line\_2}}}$$ //compute the minimization criterion

if (criterion < min_value)

min_value=criterion;

Line_1 = fabs ($X_{1_{line\_1}} - X_{2_{line\_1}}$) , Line_2 = fabs ($X_{1_{line\_2}} - X_{2_{line\_2}}$) //extract final lines

```
46.    if ( X_m[line_1] > w/2 )                    //check the midpoint
47.      X_right = X_m_line_1 , X_left = X_m_line_2 , θ_right = θ_line_1 , θ_left = θ_line_2    //Line_1 is right line
48.    else
49       X_right = X_m_line_2 , X_left = X_m_line_1 , θ_right = θ_line_2 , θ_left = θ_line_1    //Line_2 is right line
50.    end
51.  end
52. end
end
```

4.3.3.3.2.1 Transformation (Algorithm Execution)

In the algorithm the lidar scan points are first extracted and a virtual range image is formed figure (4.38 a), and the points are stored in an array constituting the feature space, lines (2-6). This feature space is use for the voting process in the Hough space. Each point from the feature space is checked if its in the limits of the range image (line 8) and forwarded to the voting loop. The voting loop computes the corresponding *'ρ'* and *'θ'* for each scan point (x, y), for the range of 'θ = 0 :180' producing sinusoids and incrementing value of each cell where these sinusoids pass, lines (7-15). Once the voting is finished the next step is to detect peaks in the accumulator array. This is achieved in the loop of lines (16-22). The loop runs for the range of '$\rho = 0 : \rho_{max}$' *and* 'θ = 0 :180'. If the count in a certain cell Cell[ρ_i][θ_i] is greater than a 'limit' then the 'indices' of that cell are extracted and 'ρ' and 'θ' are used to compute the parameters of the lines corresponding to this certain peak. The slope 'α' and intercept 'b' is computed from the equation

$$\rho = (x - \frac{w}{2})\cos\theta + (y - \frac{h}{2})\sin\theta \quad \Rightarrow \quad \alpha = -\frac{\cos(\theta_i)}{\sin(\theta_i)} \, , \, b = \frac{\rho}{\sin(\theta_i)} + \frac{h}{2} - \alpha . \frac{w}{2} \qquad (4.22)$$

the slope and intercept are then used to compute the start point (X_1), midpoint (X_m), the endpoint (X_2) and the orientation 'θ_{line}' of the line in the range image line(21) i.e. figure(4.41). The lines in the feature space corresponding to the peaks in the Hough space are saved in the form of these point parameters, lines (22,23). The 'limit' corresponds to the number of lines to be extracted greater than the limit. The use of this 'limit' selects the most significant lines in the data, since all the peaks above the limit will be selected for producing a line.

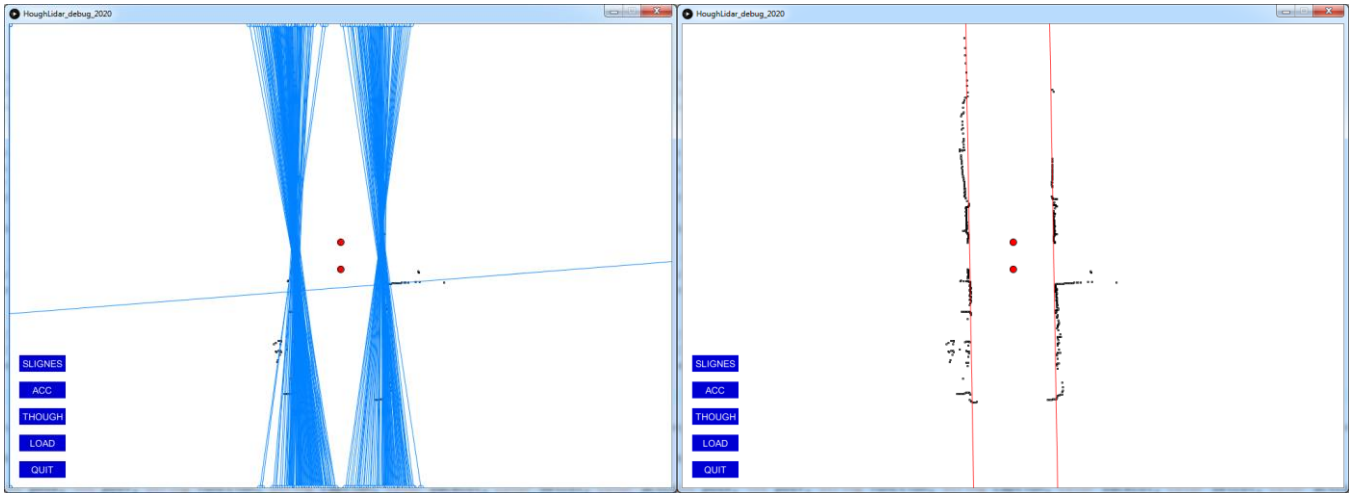


Figure 4.39 : Hough Lines for the Limit of 20 (Left) and Corresponding Two Lines Extracted (Right)

4.3.3.3.2.2 Filtration (Algorithm Execution)

The previous processing will produce hundreds of lines based on the peaks in the Hough space. These lines will have various orientations and placement in the range image. A filtration process is necessary since only those lines are of interest which are parallel to the robot. This filtration will extract the lines which are parallel and within an orientation angle threshold. From this filtration, the final two lines will be selected for the navigation reference of the robot. The filtration is based on the validation of certain conditions and a minimization criterion to select the best pair of lines. The process starts by comparing each pair of lines in the feature space, lines (26-end) and runs in a loop selecting one pair of lines on each iteration, lines (26,27). The conditions are formulated using the midpoints of the lines (X_m). 'Conditions_1' and 'Conditions_2' check if each line in the pair in the current iteration, falls on either left or right of the virtual range image. The validation of these conditions is necessary since the range image is constituted in the first place with the objective of having the lidars in the center of the image and the extracted lines on either left or right of the robot. Further, 'Conditions_3 and 'Conditions_4' check if the width between the lines is of ±10% of the specified rack width. The specified width for this thesis is 3 m, so the conditions must ensure it to be within 3.3 m and 2.7m. All the four conditions ensure that the lines under consideration in the current iteration, are on the left and right respectively and within the desired width threshold.

4.3.3.3.2.3 Extraction (Algorithm Execution)

When all the four conditions evaluate to true, then the lines are further checked for collinearity, i.e. if they are also parallel. This is done by computing the determinant of the two lines, in Algo-line (33), since if the determinant is or close to zero, the lines will be parallel.

Finally, the lines are subjected to the minimization criterion, if they satisfy the criterion then these will be saved as the final pair of lines.

4.3.3.3.2.3.1 Minimization Criterion (Algorithm Execution)

This minimization criterion is given by equation (4.23) and takes into account the collinearity of the lines and total number of scan points present inside the space between the lines.

$$\text{criterion} = \frac{(1+\text{total\_scan\_Points})\times(1+\text{fabs(determinant)})}{\text{Hough\_peak}_{\text{line\_1}} + \text{Hough\_peak}_{\text{line\_2}}} \tag{4.23}$$

This criterion is constituted by first considering the lines which are exactly parallel. Then determinant of two lines will be zero. Then it is more reasonable to consider the lines which are formed by the maximum number of points, i.e. the points which have voted maximum in the Hough space. This step will discard any shorter lines present in the data. The maximum votes (peaks) are considered in equation (4.23). The last step is to consider those lines which enclose a smaller number of points in the space between lines therefore the total scan points lying inside the lines are added to the equation. Equation (4.23) gives the ratio of the number of points in between the lines (scaled by the determinant) and the maximum number of points (peaks) that form the lines.

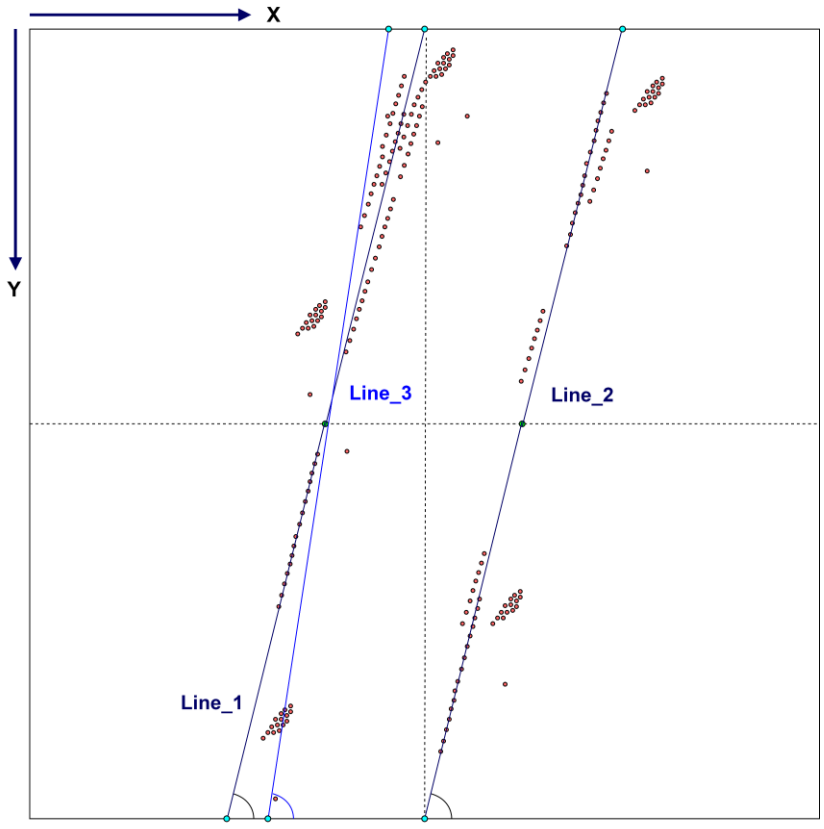


Figure 4.40 : Lines Consideration for Criterion Minimization

For example, considering the figure (4.40) above, selecting the blue line (Line_3) will enclose more number of scan points between 'Line_3' and 'Line_2'. Hence the minimization criterion ratio will be higher as compared to the 'criterion 'ratio' for 'Line_1' and 'Line_2'. The criterion ratio for the pair 'Line_1' and 'Line_2' will be lower as compared to the pair 'Line_2' and 'Line_3'. So, the former pair will be picked as the best lines at the end.

To check the scan data points present inside the lines, the slope and intercept of the lines in the current iteration is computed in the range image, lines (34,35) from the equations

$$y = \alpha x + \cancel{b}^{0} \Rightarrow \alpha = \frac{y_2 - y_1}{X_2 - X_1} \Rightarrow \alpha = \frac{h}{X_2 - X_1} \quad and \quad \cancel{y}^{0} = \frac{h}{X_2 - X_1}.X_1 + b \Rightarrow b = \frac{h}{X_1 - X_2}.X_1$$

and used in 'test_1' and 'test_2' in lines (38,39). The scan points are found inside the lines by comparing the x-coordinates of each point and each line on the x-axis of range image.

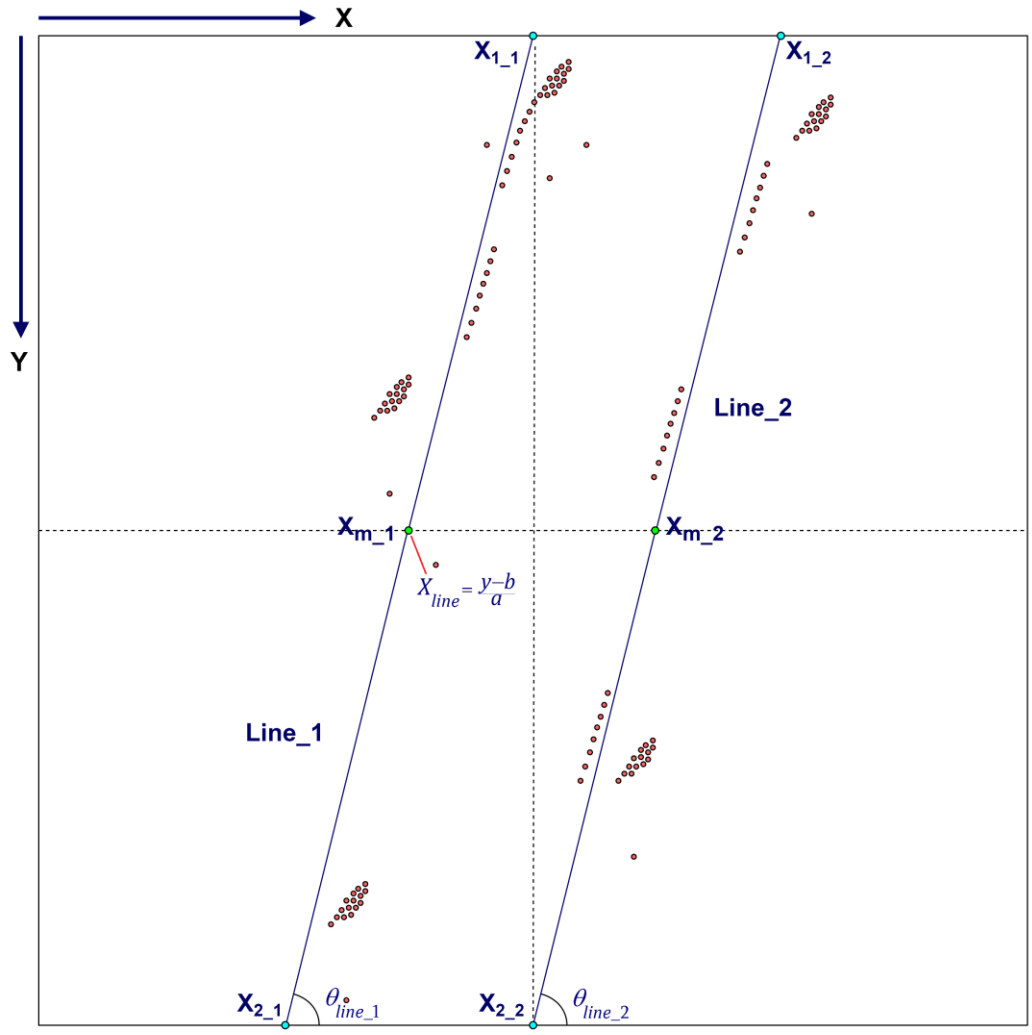


Figure 4.41 : Line Segments Representation in the Virtual Range Image

The validation of either test results in the existence of the points (cptPoints++). Once the total points are extracted, then the minimization criterion is computed. The final line segments are computed in line (45) as Line_1 = fabs ($X_{1_{line\_1}} - X_{2_{line\_1}}$) by satisfying the criterion to be less than a minimum value. The placement of these final lines in the range image for right and left, is checked by checking the midpoints with the center of the image, line (46). The final parameters ' X_{left} ' and ' X_{right} ' (midpoints) along with the orientations of the lines ' θ_{left} ' and ' θ_{right} ' are saved as the final reference parameters to be used in the control to align the robot parallel to the racks.

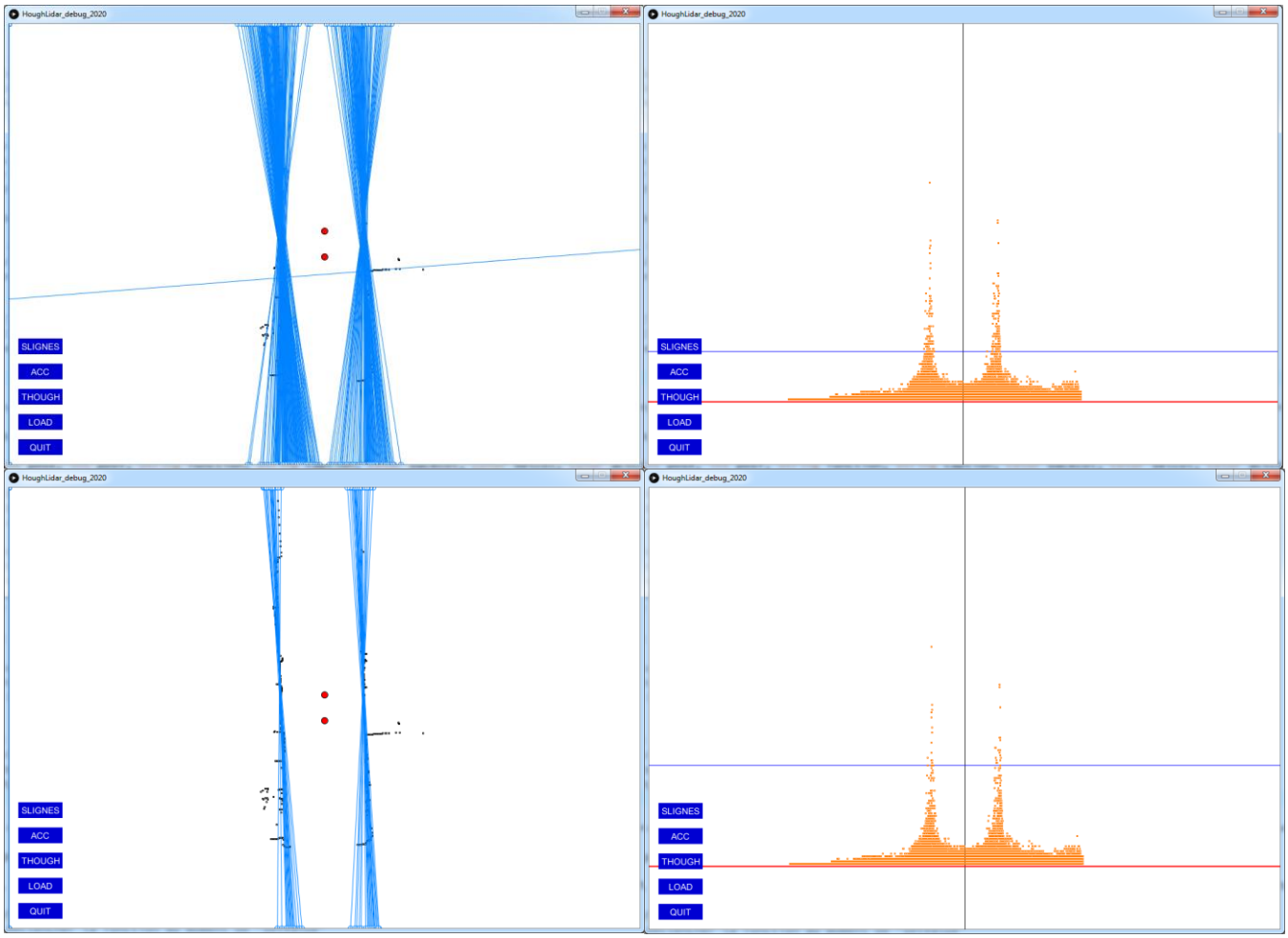


Figure 4.42 : Hough Lines Produced for Different Limits (Left) Corresponding Limits of Accumulators (Right)

In the above figure, the plots on the right show the 2D representation of the accumulator or the Hough space. The points in orange show the accumulation value of the peaks corresponding to ' ρ 'and ' θ '. Each point is a peak and the convergence towards the summit gives the maximum value of the peaks. The blue line represents the 'Limit' threshold in the algorithm. This threshold is a bound on the selection of maximum accumulated values. The

points (peaks) below the 'blue' lines have a smaller number of votes as compared to the points above the blue line. A threshold of '20' will select those peaks which have a vote more than '20', which is shown in the top right plot. All the points (peaks) above the blue line have votes more than 20 and are selected to produce the corresponding Hough lines in the top left plot. Similarly, the 'lower right' plot shows the selection of point (peaks) for the limit of '40' and corresponding Hough lines in the 'lower left' plot. It is clear from the 'left' plots, that raising the accumulators or peak selection limit produces a smaller number of lines. Therefore, the lines are filtered in this step by the 'Limit' threshold. Further the 'conditioning' and 'minimization' criterion selects the final two lines to be used for the navigation reference. Figure (4.43) below shows the lines extracted for the figure (4.42). The left plot in the figure (4.43) below gives the lines for the top left figure (4.42) for the limit of 20 and the right plot shows the lines for the limit of 40.

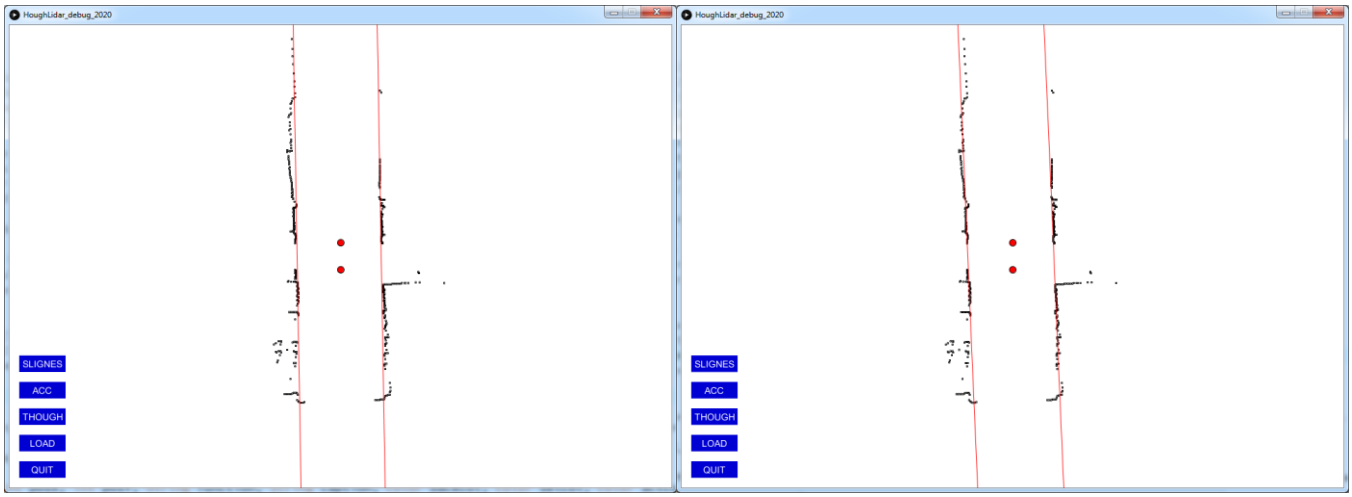


Figure 4.43 : Reference Lines for the Limit of 20 and Reference Lines for the Limit of 40

It is evident from the above figures that the two pair of lines extracted for fixing a limit of 20 and 40, are almost similar with a minor difference of orientation. But the two lines in a pair are exact (collinear) with respect to each other. The overall Hough algorithm extracts the final set of lines with filtration, which is necessary for two reasons. First the rack navigation requires a line reference to be used in real time which restricts the scope for any heavy computations for such a reference. The filtration reduces this computational time by considering only those lines which are above a limit, hence selecting a lower number of lines from a large spread. Second the use of the criterion produces the lines which are closest to the true reference (rack) and are parallel providing the exact reference for navigation. It is worth mentioning here that the 'limit' threshold is fixed manually. In the figure (4.44) below the same transform is applied to sparse data (scan points), which produces the accumulator with a lower spread (lower left plot). In this case the 'limit' (blue line) has to be decreased to produce lines, since a higher limit

fails to produce the lines in the feature space (lower right plot). A higher volume of data (scan points) will produce a large spread in the Hough space, and smaller volume will produce less. Therefore it is imperative to consider the 'limit' as being 'adaptive' rather being fixed manually, since the adaptive limit will be fixed automatically as a function of the spread of accumulation, and it will guarantee the finding of lines in the feature space when there is sparse data (lidar scan points) available to process. The development of an 'Adaptive Limit' has not been pursued in this thesis and has been delegated to a later research.

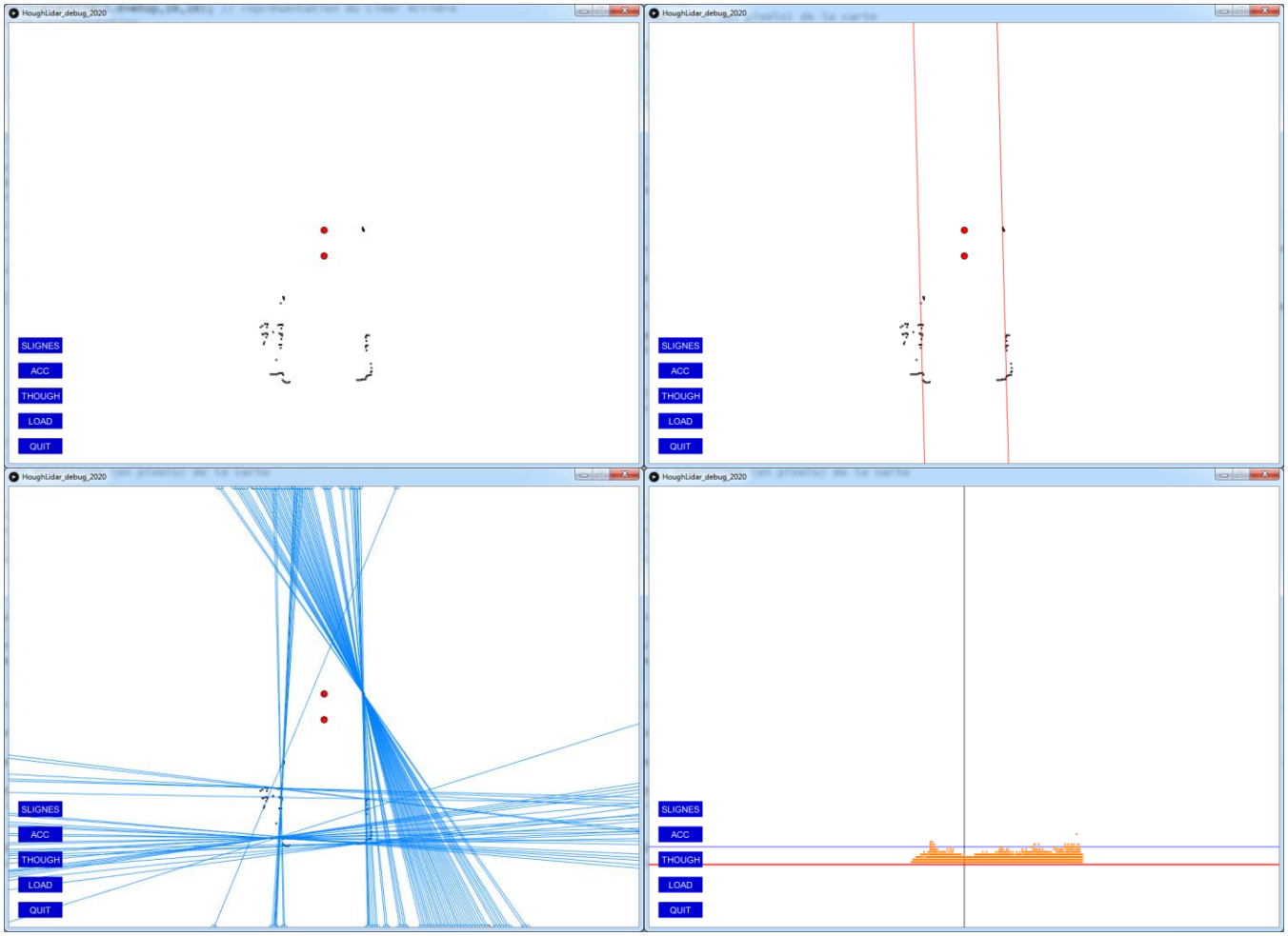


Figure 4.44 : Hough Lines (Left) and Corresponding Accumulation (Right) For Sparse Data (top left)

4.3.3.4 Control

Since it has already been mentioned in section (4.3.3.2) that the corridor following strategy requires a virtual reference to track and align the motion during navigation. This reference is provided by the lines extracted by the Hough transform algorithm in the previous section. The final parameters of the lines extracted, that is the mid-point '$X_{m_{line}}$' and the orientation 'θ_{line}' are used as reference parameters of control in the error compensation for the position alignment with respect to the rack.

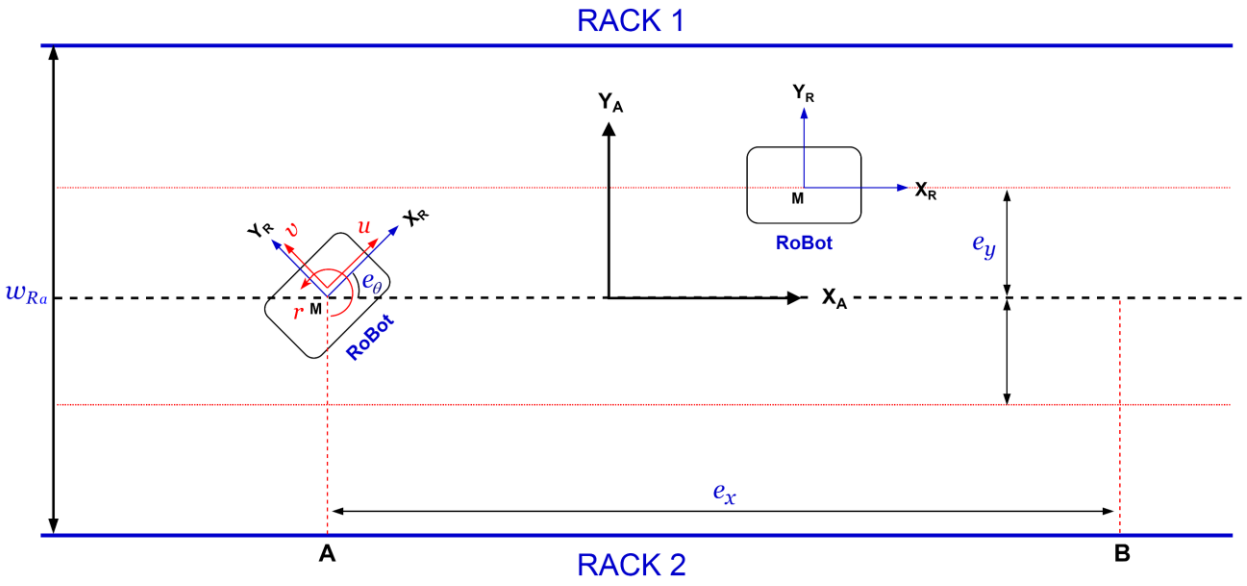


Figure 4.45 : Robot Modeling With Racks

In the above figure the notations are :

$(X_A, Y_A) \rightarrow$ Global or Rack Coordinate frame

$(X_R, Y_R) \rightarrow$ Robot coordinate frame

$(w_R) \rightarrow$ Width between the Racks

The hypothesis on the orientation error and lateral error are

$$hyp_{e_\theta} : -\frac{\pi}{3} \le e_\theta \le \frac{\pi}{3} \quad \text{and} \quad hyp_{e_y} : -\frac{w_{Ra}}{\cos(e_\theta)} \le e_y \le \frac{w_{Ra}}{\cos(e_\theta)}$$

Consider the figure (4.45) above in which the robot with its center 'M' is placed in front of the rack and it has to move from point 'A' to 'B'. Point 'A' is the starting point for the rack navigation and point 'B' will be provided by the global path planner. The objective of control is to align the robot frame (X_R, Y_R) with the global frame (X_A, Y_A) and then move to point 'B'. To achieve this the errors on the position are the lateral error 'e_{Y_R}', the orientation error 'e_θ' and the linear error 'e_{X_R}'. These errors in the robot coordinate frame are given as follows.

$$e_{Y_R} = d_{ref} - (X_{m\_line} - \frac{W_{rack}}{2}), \qquad e_{X_R} = X_A - X_R, \qquad e_{\theta_R} = \theta_{line} - \theta_{ref} \tag{4.24}$$

The objective of control is to regulate these errors to zero.

$$\lim_{t\to\infty}(e_{X_R}, e_{Y_R}, e_{\theta_R}) = 0 \tag{4.25}$$

For the motion control, the wheeled mobile robot (WMR) is a holonomic one and provides three control inputs to control three degrees of freedoms, i.e. the linear, lateral and orientation motions. The low-level controllers are able to generate the output velocities for the omnidirectional wheels of the robot for the three axes of motions. Considering the low-level controllers, as explained in section (4.3.2.1) the controllers are assumed to be perfectly tuned, meaning the performance is devoid of any static errors, damping or any oscillations. Therefore, under these assumptions, for the high-level control a first order dynamic model of the 'WMR' can be considered as before

$$u + \tau\dot{u} = Ku_c \tag{4.26}$$

where 'u_c' is the control input.

Considering the state space of the system the state, control and output vectors are given by

$$x = [u \quad v \quad r]^T \quad , \quad u = [u_c \quad v_c \quad r_c]^T \quad , \quad y = [u \quad v \quad r]^T \tag{4.27}$$

The full state of the system is given by

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \tag{4.28}$$

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} -1/\tau_u & 0 & 0 \\ 0 & -1/\tau_v & 0 \\ 0 & 0 & -1/\tau_r \end{bmatrix} . \begin{bmatrix} u \\ v \\ r \end{bmatrix} + \begin{bmatrix} k_u/\tau_u & 0 & 0 \\ 0 & k_v/\tau_v & 0 \\ 0 & 0 & k_r/\tau_r \end{bmatrix} . \begin{bmatrix} u_c \\ v_c \\ r_c \end{bmatrix} \tag{4.29}$$

$$C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad D = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

'τ_u', 'τ_v' and 'τ_r' are the time constants whose value was specified to be 0.02 after the experiments. 'k_u', 'k_v' and 'k_r' are the static gains and their value is '1' since the reference control inputs are given through 'ROS' in m/s. The navigation between the racks has to be done to go from 'A' to 'B', therefore the compensation of position error 'e_X' along the 'X_A' axis will be done at maximum velocity 'u_0' lower then '$u_{\max}$', i.e. ($u_0 < u_{\max}$). Likewise, the compensation of the position error 'e_Y' along the 'Y_A' axis will be done at a velocity ($v_0 < v_{\max}$) and compensation of orientation error 'e_θ' about the 'Z_A' axis will be done at a velocity ($r_0 < r_{\max}$).

For this the controller proposed is

$$\dot{e}_X = T_{u_x} - (\tau_u \dot{u} + u) = T_{u_x} - k_u u_c = f(e_x, e_y, e_\theta) = -k_u u_0 \frac{1 - e^{-\lambda_x e_x}}{1 + e^{-\lambda_x e_x}} . \left(1 - \left| \frac{e_y}{y_A} \right| . \cos(e_\theta) \right)$$

(4.30)

$$\dot{e}_y = T_{v_y} - (\tau_v \dot{v} + v) = T_{v_y} - k_v v_c = g(e_y, e_\theta) = -k_v v_0 \frac{1 - e^{-\lambda_y e_y}}{1 + e^{-\lambda_y e_y}} . \left(\cos(e_\theta) \right) \qquad (4.31)$$

$$\dot{e}_r = T_{r_z} - (\tau_r \dot{r} + r) = T_{r_z} - k_r r_c = h(e_\theta) = -k_r r_0 \frac{1 - e^{-\lambda_z e_\theta}}{1 + e^{-\lambda_z e_\theta}} \qquad (4.32)$$

In the above equations 'T_{u_x}', 'T_{v_y}', and 'T_{r_z}' are the target velocities but they are zero since point 'B' is a fixed point and the objective of the control is not to track and pursue the target, but rather just reach it. The sigmoid functions are used to get a smooth behavior and 'λ_x', 'λ_y' and 'λ_z' and are specified to reduce the speed just before 1 to 2 meters from the goal to avoid overshooting, whereas 'k_u=1', 'k_v=1' and 'k_r=1'. After derivations the final control inputs to control the motion in lateral, linear and rotational directions are

$$u_n = \frac{\tau_u . u_{n-1}}{T + \tau_u} + \frac{T}{T + \tau_u} k_u . u_0 . \frac{1 - e^{-\lambda_x e_x}}{1 + e^{-\lambda_x e_x}} . \left(1 - \left| \frac{e_y}{w_R} \right| \right) . \cos(e_\theta) \qquad (4.33)$$

$$v_n = \frac{\tau_v}{T+\tau_v}.v_{n-1} + \frac{T}{T+\tau_v}k_v.v_0.\frac{1-e^{-\lambda_y e_y}}{1+e^{-\lambda_y e_y}}.\cos(e_\theta) \tag{4.34}$$

$$r_n = \frac{\tau_r}{T+\tau_r}.r_{n-1} + \frac{T}{T+\tau_r}k_r.r_0.\frac{1-e^{-\lambda_z e_\theta}}{1+e^{-\lambda_z e_\theta}} \tag{4.35}$$

In the above control laws the sigmoid functions act as saturations. The control equations (4.33) and (4.34) are influenced by the term 'e_θ'. This term will enforce the robot to correct its orientation first and equation (4.35) will be in effect, since as long as the robot is not in the correct orientation, i.e. parallel to the rack, it will neither move forward significantly or move lateral. As soon as the orientation error goes to zero, it will move lateral to bring itself close to the required reference, since equation (4.33) will have a small effect due to the presence of '$\left(1-{}^{e_y}/_{w_R}\right)$' term and the forward motion will be limited. When the robot brings itself at the required reference from the side, this term will vanish and the equation (4.33) will be in effect to move the robot forward. The figure below presents the speed scaled by sigmoid functions for the robot linear 'U_x',lateral 'V_y' and rotational velocity 'R_z'. The values of 'λ_x' , 'λ_y' and 'λ_z' were empirically specified to be '5' after various trials for a smooth behavior.

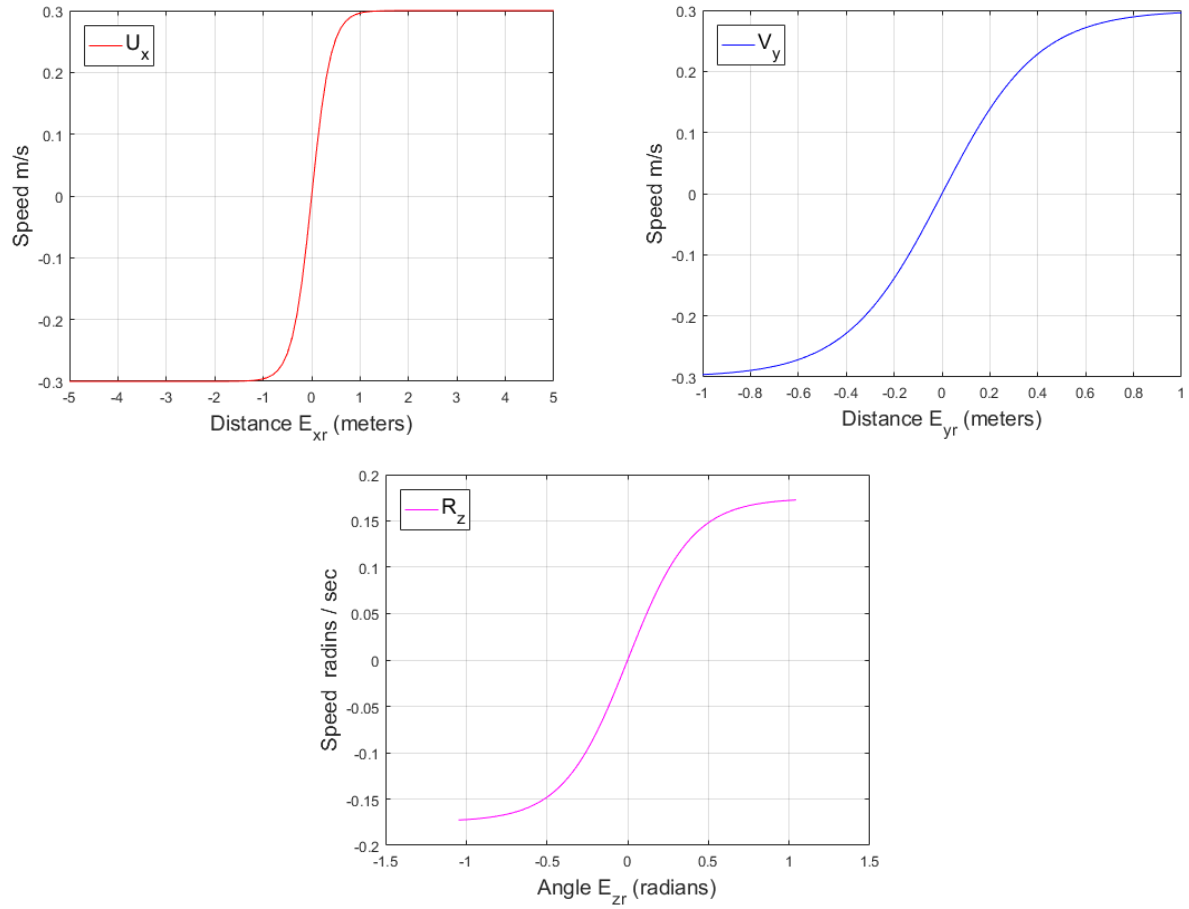


Figure 4.46 : Sigmoid Functions for Smooth Speed Control

4.3.3.5 Rack Tracking

The objective of tracking is to essentially align the robot with the rack. The robot must converge to the required alignment irrespective of the initial placement. For the alignment it should acquire both the specified separation distance and the orientation from the rack. This is achieved by the control algorithm which enables the robot to track the desired parameters in real time and converge it to the desired configuration. The effectiveness of the control is evaluated by placing the robot in particular initial configurations to execute motion subsequently, enabling the robot to converge to the desired configuration to track the rack consistently. These scenarios along with simulated results are discussed in detail below.

4.3.3.5.1 Scenario 1

In this scenario the robot is tested to align itself to the required separation distance from the rack without any orientation errors. The initial and desired parameters of the robot configuration are given in the table (6) below. For the initial placement the robot is already parallel to the rack i.e. '$e_\theta = 0$', but requires to converge to 'd_{ref}'. A visual description of the scenario is given in the figure below. The robot with its center 'M' is placed in the middle of the racks. The dotted red lines indicate the required separation distance to achieve and also the Hough lines with its orientation 'θ_{line}' with respect to the rack.

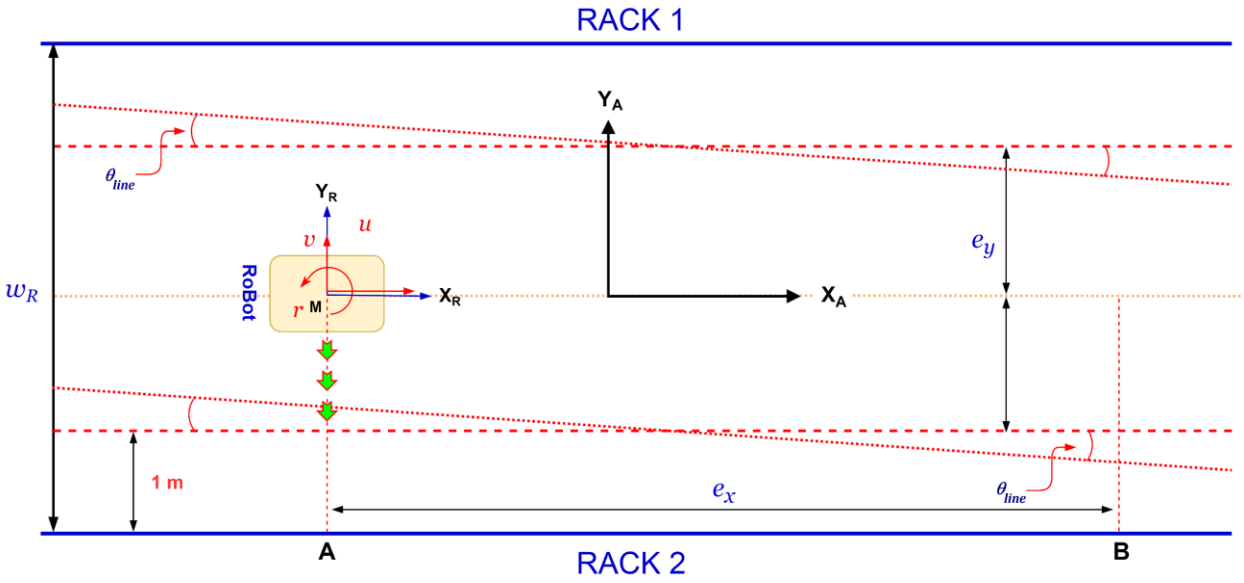


Figure 4.47 : Scenario 1 - Robot Lateral Motion for Rack Alignment

A plausible motion of the robot is shown with the green arrows in the above figure. The robot only executes a lateral motion to align itself to the required reference. This virtual reference is provided by the Hough lines in real time in the feedback loop of control enabling the robot to

converge to the desired distance. A test was performed in real time to evaluate the performance of control for this motion. An overview of the resulting motion and configuration evolution of the robot is given in plots in the figure below.

Rack Width	X_i	X_f	Y_i	Y_f	θ_i	θ_f	E_{xr}	E_{yr}	E_{zr}
3 m	0	0	0	1 m	0	0	0	1 m	0

Table 7 : Robot Configuration Parameters

In the above table 'X_i' and 'Y_i' are the initial placement of the robot. 'X_f' and 'Y_f' are the final configuration to acquire. 'θ_i' is the initial orientation angle of the robot while 'θ_f' is the final orientation angle and is equal to 'θ_{line}'. 'E_{xr}' , 'E_{yr}' and 'E_{zr}' are according to equations (4.26).

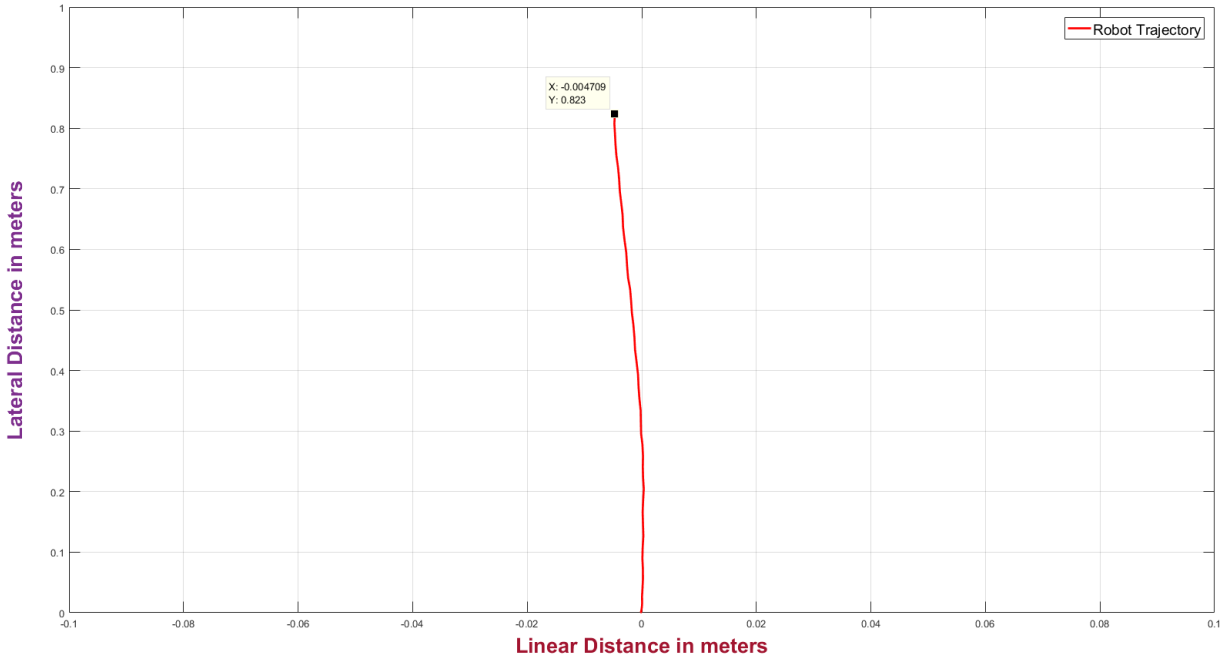


Figure 4.48 : Robot Lateral Motion Trajectory

The plot above shows the trajectory evolution of the robot for the lateral motion executed to acquire the desired reference. It is clear from the plots that the robot only moves in the lateral direction and there is no movement in the horizontal direction.

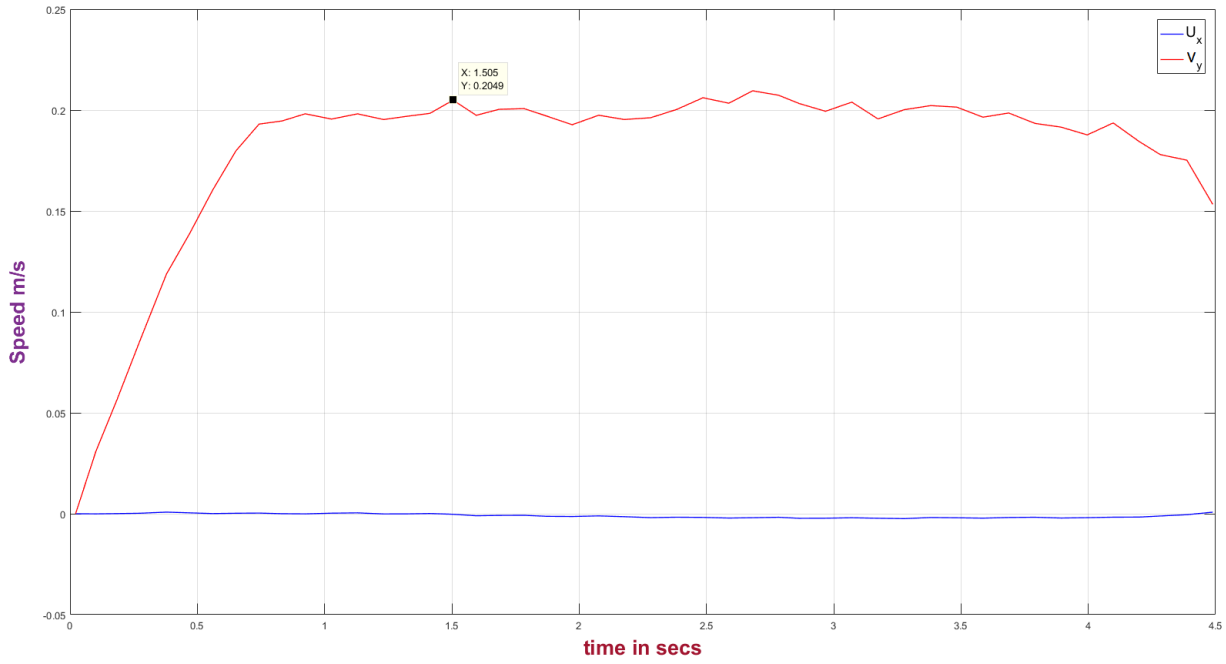


Figure 4.49 : Robot Lateral and Linear Speeds

Similarly, the plot in figure (4.49) above shows the robot speed in the lateral direction 'y-axis of robot frame' at the same time indicating the first order response of the control. The plot in figure (4.50) shows a lateral error ' E_{yr}' of almost 1 m. This is the error between the required reference to achieve 'd_{ref}' and the current position in the robot reference or the range image containing the Hough lines. This error converges to zero with time which shows that the robot is able to acquire the required reference from the rack.

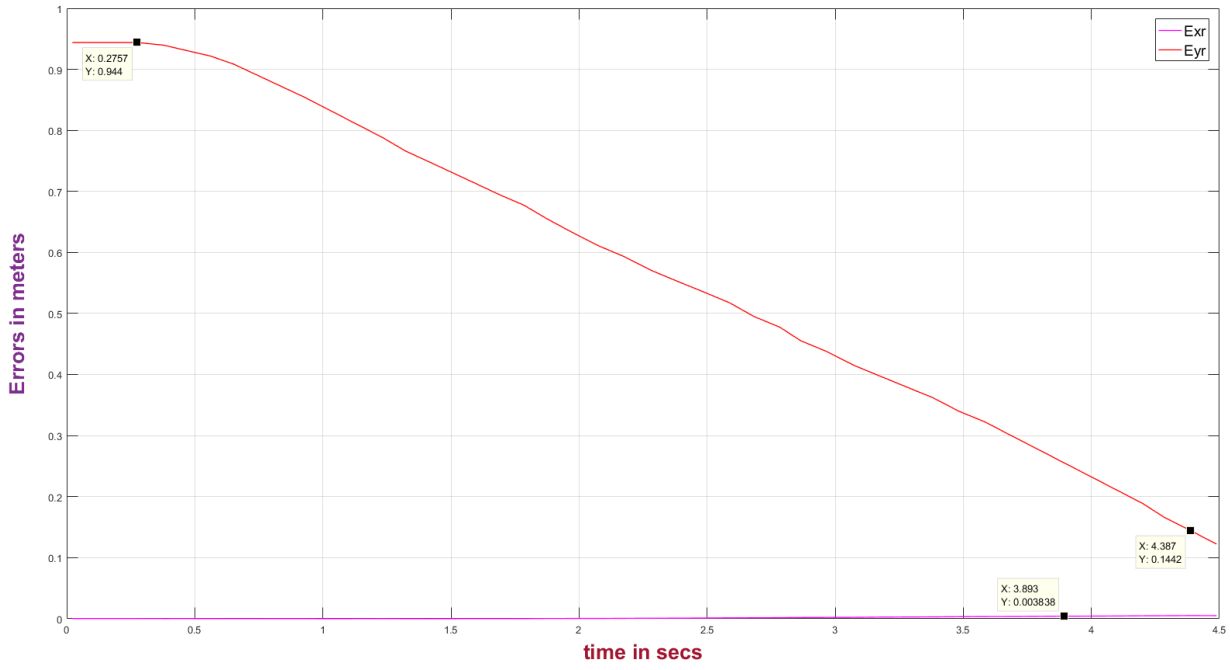


Figure 4.50 : Robot Lateral Alignment and Linear Distance Errors

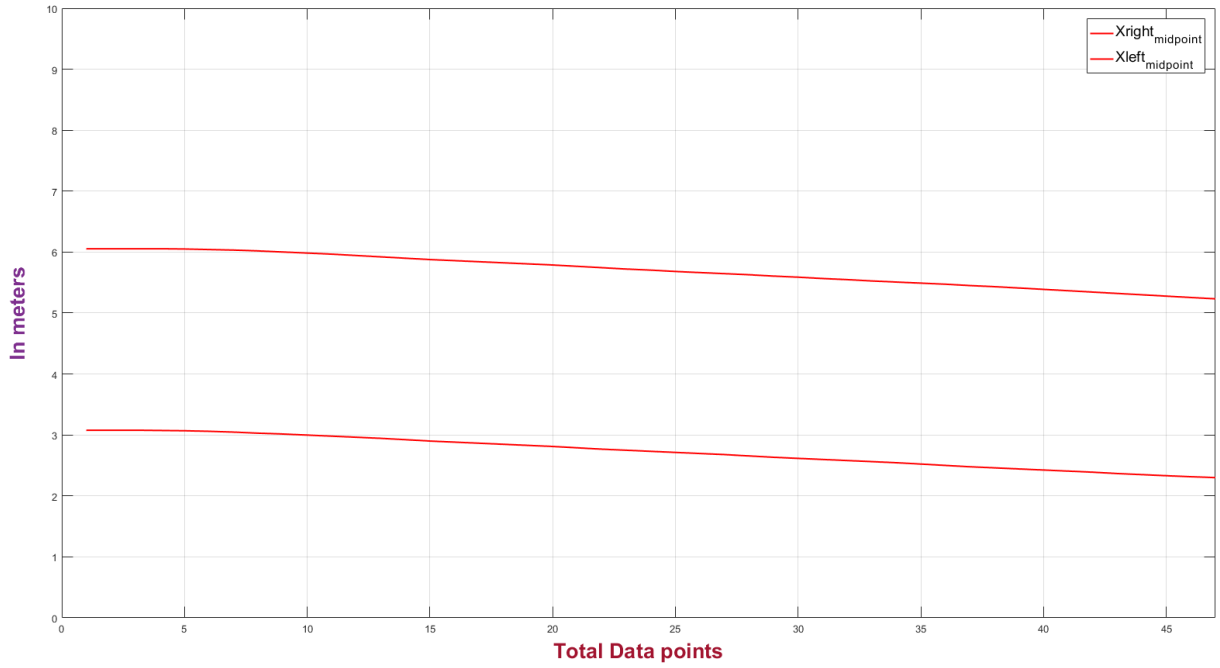


Figure 4.51 : Evolution of Hough Lines tracking (X_{m_1}, X_{m_2})

In the figure above '$X_{right_{midpoint}}$' and '$X_{left_{midpoint}}$' are the midpoints (X_{m_1}, X_{m_2}) of each Hough lines according to figure (4.42). 'X_{m_line}' is the mid-point of a valid Hough line found in the range image and is the tracking reference or the tracking point for the control. The plot above gives the evolution of these midpoints for the full motion of the robot. The y-axis gives the scale in meters and the data has been scaled down to fit the plot. The significant observation in the above plot is that the evolution of each midpoint presents a line, and the width between these two lines is within 3 meters (on y-axis) corresponding to the actual width between the racks. The figure (4.52) shows the 'difference' between the two lines of the above plots.

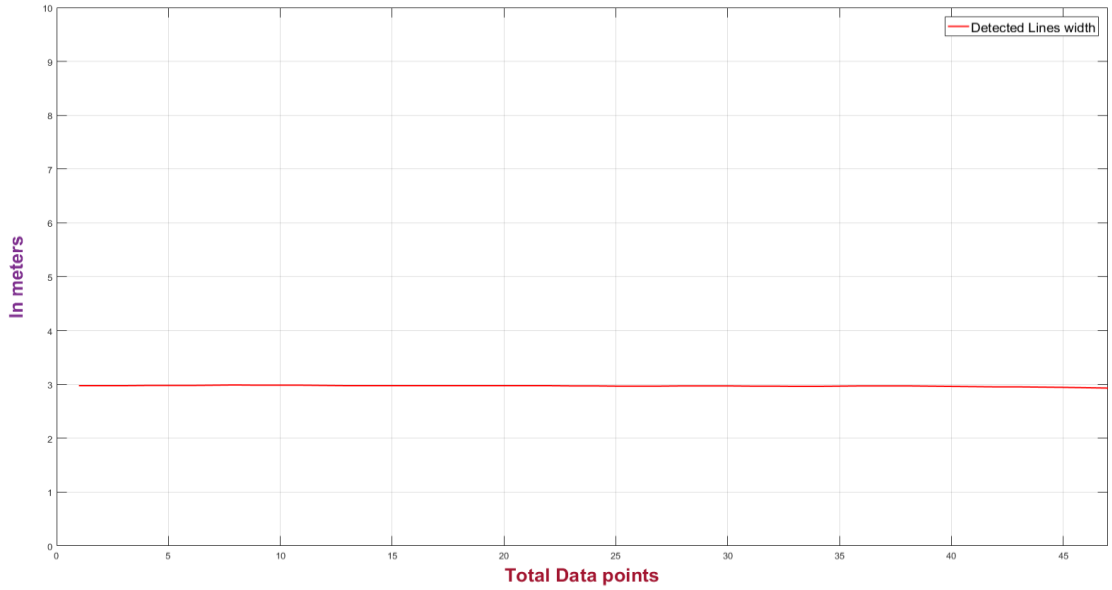


Figure 4.52 : Difference of Hough lines Midpoints $X_{right_{midpoint}}$ and $X_{left_{midpoint}}$

4.3.3.5.2 Scenario II

In this scenario the robot is tested to align itself to the required separation distance and then continue to move along the rack to the next point while tracking the separation reference. For the initial placement the robot is already parallel to the rack i.e. '$e_\theta = 0$', but requires to converge to 'd_{ref}' and 'X_a' distance to move, that is to align itself at 1 m with respect to the rack and then move a distance of 5 meters along the rack. A visual description of the scenario is given in the figure below. The robot with its center 'M' is placed in the middle of the racks. The dotted red lines indicate the required separation distance to achieve and also the Hough lines with its orientation 'θ_{line}' with respect to the rack.

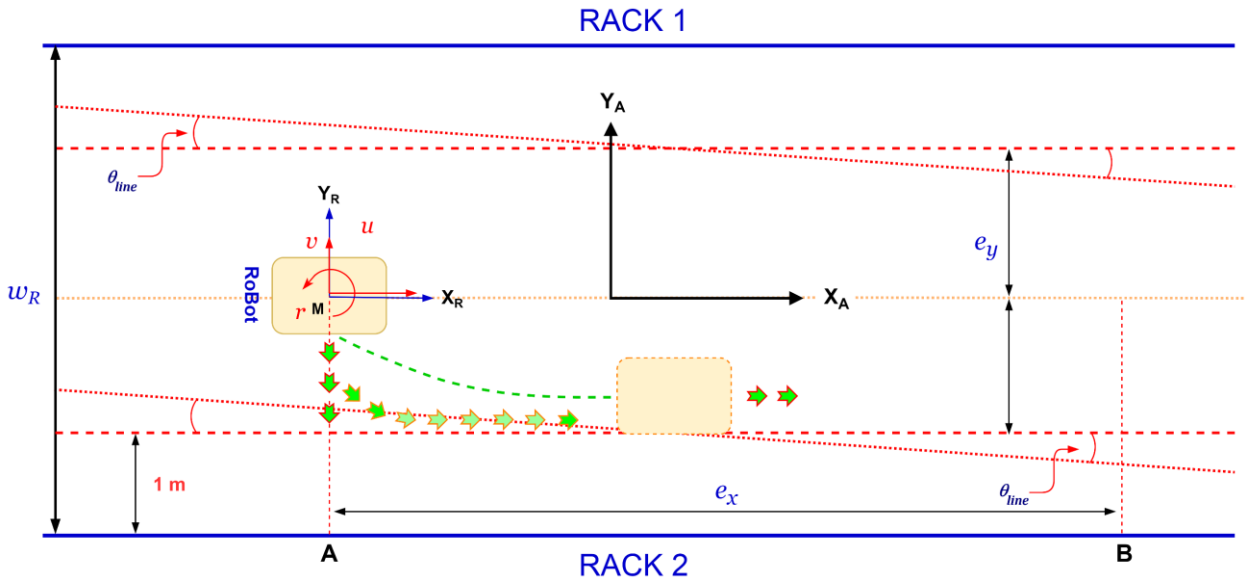


Figure 4.53 : Scenario 2 - Robot Lateral and Linear Motion for Rack Tracking

A plausible motion of the robot is shown with the green arrows in the above figure. The green dotted line presents the plausible trajectory of the robot. The robot executes a lateral and linear motion simultaneously to align itself to the required reference distance and then move along the rack to reach the desired point. This virtual reference to track during this motion is provided by the Hough lines in real time. A test was performed in real time to evaluate the performance of control for this motion. An overview of the resulting motion and configuration evolution of the robot is given in plots in the figure below.

Rack Width	X_i	X_f	Y_i	Y_f	θ_i	θ_f	E_{xr}	E_{yr}	E_{zr}
3 m	-2.5 m	2.5 m	1.5 m	1.0 m	0	0	5 m	1 m	0

Table 8 : Robot Configuration Parameters

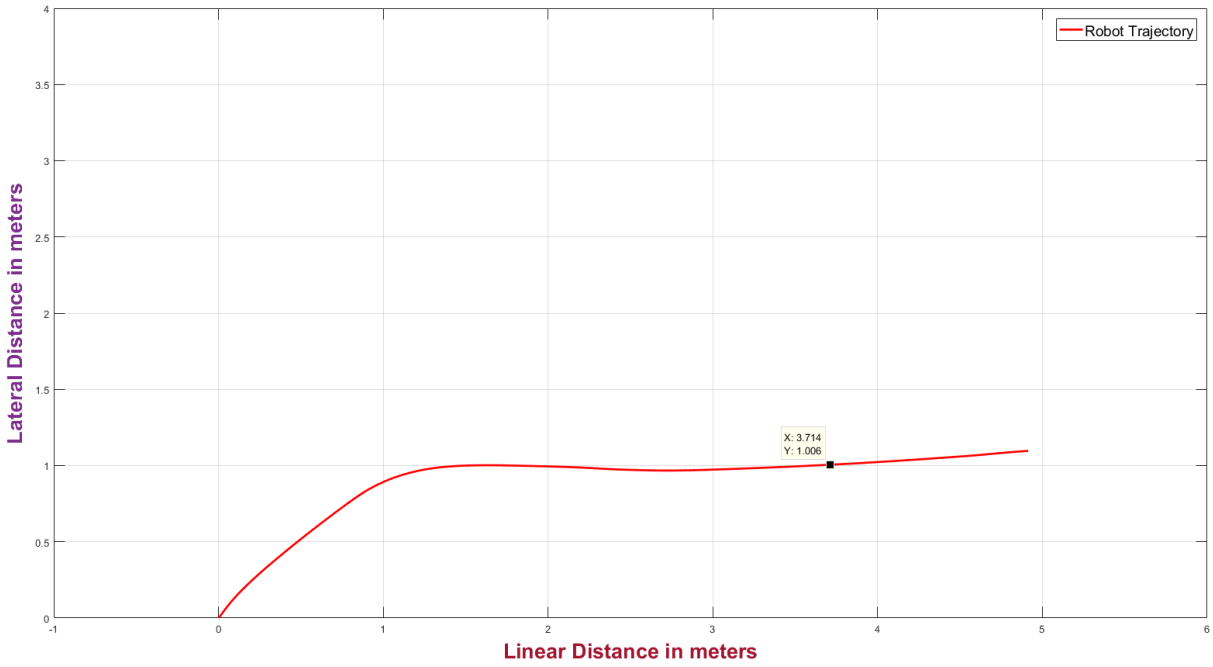


Figure 4.54 : Robot Lateral and Linear Motion Trajectory

The above plot shows the trajectory evolution of the robot for the lateral and linear motion executed to acquire the desired references. It is clear from the plot that the robot moves in the lateral and linear direction at the same time for a distance of 1 meter along the x-axis. After it has converged to the 'd_{ref}' of 1 m along the y-axis it continues to move along the x-axis for 4 meters. Similarly, the plot below shows the lateral and linear robot speeds for the full motion.

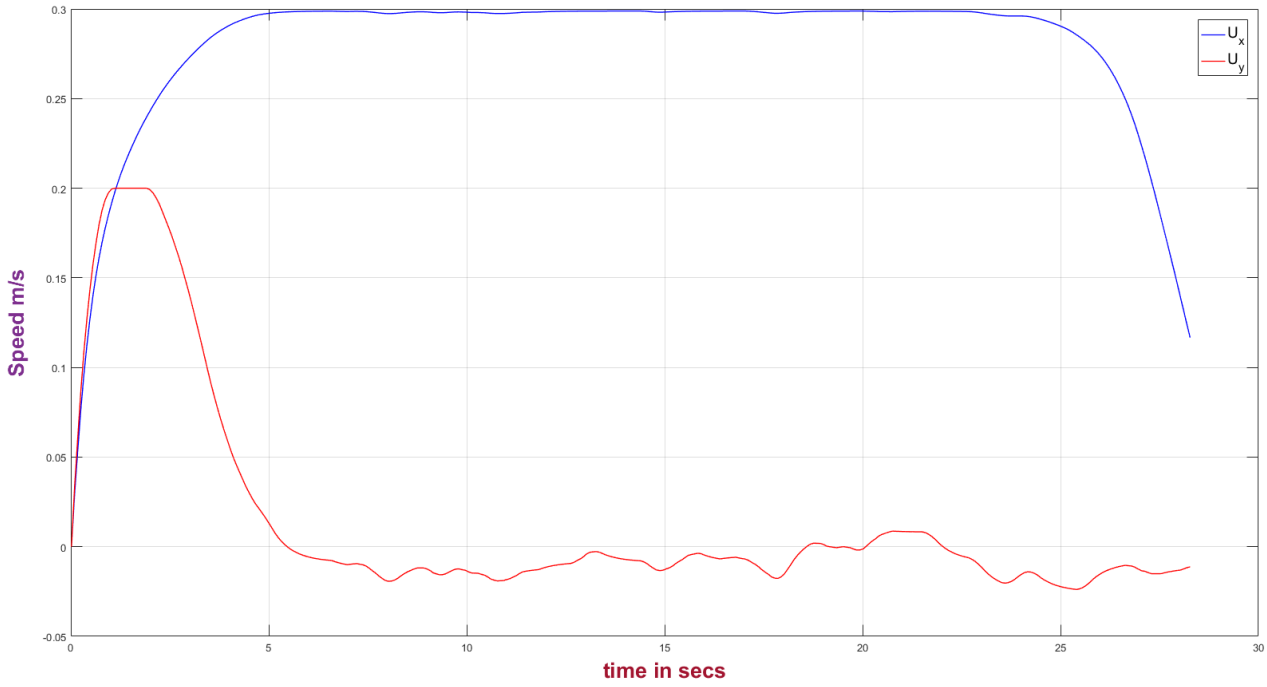


Figure 4.55 : Robot Lateral and Linear Speeds

The increase in the lateral speed in the above plots drops down after 4 seconds. For the first 4 seconds the lateral speed is 0.2 m/s which is when the robot is laterally aligning itself to the rack. At the same time the linear speed is increasing which shows that the robot is also moving forward while moving laterally. After 4 seconds the lateral speed stays close to zero with little fluctuation which indicates the lateral alignment motion is also in 'effect' along the linear move. The linear speed increase to 0.3 m/s during the first 6 seconds and starts to decreases before the robot reaches the goal point.

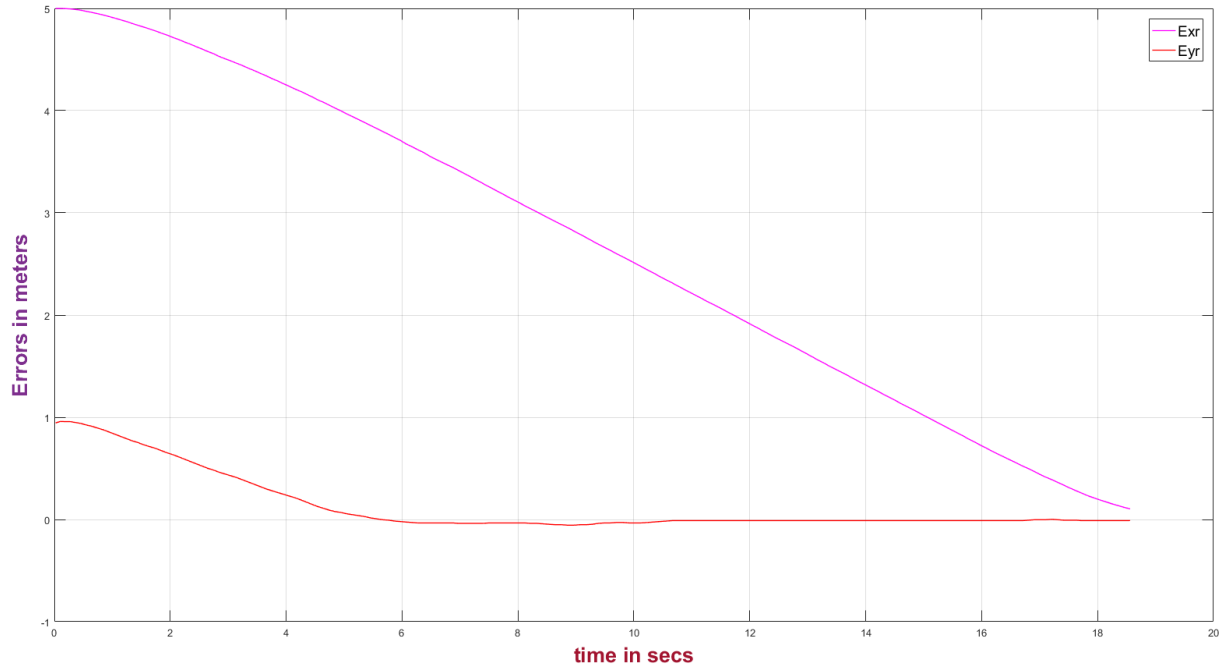


Figure 4.56 : Robot Lateral Alignment and Linear Distance Errors

The above plot shows the lateral and linear errors. The lateral error ' E_{yr}' is 1m in the start which is the reference separation to acquire from the rack side, while the linear distance error ' E_{xr}' is 5 meters, which is the linear distance the robot has to move. In congruence to figure (4.54) the lateral error decreases in first 4 seconds which indicates the convergence of the robot to required separation from the rack while the linear error gradually decrease to zero as the robot moves along the rack to reach the specified point. The plot below in figure (4.57) gives the evolution of the line midpoints for the full motion of the robot to achieve its target. The evolution of each midpoint represents a line, and the width between these two lines is within 3 meters (on y-axis) corresponding to the actual width between the racks. This is verified by figure (4.58) which shows the 'difference' between the two lines, i.e. difference between '$X_{right_{midpoint}}$' and '$X_{left_{midpoint}}$'.

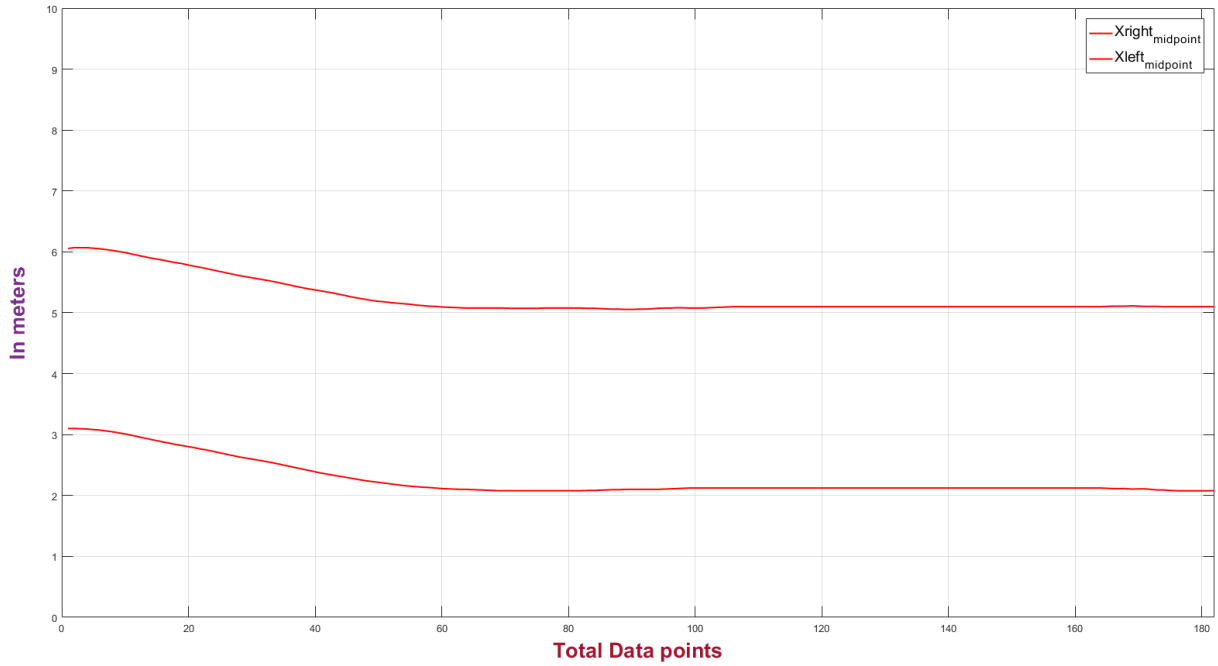


Figure 4.57 : Evolution of Hough Lines Tracking Points (X_{m_1}, X_{m_2})

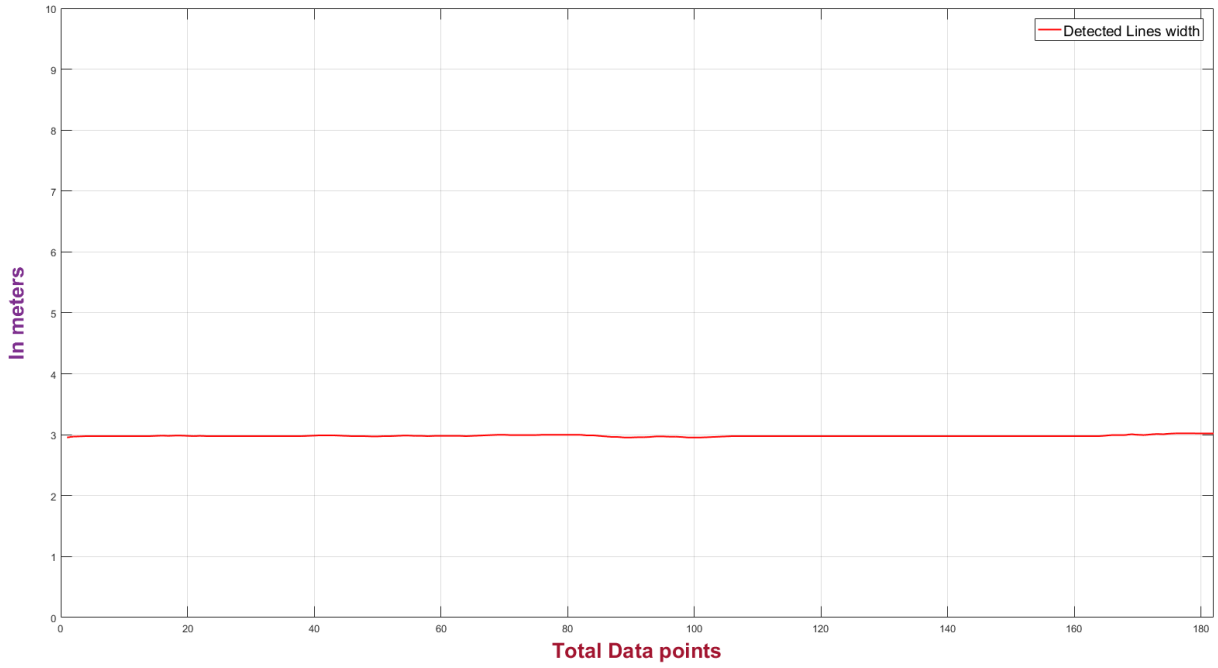


Figure 4.58 : Difference of Hough lines Midpoints $Xright_{midpoint}$ and $Xleft_{midpoint}$

The above plot is just the difference between the lines of plot of figure (4.57) indicating the Hough lines detected are within the specified rack width.

4.3.3.5.3 Scenario III

In this scenario the robot is tested to align itself to the required separation distance and then continue to move along the rack to the next point while tracking the separation reference. For the initial placement the robot is not parallel to the rack and '$e_\theta > 0$' that is '$\theta_{line} > 0$'. It therefore requires to converge first to 'θ_{ref}' then 'd_{ref}' and then the distance to move along the rack. The robot first has to make itself parallel to the rack, then align itself at 1 m with the rack and then move a total distance of 5 meters along the rack. A visual description of the scenario is given in the figure below. The robot with its center 'M' is placed in the middle of the racks. The dotted red lines indicate the required separation distance to achieve and also the Hough lines with its orientation 'θ_{line}' with respect to the rack.

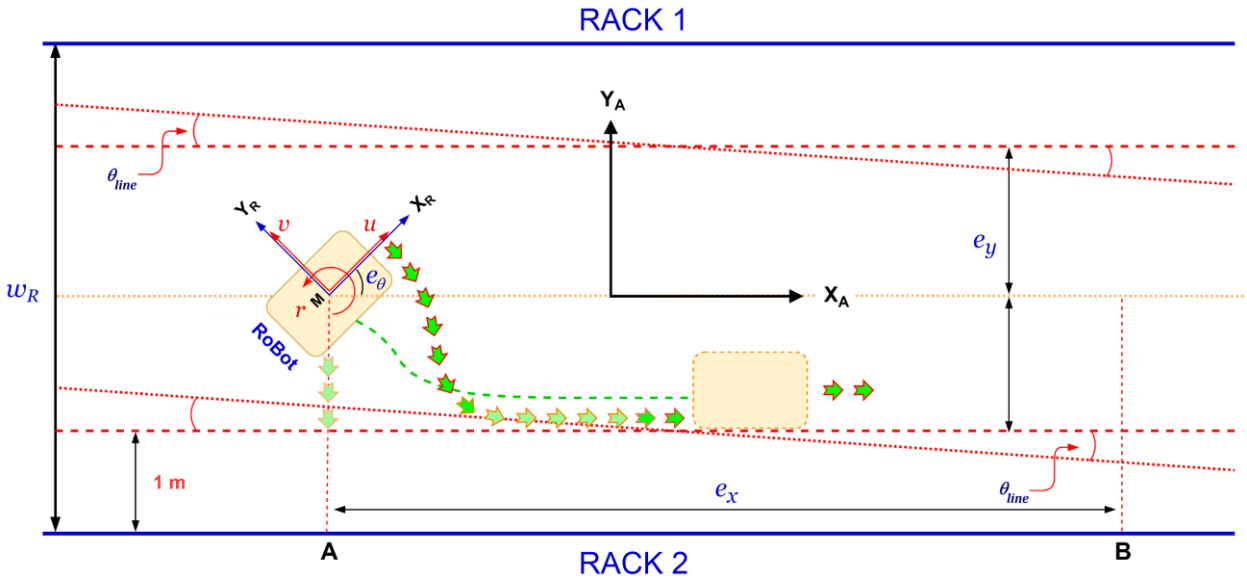


Figure 4.59 : Scenario 2 - Robot Rotational, Lateral and Linear Motion for Rack Tracking

A plausible motion of the robot is shown with the green arrows in the above figure. The green dotted line represents the probable trajectory of the robot. The robot executes a rotational, lateral and linear motion simultaneously to align itself to the required reference distance and then move along the rack to reach the desired point. The virtual reference to track during this motion is provided by the Hough lines in real time. A test was performed in real time to evaluate the performance of control for this motion taking into consideration the orientation error. This test was done to mimic the placement of the robot in a random configuration in front of the racks. The motion executed by the robot was the same as indicated by the green arrows which was observed physically. An overview of the resulting motion and configuration evolution of the robot is given in plots in the figure below.

Rack Width	X_i	X_f	Y_i	Y_f	θ_i	θ_f	E_{xr}	E_{yr}	E_{zr}
3 m	-2.5 m	2.5 m	1.5 m	1 m	25°	0	5 m	1 m	-25°

Table 9 : Robot Configuration Parameters

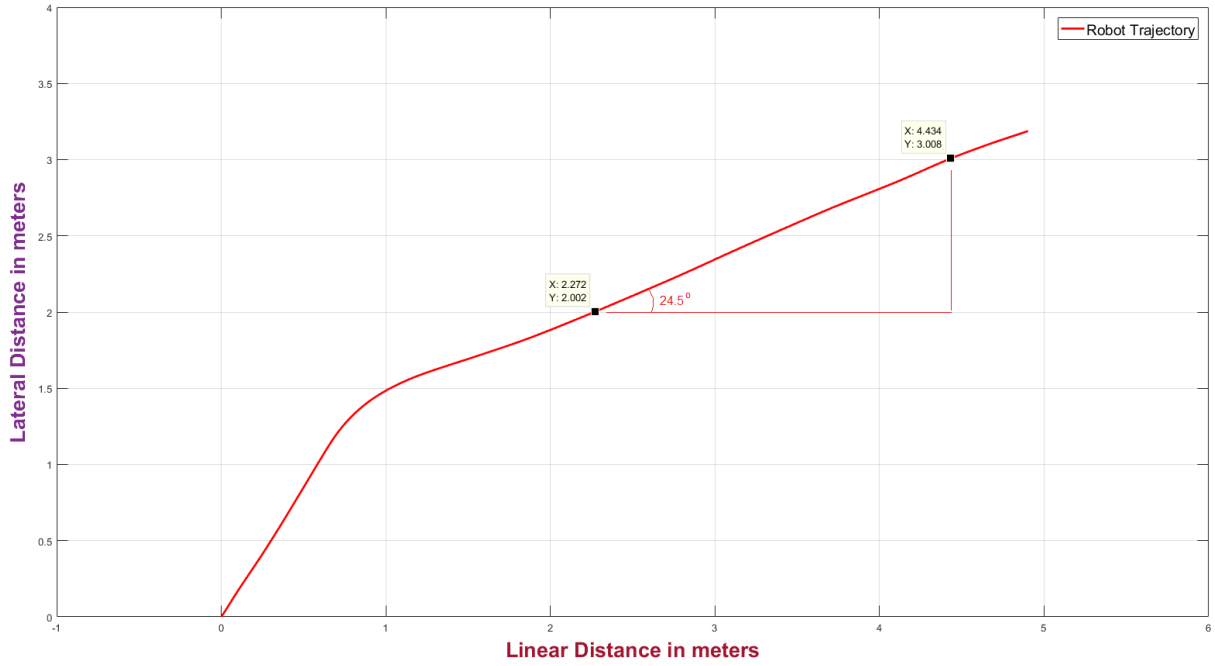


Figure 4.60 : Robot Rotational, Lateral and Linear Motion Trajectory

The above plot shows the trajectory evolution of the robot for the rotational, lateral and linear motion executed to acquire the desired references. The plot should be almost identical to figure (4.54), but instead it shows a linear rise after 1 meter. This is due to the fact that the robot rotates and then translates at the same time and therefore there is a bias of orientation present in the trajectory acquired. This trajectory is acquired from odometry of the robot in the 'robot' frame and not in the global frame. When this orientation bias (in figure above), is compensated the true trajectory of the robot is given below in figure (4.61), which gives a similar plot as in figure (4.54).

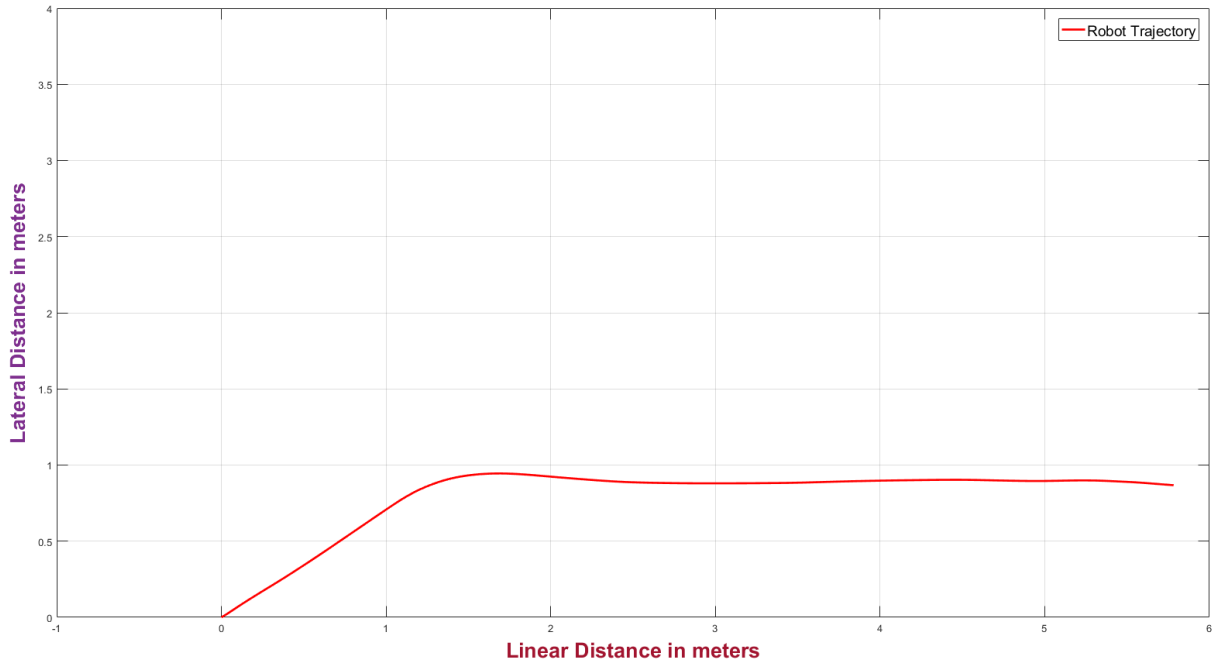


Figure 4.61 : Robot Motion Trajectory with Bias Compensation

The robot rotational speed is shown in figure below, while the lateral and linear speeds is given in figure (4.63). The rotational speed increases for the first one second to compensate for the orientation error and align to the correct orientation. The speed is negative due to rotation in counter-clockwise angle. The speed reduces under 5 seconds due to decrease of 'E_{zr}' error.

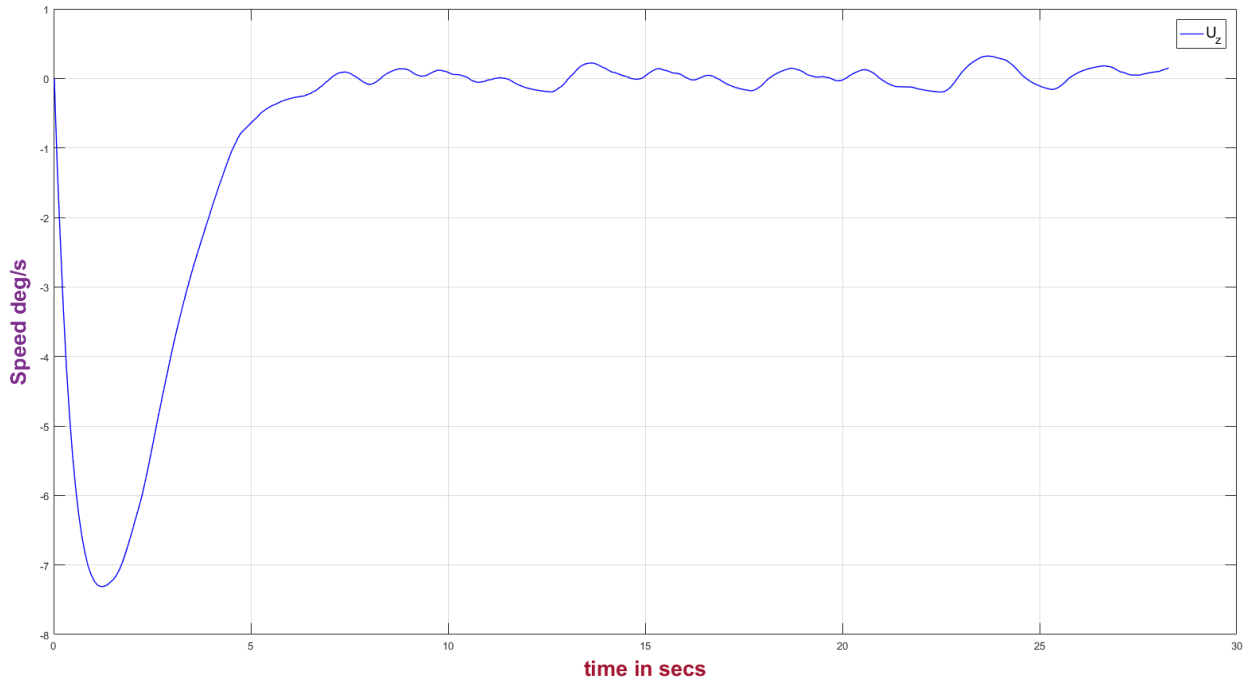


Figure 4.62 : Robot Rotational Speed

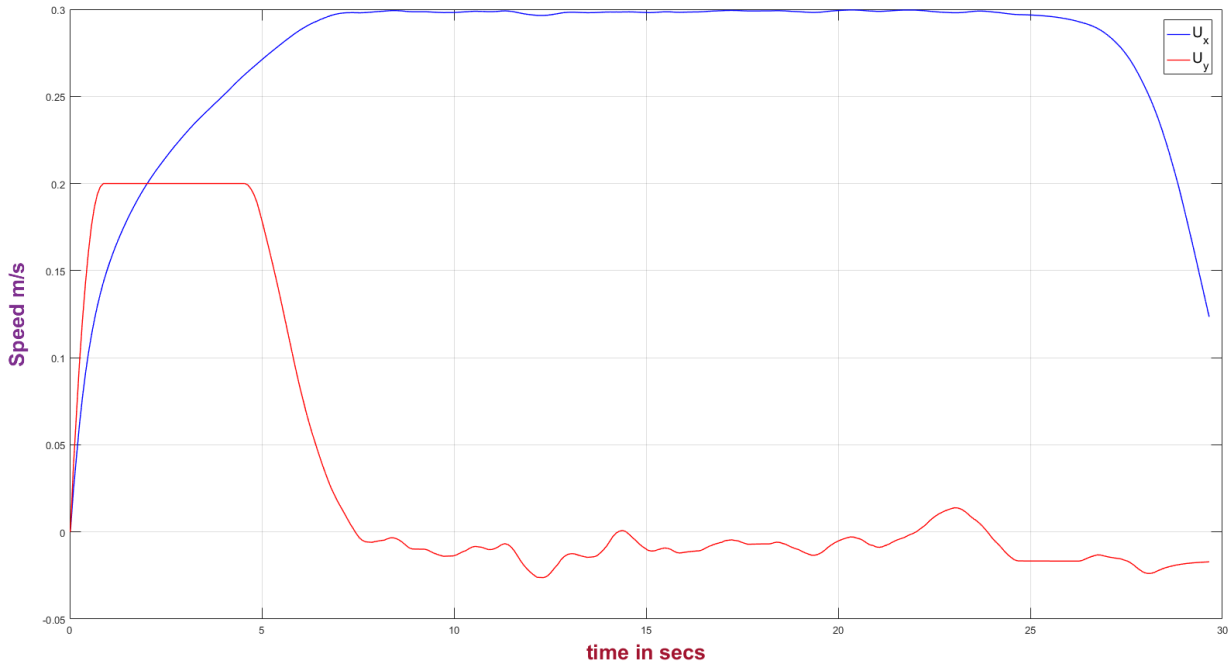


Figure 4.63 : Robot Lateral and Linear Speeds

Similarly, in the above plot the lateral speed increases in the first 5 seconds when the robot is aligning itself to the rack and after 5 seconds the linear speed increase to force the robot to travel the required linear distance.

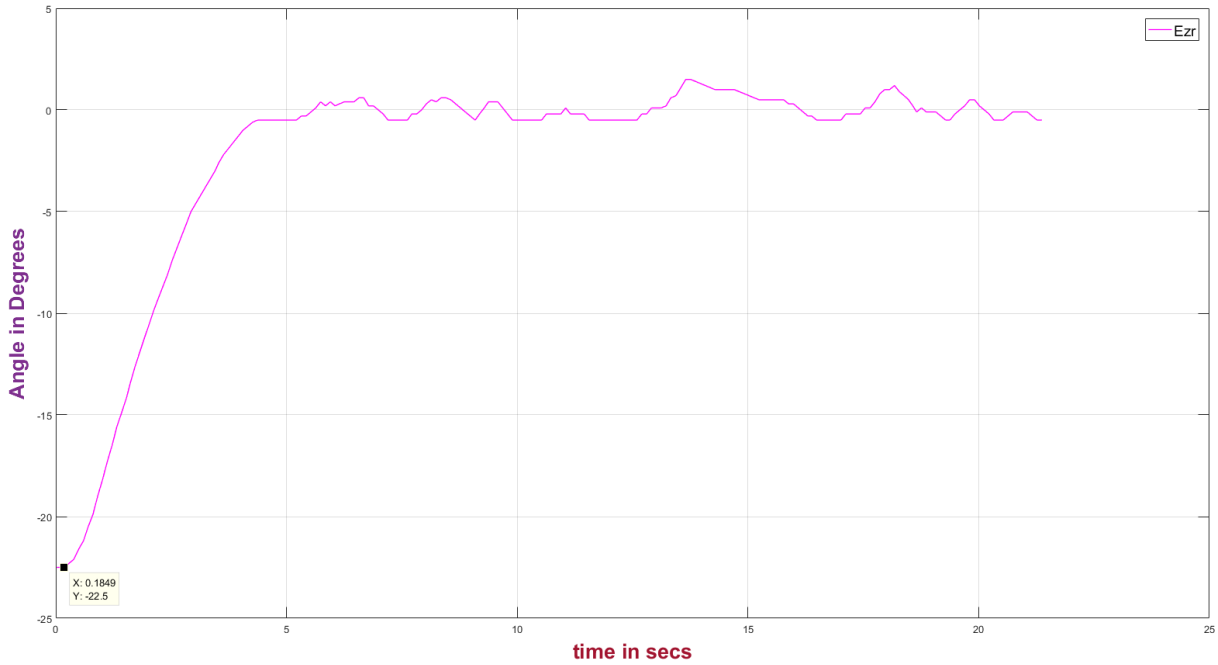


Figure 4.64 : Robot Initial Orientation Error

The above plot indicates the initial orientation error that the robot has with respect to the rack. This error is the orientation difference between the current orientation of the robot and the parallel side of the rack given by the Hough line. This error converges to zero in the first 5 seconds indicating the robot is able to acquire the orientation from its initial placement to a configuration parallel to the rack. This is also evident from the orientation angle given in the robot frame by odometry in figure (4.65) below. The orientation angle of robot in its own reference is zero which starts to increase when the robot starts to turn counter-clockwise. The final angle before 5 seconds is almost the same as the orientation error reported in figure (4.64) at the very start.

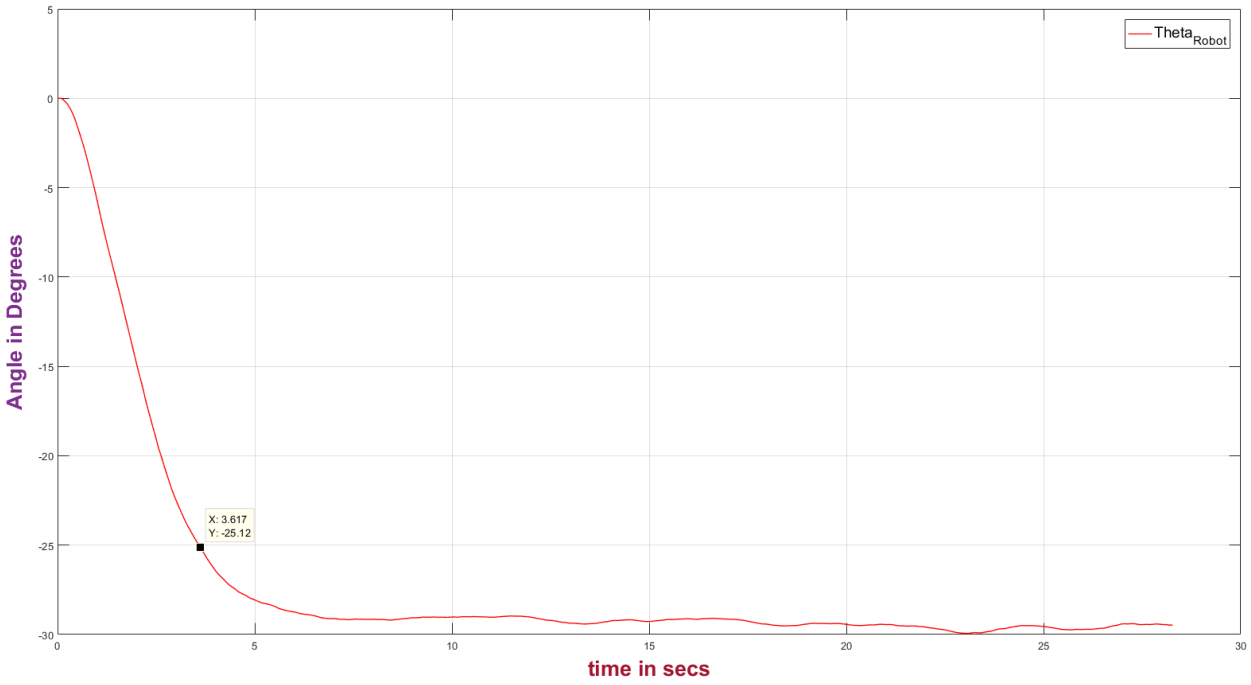


Figure 4.65 : Robot Orientation Angle from Odometer

The plot in figure (4.66) below is similar to figure (4.56) in which the lateral and linear errors decrease with time during the alignment and linear motion of the robot along the rack. Further the plot in figure (4.67) below again gives the evolution of the midpoints of the two Hough lines on rack sides for the full motion of the robot to achieve its target. The width between these two lines formed by the evolution of midpoints is within 3 meters (on y-axis) corresponding to the actual width between the racks.

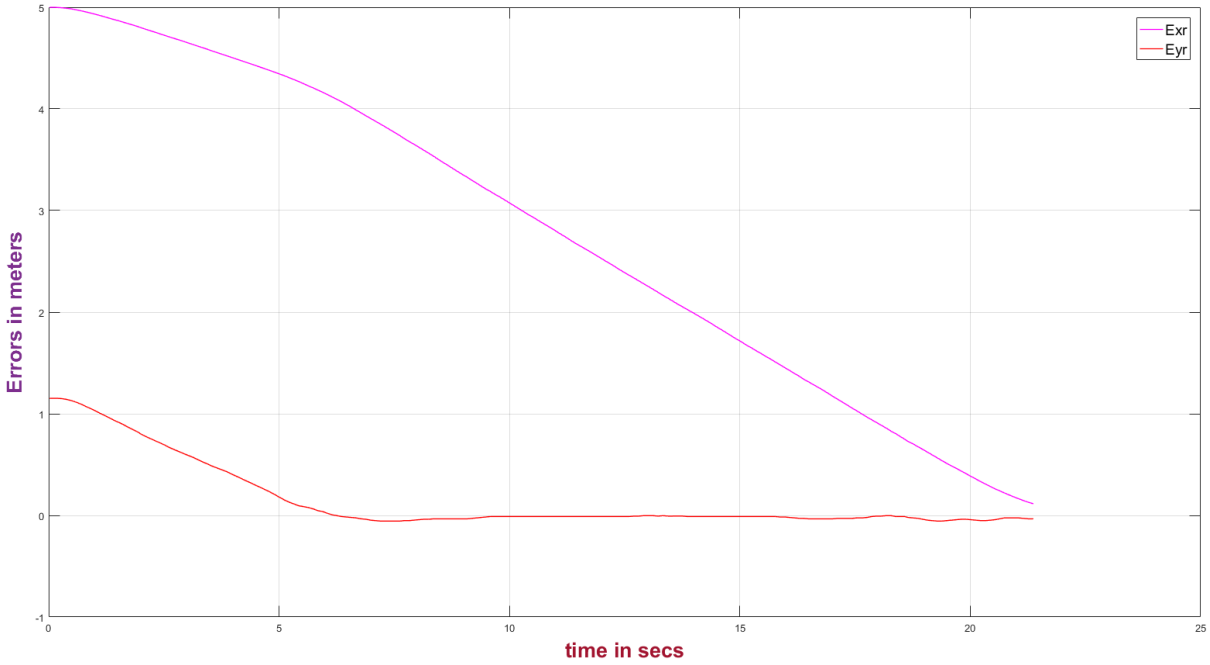


Figure 4.66 : Robot Lateral Alignment and Linear Distance Errors

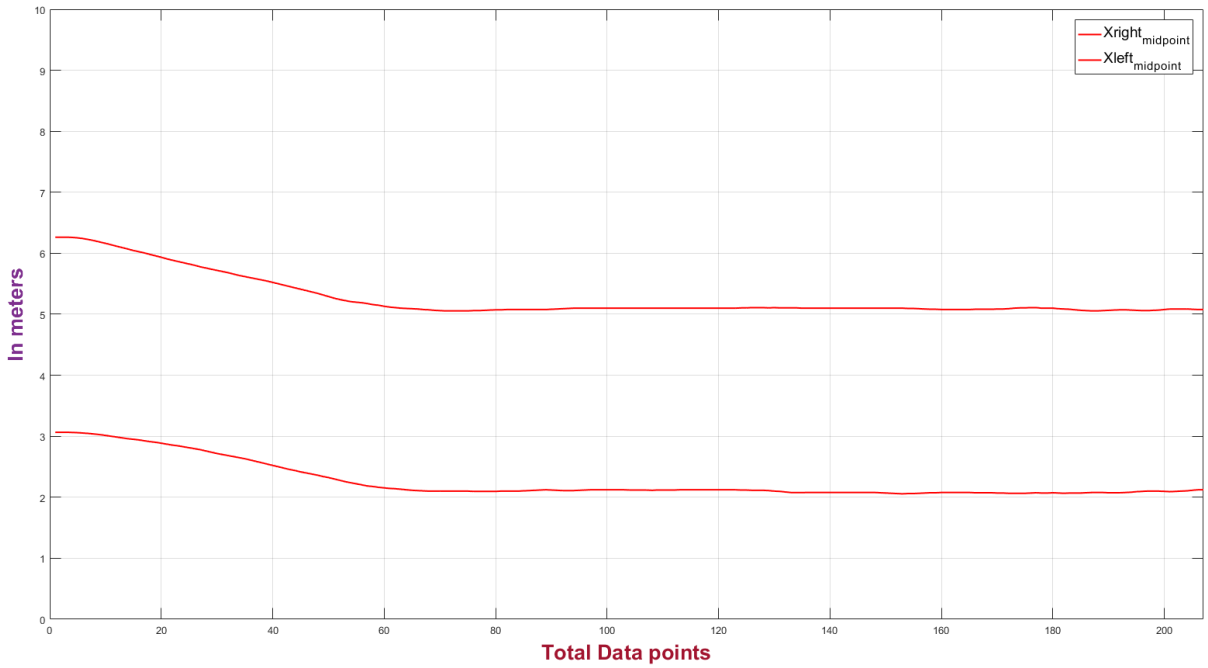


Figure 4.67 : Evolution of Hough Lines Tracking Points (X_{m_1}, X_{m_2})

4.4 Motion planning manipulator

For the task of picking the grasping is acquired using the articulated arm of the mobile manipulator. Object retrieval consists of moving the gripper to the package location (pick point), grasping the package and then moving it to another specified pose. Considering the motion of the arm to execute such actions, it is primitive to respect the constraints of collision avoidance, workspace limitation, shortest path, speed and smooth motion. This means that while remaining in the workspace limit, and executing a smooth motion to the target point with the desired speed, the arm must not collide with itself, the mobile base or other packages on the pallet. This requires the development of a sophisticated motion planner which can satisfy all the constraints and desired optimal criterion, and can produce such trajectories which guarantee the safe motion of the arm from a current to target point. The development of such a motion planning framework was not the scope of this research due to the constraints of time and objective of the enterprise. Hence a simple motion strategy was adopted and a simplified motion planner was developed to execute Cartesian trajectories from a current to the target point while respecting the boundaries of the Cartesian workspace.

4.4.1 Workspace

For the motion planning of the manipulator, it is imperative to have an idea of the Cartesian workspace in which the arm is required to operate. The Cartesian workspace of a generic manipulator can be generated by the forward kinematics using the DH parameters. The generated workspace envelope i.e. circular, sphere etc. depends on the arrangement and configuration of the joints. If the configuration of joints is influenced by constraints, then the limitation is also reflected in the resulting Cartesian workspace, i.e. the corresponding workspace will be limited.

In the current scenario of the research project, the hardware configuration of the robotic platform and the placement of the 'UR5' on the mobile base imposes specific constraints on the movement of the robot in its Cartesian workspace. This means the end effector cannot, or should not be allowed to reach those positions which violate these constraints. After experimental analysis of the given platform, three significant constraints were identified. The first being the maximum reach of the arm, the second is the collision with the structure housing the arm on the mobile base, and the third is to keep the orientation of the package/box consistent during motion after the box is already picked. This is desired to maintain 'the stability of the box with vacuum gripping' while being moved around by the arm.

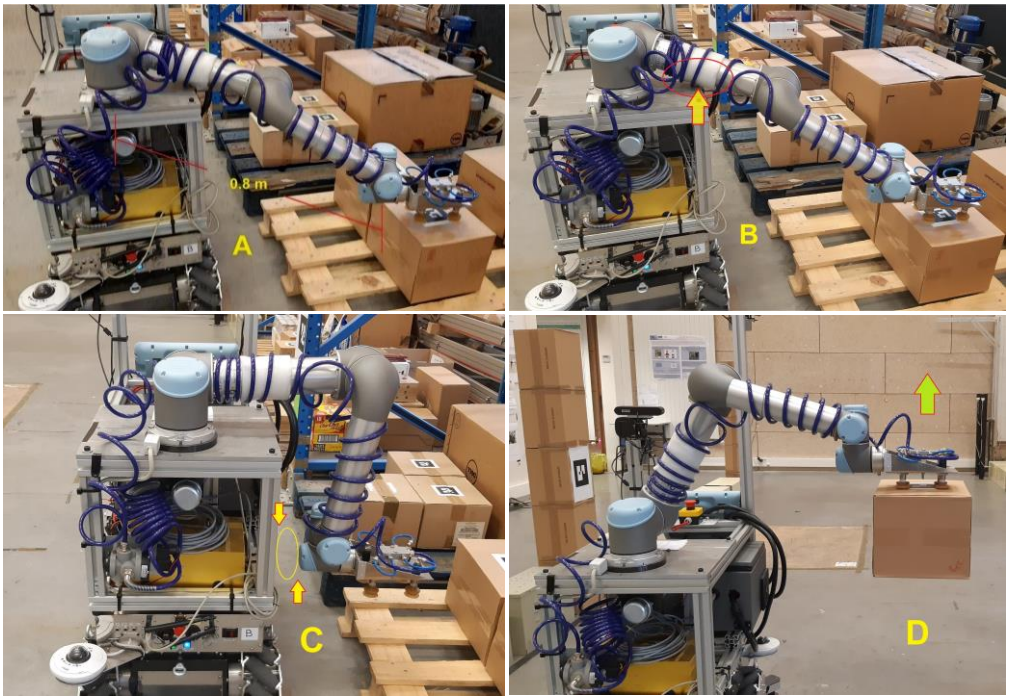


Figure 4.68 : Constraints imposed by the platform hardware and placement of UR5

The figure above highlights a description of the constraints imposed on the movement of the arm. In figure 'A' the arm is fully extended showing the maximum reach. This is the maximum limit of the workspace and also the arm is close to a singular configuration. It is difficult to pick up the box straight upwards in this position. This configuration can be avoided by not extending the arm and bringing the mobile platform close enough for the arm to conveniently reach the pick point. A motion planner therefore must take into account this constraint by planning a motion respecting the joint limits adhering to this constraint. The presence of this constraint also leads to condition 'B', where the arm is also in collision with its base housing on top of the mobile manipulator. This collision point is indicated by the arrow in the figure 'B', where the arm link is in collision with the base plate. The joint limits for this configuration are given in table (10) below.

The collision constraint is again highlighted by figure 'C' which shows the arm link very close to the side of the assembly housing indicated by two arrows. This is the configuration in which a part (link) of the arm gets very close to the platform assembly, and there is a danger of collision with the platform housing during motion. The motion planner therefore must produce a trajectory which ensures that any part of the arm does not come into close proximity of the assembly housing during motion. Figure 'D' shows the third and the last constraint pertaining to the configuration of the box/package. In this configuration the orientation of the box has to be kept consistent. This is due to the fact that once the arm picks up the box from the top with the vacuum gripper, the stability of the box has to be ensured to

retain the vacuum suction force. If the box is not stable during motion or if the gripper turns, then a component of force will be induced at the box-suction cup interface due to inertia of weight of the box, which will break the contact between the box and the suction cup. The suction force will not be retained anymore and the box will break away from the grip due to inertia.

Joints / Constraint	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
A	$-175° \leq \theta \leq 60°$	$-90° \leq \theta \leq 0°$	$60° \leq \theta \leq 135°$	$-100° \leq \theta \leq 0°$	$\theta \leq 90°$	$\theta = 0$
B	$-175° \leq \theta \leq 60°$	$-90° \leq \theta \leq 0°$	$60° \leq \theta \leq 135°$	$-100° \leq \theta \leq 0°$	$\theta \leq 90°$	$\theta = 0$
C	$-175° \leq \theta \leq 60°$	$-90° \leq \theta \leq 0°$	$60° \leq \theta \leq 100°$	$-100° \leq \theta \leq 0°$	$\theta \leq 90°$	$\theta = 0$
D	$-175° \leq \theta \leq 60°$	$-90° \leq \theta \leq 0°$	$60° \leq \theta \leq 135°$	$-100° \leq \theta \leq 0°$	$\theta \leq 90°$	$\theta = 0$

Table 10 : Joint ranges to respect all constraints

With reference to the default configuration of the UR5 shown in figure (3.6) and the DH parameters mentioned in Table-1, the table above gives the ranges of joint limits that satisfy the three main constraints, i.e. the configurations that must be avoided. These joint limits were identified manually and a motion planned within these limits will ensure the arm neither collides, nor is fully extended within the workspace. The limits in red for each joint angle of the arm, indicate the primary joints which allow the arm to be in a constrained configuration. For example, in the case of 'C', if 'θ_2' is above '0' the arm shoulder is in collision with the base plate, likewise if 'θ_3' (elbow joint) is below '60' the arm part i.e. wrist link will collide with the side of the housing assembly. Similarly in the case 'D' if the wrist_1 angle 'θ_4' is greater than '0', then the gripper will collide with arm elbow link, and if wrist_3 angle '$\theta_6 \neq 0$' then the box will break away from the gripper due to inertia.

The Cartesian workspace pertaining to these joint limits in figure (4.69) was generated by plugging in the DH parameters using equation (3.9) and then using the forward kinematic equations given below. Where 'Px', 'Py' and 'Pz' give the position coordinates of the end effector in 3D space.

$$P_x = d_6(c\theta_1 s\theta_4 s\theta_5 s(\theta_2+\theta_3) - c\theta_1 c\theta_4 s\theta_5 c(\theta_2+\theta_3) + s\theta_1 c\theta_5) + d_5(c\theta_1 s\theta_4 c(\theta_2+\theta_3) + c\theta_1 c\theta_4 s(\theta_2+\theta_3)) + a_3 c\theta_1 c(\theta_2+\theta_3) + d_4 s\theta_1 + a_2 c\theta_1 c\theta_2 \quad (4.36)$$

$$P_y = d_6(s\theta_1 s\theta_4 s\theta_5 s(\theta_2+\theta_3) - s\theta_1 c\theta_4 s\theta_5 c(\theta_2+\theta_3) - c\theta_1 c\theta_5) + d_5(s\theta_1 s\theta_4 c(\theta_2+\theta_3) + s\theta_1 c\theta_4 s(\theta_2+\theta_3)) + a_3 s\theta_1 c(\theta_2+\theta_3) - d_4 c\theta_1 + a_2 s\theta_1 c\theta_2 \quad (4.37)$$

$$P_z = -d_6 s\theta_5(c\theta_4 s(\theta_2+\theta_3) + s\theta_4 c(\theta_2+\theta_3)) + d_5(s\theta_4 s(\theta_2+\theta_3) - c\theta_4 . c(\theta_2+\theta_3)) + a_3 s(\theta_2+\theta_3) + a_2 s\theta_2 + d_1 \quad (4.38)$$

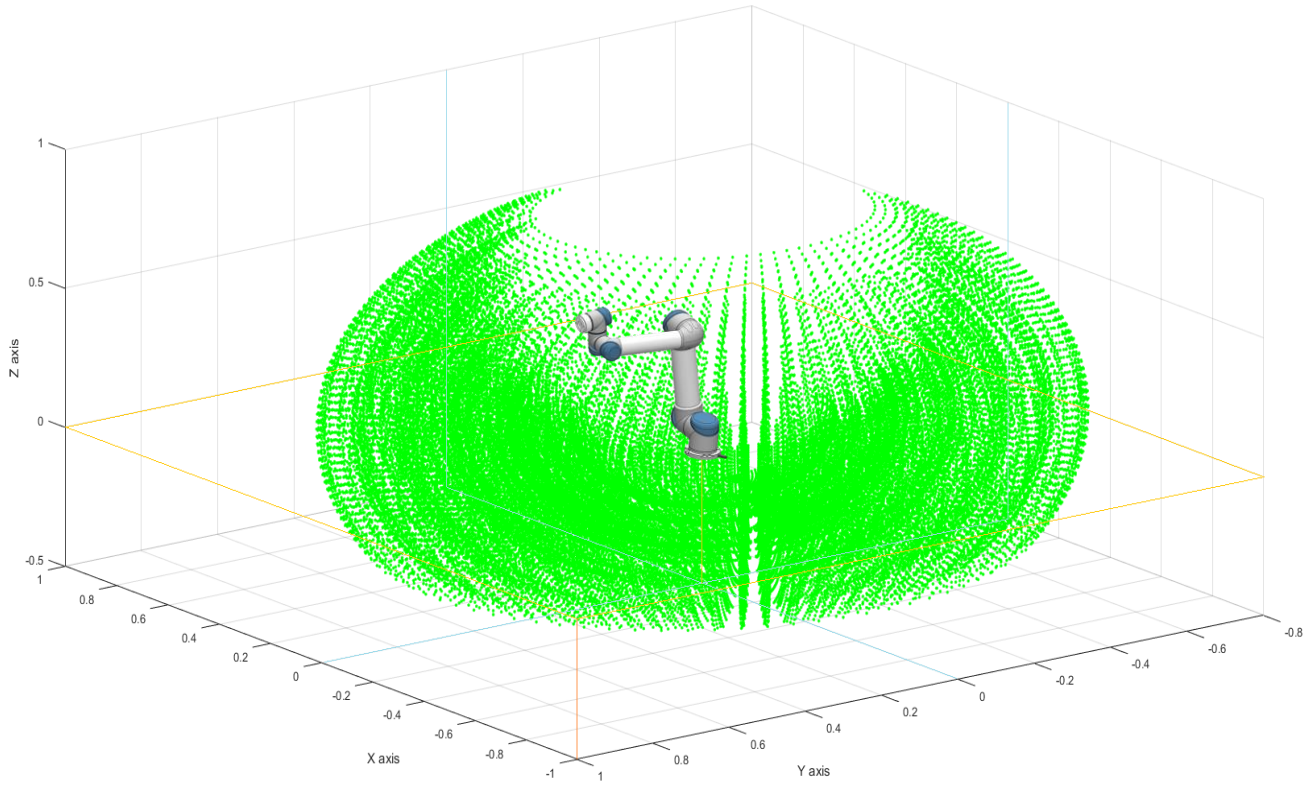


Figure 4.69 : The feasible Cartesian workspace of UR5 under the joint limits

The above figure shows the feasible workspace of the UR5 with respect to its placement on the mobile manipulator. The arm base is placed at a height of 80cm from the ground on the housing plate as in figure (4.68). The above figure indicates that the arm can reach both above and below its placement where the lower height limit is '-0.5m'. This is due to the limit of the shoulder joint ($90° \leq \theta_2 \leq 0°$) that is if '$\theta_2 > 0°$' then the arm will collide with the base plate.

4.4.2 Motion Planner

Considering the placement of the UR5 arm with respect to the physically modelled hardware, it is found to be convenient for the robot to align its side to a pallet and pick up the package/box from this side. Once aligned to the pallet, the target pick location is detected by the camera and once the pose is acquired, the arm will move its end effector to this pick pose. This motion will consist of moving the end effector from a current configuration to the target configuration. The current configuration is the start point of the motion and the target configuration will be the end point. The end effector will thus follow a trajectory in the Cartesian space from a start point to the end point. The objective of the motion planner is to produce such a trajectory in the Cartesian space, which governs the motion of the arm taking into account the start point, the end point and the joint limits. The trajectory generation can be done both in the joint space and the Cartesian space. In the current scenario the motion planning for UR5 was done in Cartesian space. This consisted of generating a waypoint trajectory from the start point to the goal and then finding the 'inverse kinematic' solution for each point of the trajectory.

4.4.2.1 Kinematic chain

The kinematic chain of UR5 consists of 6 links with joints. The vacuum gripper is attached rigidly to the gripper link. A frame is attached to each link according to the DH convention to describe its relative orientation in the kinematic chain. The objective is to align the gripper to the pick point to grasp the object. This is achieved by aligning the gripper frame with the acquired object pose frame. The object pose frame is acquired visually by a camera, and in the present case it is acquired using the ROS based ar_track_alvar framework. The pose of the box/package is given by a marker attached to it. Aligning the gripper frame requires it to move from a current configuration to the target configuration i.e. marker frame. The gripper of a particular robot can acquire a specific pose in its task space ($\mathcal{T} \subset R^2$ or $\mathcal{T} \subset R^3$) depending on the configuration of joints, either by forward kinematics or inverse kinematics. However, since in forward kinematics the mapping is from the joints to the end effector, the resulting pose of the gripper is acquired by specifying a particular set of joint values. To acquire a specific 'desired' pose in the task space, it is difficult to predict which set of particular joint configuration will enable to do so. Hence to achieve this, the reverse topology is applied, i.e. to use the inverse kinematics (IK) by first specifying a target pose of the gripper, and then get the corresponding solution of joints to execute. The resulting values of the joints acquired by the 'IK' solution will bring the end effector to the desired pose $(x, y, z, \gamma, \alpha, \psi)$ in the workspace.

The pose of the gripper (end effector) is generally given with respect to the base of the arm, and this is acquired by the homogenous transformations from the base to the end effector. The transformations for a particular 6D robot, e.g. UR5 is given as

$$ {}^{0}_{6}T = {}^{0}_{1}T(\theta_1).{}^{1}_{2}T(\theta_2).{}^{2}_{3}T(\theta_3).{}^{3}_{4}T(\theta_4).{}^{4}_{5}T(\theta_5).{}^{5}_{6}T(\theta_6) \tag{4.39} $$

In the above equation '${}^{5}_{6}T$' is the transformation for the end effector link, and the generic transformations from the base to the gripper frame are given visually as

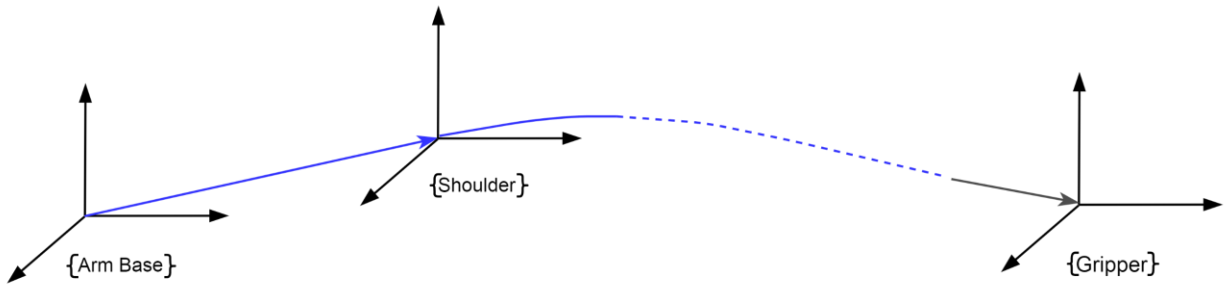


Figure 4.70 : Graphical representation of frame transformations from base to gripper

The objective is to align the gripper with the object frame which is given by a marker. To achieve this, it is required to have an idea of the displacement required to align the gripper frame with the marker frame. This displacement can be computed by computing the transformation of marker frame relative to gripper frame. Computing this transformation also requires to know the location of the marker frame with respect to a global reference frame in advance. In our case we use the robot center frame as the global reference to locate the marker.

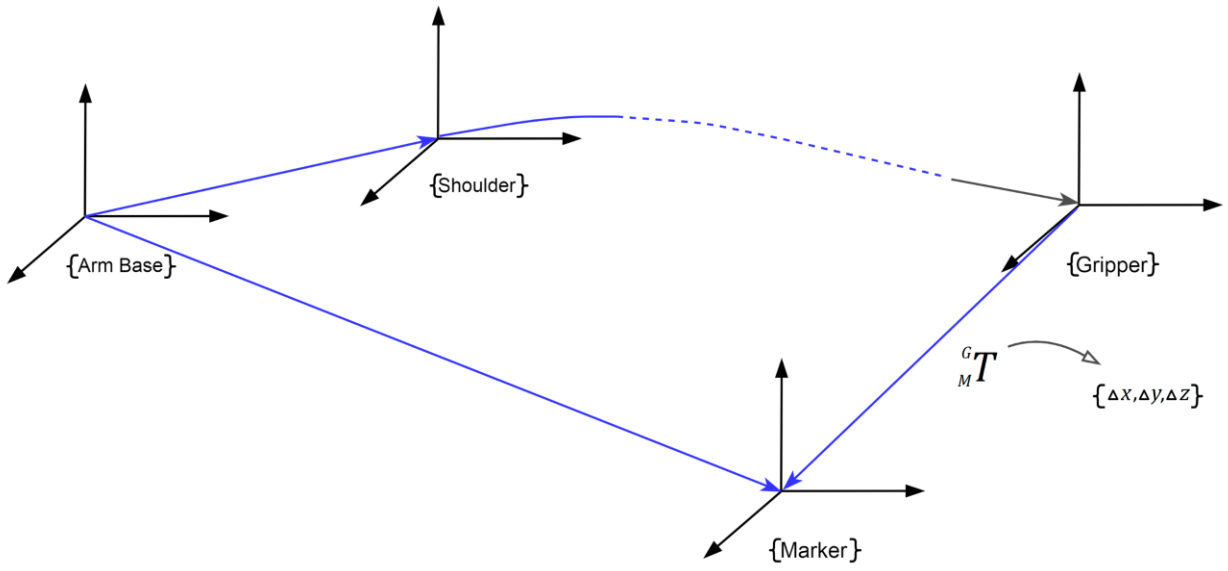


Figure 4.71 : Transformations from base to gripper and base to marker

Locating the marker frame means computing the transformation of the marker frame with respect to the specified reference frame. Once the transformation from the base (reference) frame to the marker frame is known, the pose of the marker with respect to the gripper can be computed. Referring to figure (4.71) the transformation from the arm base to the gripper frame and then marker frame is given as

$$ {}^{B}_{M}T = {}^{B}_{S}T . {}^{S}_{E}T \ \ldots\ldots \ {}^{W}_{G}T . {}^{G}_{M}T \quad \Rightarrow \quad {}^{B}_{M}T = {}^{B}_{G}T . {}^{G}_{M}T \tag{4.40} $$

The transformation from the arm base to the marker is given by ROS via ar_track_alvar on detection of the marker with the camera. Using this, the transformation from the marker to the gripper can be computed as

$$ {}^{G}_{M}T = {}^{G}_{B}T . {}^{B}_{M}T \quad \Rightarrow \quad {}^{G}_{M}T = {}^{B}_{G}T^{-1} . {}^{B}_{M}T \tag{4.41} $$

The transformation '$ {}^{G}_{M}T $' gives the displacement required to align the gripper with the marker frame. In the current scenario of the mobile manipulator, since the arm is fixed on the mobile base, the global reference frame as the center of the mobile base is referred to as the 'base_link'. An idea of the position of the package from the center of the mobile base allows for mobile robot position adjustment if required. Once the pose of the marker is acquired relative to 'base_link' by the 'ar_track_alvar', the pose of the gripper with respect to the marker is computed by the transformations given in (4.42) and

$$ {}^{G}_{M}T = {}^{AB}_{G}T^{-1} . {}^{B}_{AB}T^{-1} . {}^{B}_{M}T \tag{4.42} $$

where '$ {}^{B}_{M}T $' gives the pose of the marker with respect to the 'base_link' of the mobile robot.

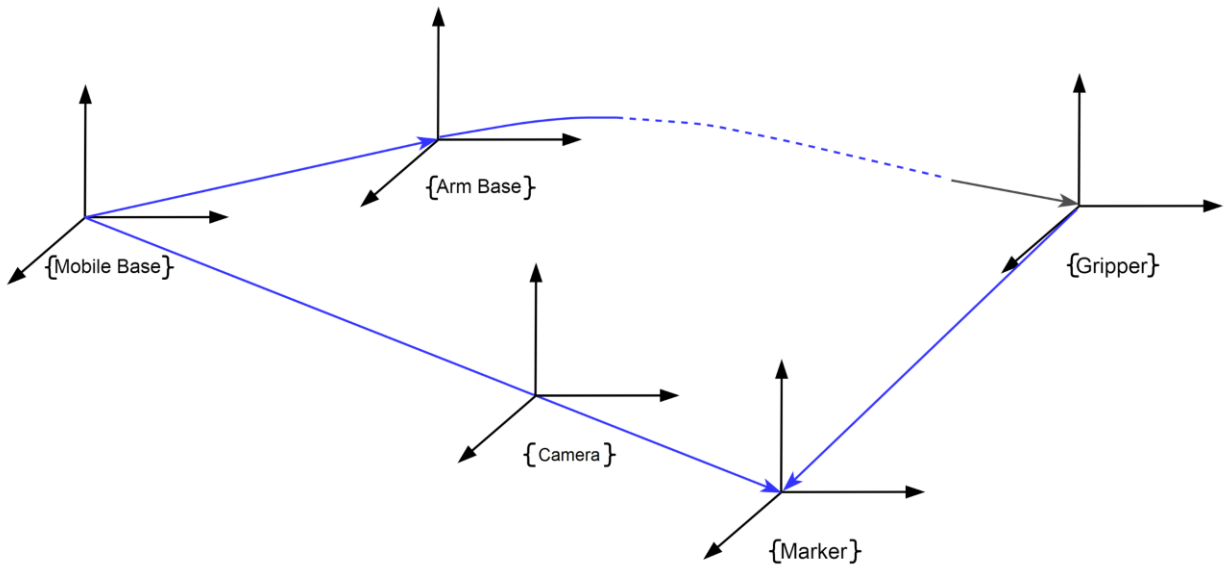


Figure 4.72 : Transformations from mobile base to gripper and to marker

4.4.2.2 Trajectory generation

Considering the figure below for a generic move in 3D space, for a basic point to point motion, i.e. from A to B, the gripper will cover more distance along 'Z' and 'X' axis respectively as compared to 'Y'. To execute this motion a Cartesian trajectory must ensure the movement of the gripper in all the axes at the same time, meaning the gripper covers its respective distances on all the axis simultaneously. The development of such a trajectory is as follows.

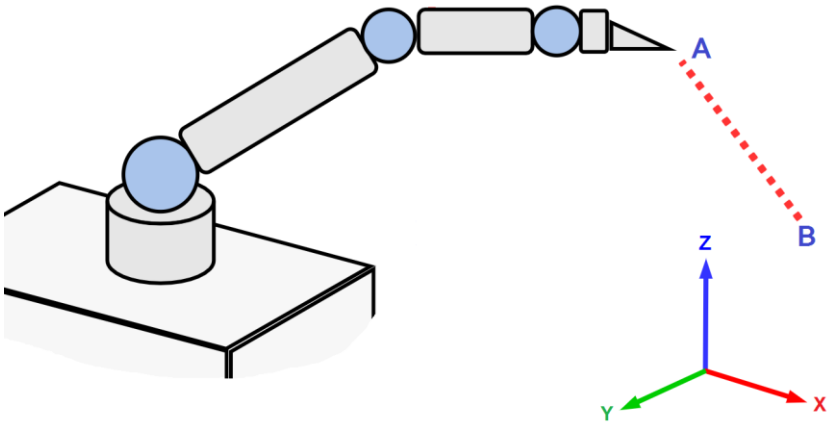


Figure 4.73 : A generic move in 3D space from point A to B

Referring to figure (4.72) the transformation from the gripper to the marker gives the alignment difference between the gripper and the pose of the package. The difference is given by the $(\Delta x, \Delta y, \Delta z, \Delta\gamma, \Delta\alpha, \Delta\psi)$ in the Cartesian space where

$$\Delta x = x_{tgt} - x_{crnt} \quad , \quad \Delta y = y_{tgt} - y_{crnt} \quad , \quad \Delta z = z_{tgt} - z_{crnt} \tag{4.43}$$

Using the $'\Delta'$ and a resolution specified manually, the waypoints are computed for a Cartesian trajectory by

$$wpts_{i=1\ldots n} = \frac{\Delta x}{res} \ , \ \frac{\Delta y}{res} \ , \ \frac{\Delta z}{res} \tag{4.44}$$

The waypoints in the above equation are computed based on the maximum value of a $'\Delta'$, either in the 'x', 'y' or 'z'. The maximum alignment difference either in x, y or z is used to compute the waypoints since the objective is to produce a Cartesian trajectory with variable step size for each waypoint of the Cartesian move corresponding to x, y or z. This variable step size is due to the fact that, the respective distance to move on either of the three axis at the

same time are not the same. Then the waypoints for moving in 'x', 'y' or 'z' are produced by incrementing the current position with a step. The step or the increment is computed by using the '$\Delta's$' from equation (4.43) and '$wpts$' from equation (4.44) by

$$x_{incrmt} = \frac{\Delta x}{wpts},\ y_{incrmt} = \frac{\Delta y}{wpts},\ z_{incrmt} = \frac{\Delta z}{wpts} \tag{4.45}$$

The increment is the step size to produce the next waypoint for each 'x', 'y' and 'z' move separately. The objective of selecting the maximum $'\Delta'$ is to produce maximum number of waypoints, which will allow to compute the step with variable step size in equation (4.45) since the waypoints is the same for all increments. After computation of the increments, the waypoints of the Cartesian trajectory are produced by incrementing the current 'x', 'y' and 'z' of the gripper or the pose i.e.

$$wpts_{x_i} = crnt\_x + x_{incrmt}\ ,\ wpts_{y_i} = crnt\_y + y_{incrmt}\ ,\ wpts_{z_i} = crnt\_z + z_{incrmt} \tag{4.46}$$

The waypoints Cartesian trajectory is stored in a vector and then fed to the inverse kinematic function to compute the inverse kinematic (IK) solutions corresponding to each waypoint. This is done in a loop and the resulting IK or joint trajectory is also stored in a vector. Once the joint trajectory is available, it is fed to the position controller of UR5 to execute this trajectory or the IK solutions to produce the corresponding Cartesian motion (trajectory) in 'x', 'y', 'z'. For the computation of the IK solutions, the opensource Kinematic Dynamic Library[25] (KDL) framework has been used. A custom IK solver for computation of joint trajectory for the Cartesian motion of UR5 was not developed in this thesis due to constraints of time. The computation of the Cartesian trajectory and its corresponding IK solutions are summarized below.

25 https://orocos.org/

Algorithm 4 : Compute trajectory

Input: $Package\ Pose \rightarrow (x, y, z)$ – A 3D pose of package given by the marker

Output: trajectory → $(q_1 \cdots q_n)$ – A joint trajectory corresponding to Cartesian waypoints

Begin

//get current pose of the gripper with respect to arm_base

1. Compute → ${}^{B}_{G}T$ → $(x_G, y_G, z_G, \gamma_G, \alpha_G, \psi_G)$ //(Transformation from Arm base to gripper)

//Detect Marker and extract coordinates → //(Transformation of pose in gripper frame)

2. Compute → ${}^{G}_{M}T = {}^{AB}_{G}T^{-1} \cdot {}^{B}_{AB}T^{-1} \cdot {}^{B}_{M}T$ → $(x_M, y_M, z_M, \gamma_M, \alpha_M, \psi_M)$

3. Compute current theta → $\theta_{crnt} = y_G / x_G$ //(The yaw (ψ_G) in the global frame)

4. Compute the reach of the pose → reach= $\sqrt{x_t^2 + y_t^2 + z_t^2}$ //(where x_t, y_t, z_t are for target pose)

5. Compute radius of circle → radius= $\sqrt{x_t^2 + y_t^2}$ // (if to move in circle)

6. Check reach →
```
if (reach < threshold)
        return
else
        continue to next step
end
```

7. Compute $'\Delta s'$ → from the gripper current pose and the marker target pose

$\Delta x = (x_t - x_{crnt}) \times 1000\ (mm)$ $\Delta y = (y_t - y_{crnt}) \times 1000\ (mm)$

$\Delta z = (z_t - z_{crnt}) \times 1000\ (mm)$ $\Delta\psi = (\psi_t - \psi_{crnt}) \times 180/\pi$

```
if ( Δψ > 60° )
    Move circular=true;
end
```

```
if ( Δx > Δy && Δx > Δz )                    //check Δx
  if (Move circular)
        wpts = Δψ / res * π/180
  else
        wpts = Δx / res                      //Compute waypoint increments based on Δx
  x_incrmt = (Δx/wpts) / 1000
  y_incrmt = (Δy/wpts) / 1000
  z_incrmt = (Δz/wpts) / 1000        ψ_incrmt = Δψ/wpts
end
```

if ($\Delta y > \Delta x$ && $\Delta y > \Delta z$) //check Δy

if (Move circular)

wpts = $\Delta\psi / res * \frac{\pi}{180}$

else

wpts = $\Delta y / res$ //Compute waypoint increments based on Δy

$x_{incrmt} = (\Delta x / wpts) / 1000$

$y_{incrmt} = (\Delta y / wpts) / 1000$

$z_{incrmt} = (\Delta z / wpts) / 1000$ $\psi_{incrmt} = \Delta\psi / wpts$

end

if ($\Delta z > \Delta x$ && $\Delta z > \Delta y$)

if (Move circular)

.....

wpts = $\Delta z / res$ //Compute waypoint increments based on Δz

.....

.....

end

8. Compute the Cartesian trajectory

for (moves=0; moves<wpts; moves++)

if (Move circular)

$crnt\_x = r.\cos(\theta_{crnt})$, $crnt\_y = r.\sin(\theta_{crnt})$ //for a circular move

else

$crnt\_x = crnt\_x + x_{incrmt}$, $crnt\_y = crnt\_y + y_{incrmt}$ //for a linear move

end

$crnt\_z = crnt\_z + z_{incrmt}$

$\psi_{crnt} = \psi_{crnt} + \psi_{incrmt}$ //Taking into account the orientation of the gripper

$\theta_{crnt} = \theta_{crnt} + \psi_{incrmt}$

// save cartesian poses in a vector

cartesian_pose[0]= $crnt\_x$, cartesian_pose[1]= $crnt\_y$, cartesian_pose[2]= $crnt\_z$,

cartesian_pose[3]= γ_G , cartesian_pose[4]= α_G , cartesian_pose[5]= ψ_G

end

9. Compute the Inverse kinematic solutions

```
for (step=0; step<wpts; step++)
//compute IK for the current step of cartesian point
        cart_pose [step]= crnt_x , crnt_y , crnt_z , γ_G , α_G , ψ_G
        Solve → IK( Q_crnt1..6 , cart_pose, Q_out1..6 )        // 'Q' is joints 1 to 6 for UR5
//save the joint solutions for the current step
        Q_out[step] = q1 , q2 , q3 , q4 , q4 , q6
end
```

10. Check the joint solutions

```
for (step=0; step<wpts; step++)
  for (joint=0; joint<6; joint++)
    if (traj_pnt[step].positions[joint]>3.1)        //check the joint solutions for the current step
        point_not_valid=true;
        break;
    end
  end
end
```

11. Publish the joint trajectory

```
if (point_not_valid)
        Message  →  "trajectory not executable"
        return
else
        Message  →  "trajectory executable"
        publish(traj_pnt)                    //publish trajectory on rostopic
end
```

end

In the above algorithm, first the gripper's current pose (x,y,z) is computed by the transformation from the arm_base to the gripper. The marker is detected in the mobile robot base frame. To compute the relative difference between the gripper position and the marker, the pose is transformed into the gripper frame via the respective transformation from mobile base to gripper frame in step 2. Depending on the location of the package (box) from the mobile base, the target pose (marker) can be close or far off to reach. If it is far off then the maximum

reach of the pose and the radius of the circle is computed in case the gripper has to execute a circular trajectory to reach the pose. If the reach is greater than a threshold, then the algorithm stops, otherwise it continues to develop the Cartesian trajectory.

The alignment difference between the gripper and the target pose is computed in step '7' by computing the respective $'\Delta s'$. The decision to move circular is determined by the difference between the current angle $'\theta_{crnt}'$(step 3) and the target angle $'\theta_{tgt}'$ or the yaw $'\psi_{tgt}'$. If the difference is more than 60°, then the arm is required to execute a circular trajectory to reach the pose. Then the waypoints or trajectory points are computed by comparing the maximum $'\Delta'$. If circular move is true then the maximum $'\Delta'$ of yaw is used to compute the trajectory points, otherwise Δx, Δy, or Δz is used. The computed waypoints are then used to compute the step size or the 'increment' for each 'x','y','z' and 'yaw' component. For the development of the Cartesian trajectory in step 8, if it is required to move circular, then the Cartesian trajectory points are computed using the equation of the circle, otherwise the Cartesian points are computed using the step size or increment computed in the previous step. The height 'z' is incremented or decremented independently of the circular move or the linear move, therefore the computation of the Cartesian 'z' is done separately without a check. Similarly, the orientation (yaw) of the gripper is also taken into account and computed based on the increments (step 7).

Once all the components have been computed, they are stored in a vector to be used for computation of the inverse kinematic (IK) solutions in a loop. This is done in step '9'. The output of the 'IK' function is the '6' joint values corresponding to the Cartesian poses. The feasibility of the 'joint' trajectory or the 'IK' solutions is verified in step '9', since the 'IK' function produces 'IK' solutions based on the minimum and maximum joint limits that is [0, 2π], while referring to table 10, it is required to have 'IK' solutions corresponding to the limits prescribed in the table, otherwise a collision will result. Therefore, the check in step '10' ensured that the joint trajectory does not lead to any configuration leading to a collision. Once all the solutions in the joint trajectory are found to be valid under the limits the trajectory is then executed on a rostopic in step '11' to produce the corresponding Cartesian trajectory. The trajectory is executed using the position controller of UR5.

4.4.2.3 Simulations

The motion of the arm is controlled by the manipulation state machine explained in section (5.3.3.1.2). This state machine executes the motion in sequential steps for a pick and drop move and utilizes the algorithm explained above to generate a Cartesian motion corresponding to each step. This state machine executes the arm motion in the form of reach to goal, descend and ascend moves and the trajectory planner produces a Cartesian trajectory for each move. This motion generation is explained below with the simulations of a real-time pick and drop task of a package. The data was recorded in real time during the motion of the UR5 and was plotted using MATLAB. It has to be highlighted that the data was recorded for a continuous motion for a complete pick and drop move and the trajectories correspond to the motion of the gripper frame in 6D.

4.4.2.3.1 Pick Task

The plot below shows the simulation of a pick task in which the arm was moved to pick up a box. The 'Start pose' in the figure shows the home pose of the arm (gripper). The arm rests in this pose prior to any move. The 'Target pick pose' gives the location of the marker on the package.

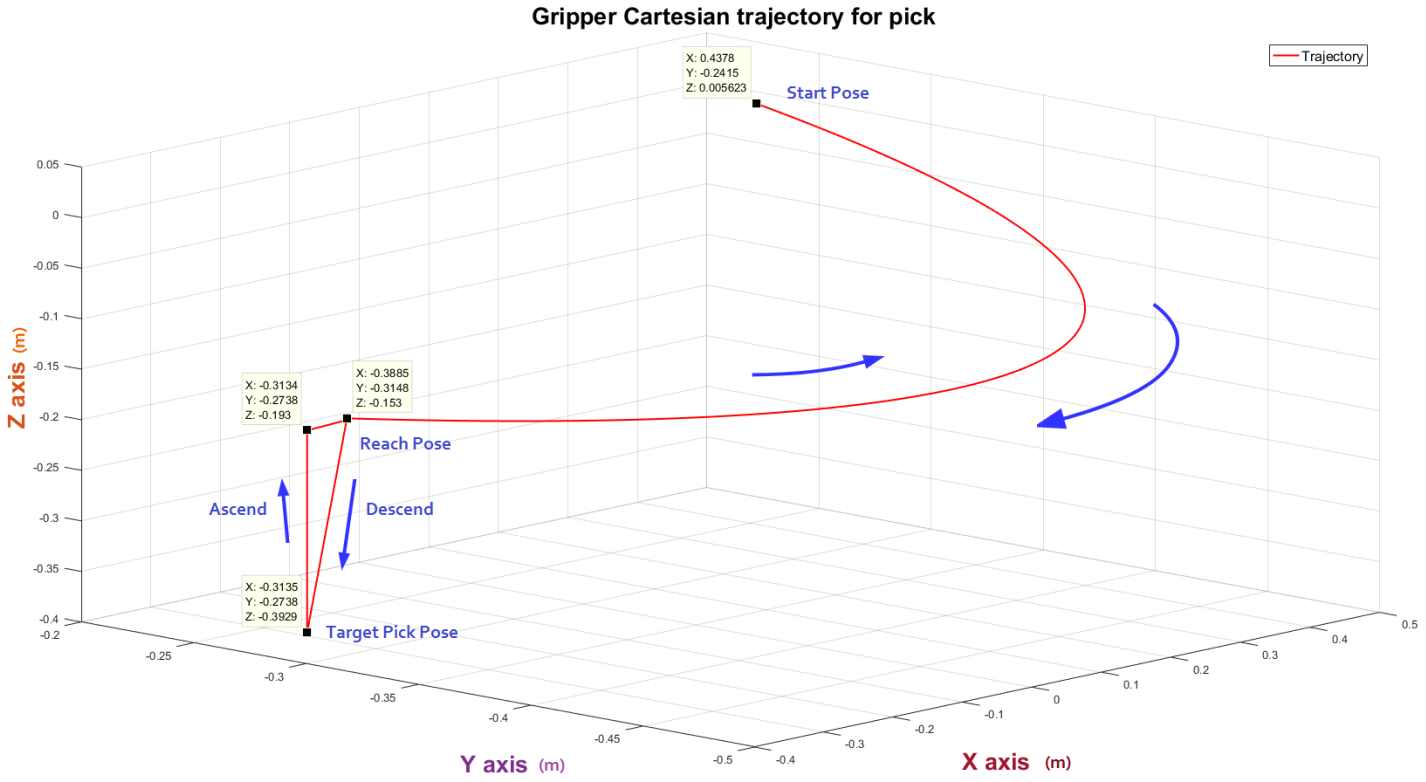


Figure 4.74 : A full Cartesian trajectory for a Pick task

Once the pose of the box (package) is extracted, the motion of the arm to grasp is executed in sequential steps producing a Cartesian trajectory for each step. In the figure above the gripper first executes a circular trajectory and reaches the position above the package, it then executes the 'descend' motion thus executing the linear Cartesian trajectory. Once the box is grasped with the gripper (vacuum) the arm then executes the linear motion of 'ascend' to pick up the box, and then goes back to the home position. In the above plots, each Cartesian trajectory was recorded separately during each motion and then all of them were combined to produce a complete trajectory execution for a full pick task. The inverse kinematic solutions, or the joint trajectory corresponding to each Cartesian trajectory in the respective moves, is given below.

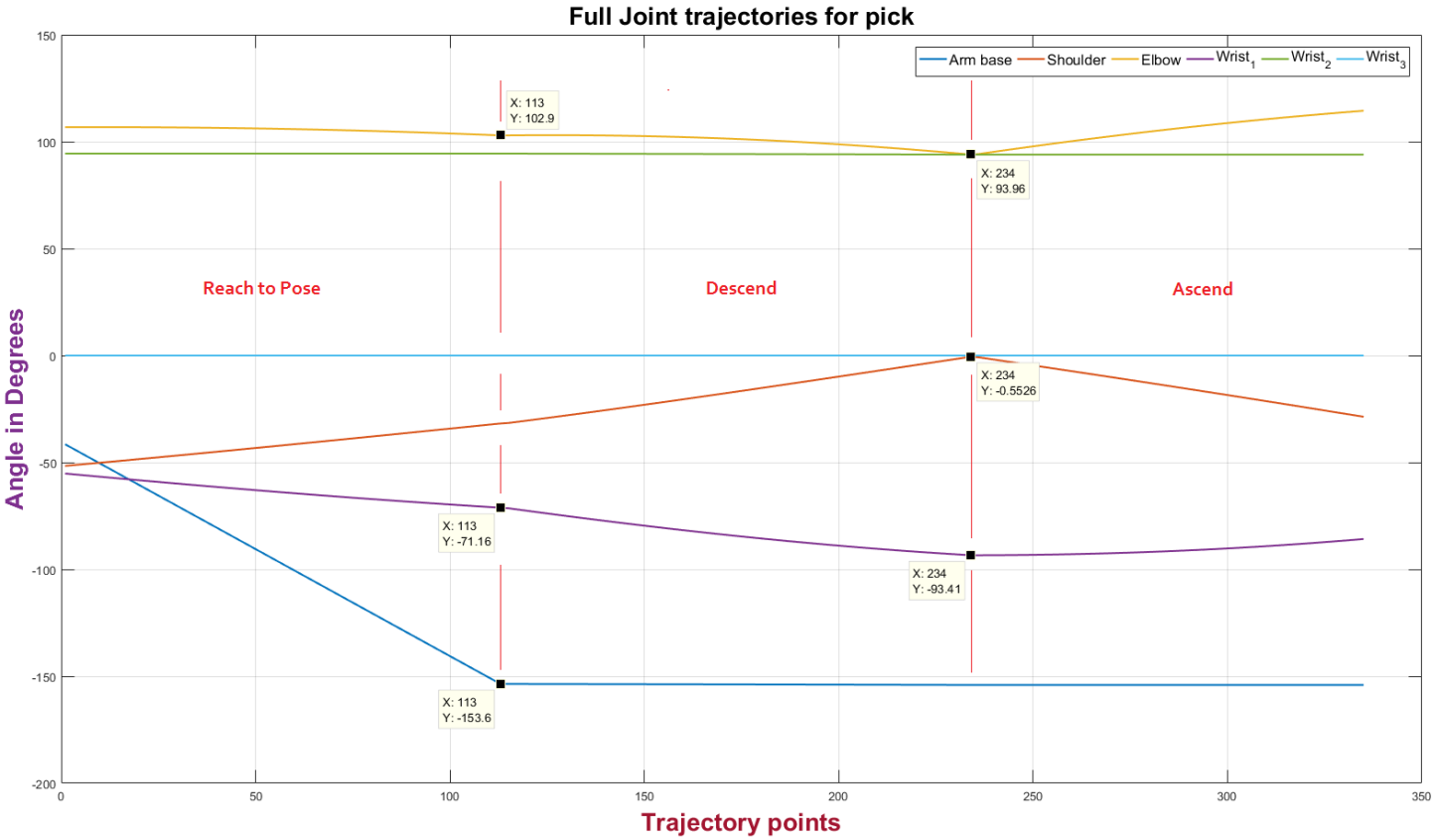


Figure 4.75 : A full joint trajectory for a full Pick task

The above plot shows the joint trajectories for the '6' joints of the UR5 corresponding to the Cartesian moves of the figure (4.74). The regions in the figure (4.75) corresponding to the moves in (4.74) are highlighted in red and it can be observed in the figure that the joint trajectories stay in the limits as prescribed in table (10). For instance, the joint trajectory for joint 'Arm base' stays within the limit "$-175° \leq \theta \leq 60°$" during the 'Reach to pose" phase. This is the phase in which the arm executes the circular move thus indicating the change in change of trajectory. After this phase the trajectory remains constant since the arm moves down and

up for the 'descend' and 'ascend' moves. Similarly, the shoulder joint also stays in the limit "$-90° \leq \theta \leq 0°$" during the 'descend' and 'ascend phases.

4.4.2.3.2 Drop Task

Similar to the pick task the plot below shows the recorded trajectories for the 'drop' of the box which was picked before. The 'Start pose' in the figure again shows the home pose of the arm (gripper). As can be seen this home pose is the same in both figure (4.74) and (4.76). The arm rests in this pose after it has already picked up the box.

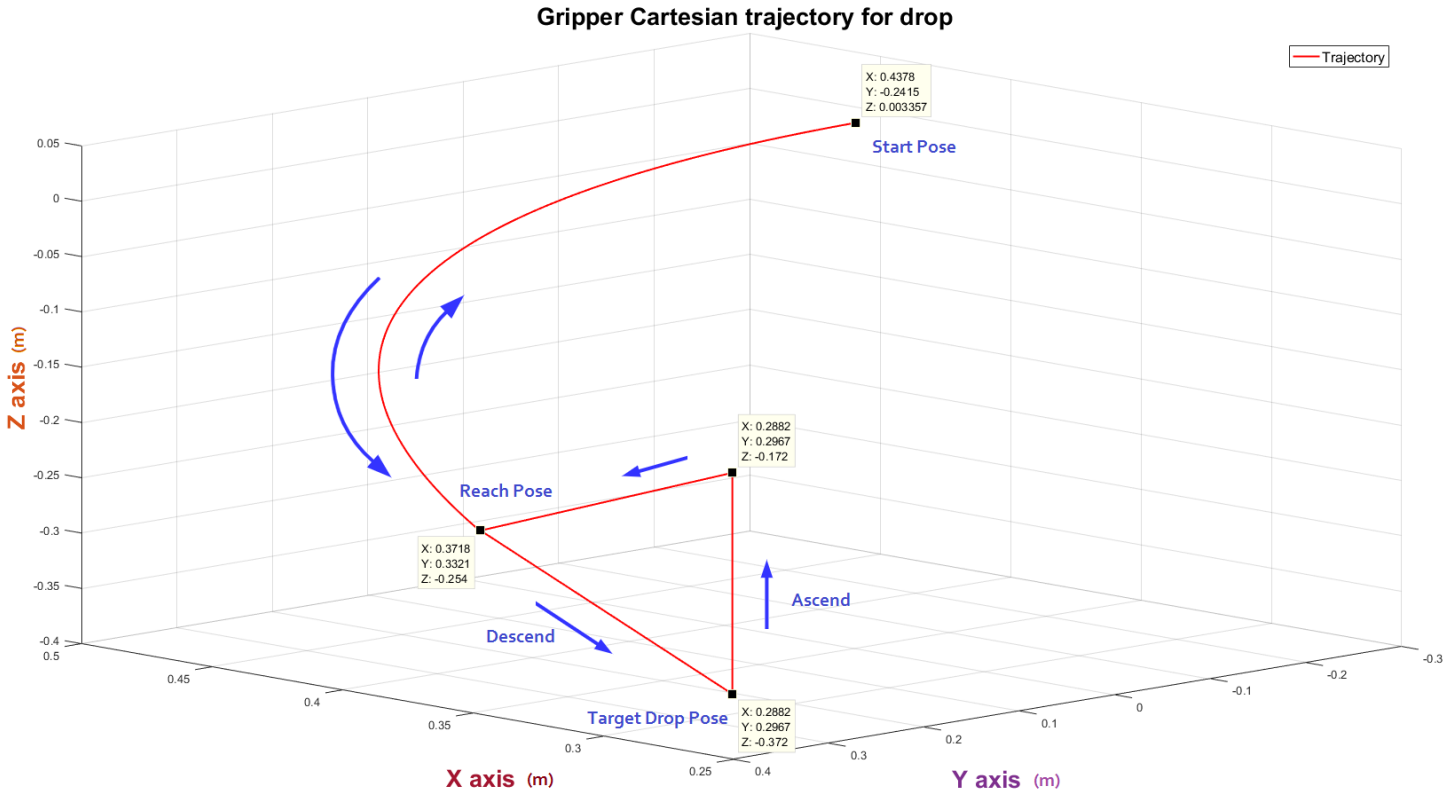


Figure 4.76 : A full Cartesian trajectory for a Drop task

The 'Target drop pose' gives the location of the marker where the box has to be dropped. The motion planner again checks the difference between the current angle "θ_{crnt}" and the target angle "ψ_{tgt}" (step 3 & 7). It then produces a circular trajectory to execute, to reach to the target pose to drop the box. Once above at a specified height above the target pose, it then executes the 'descend' move to place the box on the marker. After detaching the box from the gripper, the gripper or the arm then execute the ascend motion and goes back to the home position. The inverse kinematic solutions, or the joint trajectory corresponding to each Cartesian trajectory in the respective moves, is given below.

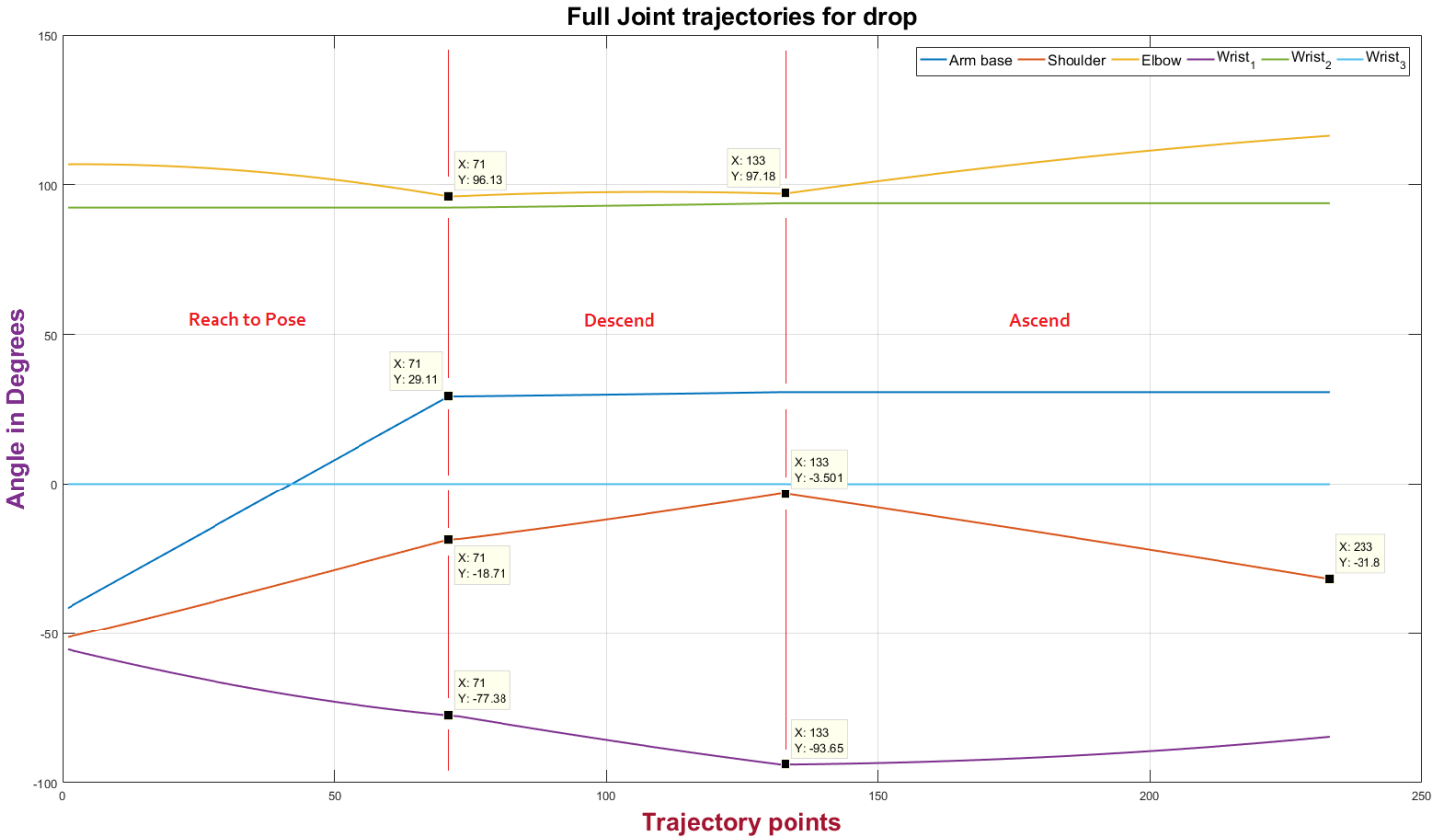


Figure 4.77 : A full joint trajectory for a full Drop task

The above plots again show the joint trajectories for the '6' joints of the UR5 corresponding to the Cartesian moves of the figure (4.76). The regions in red in the above figure correspond to the moves in (4.76) and it can be observed that the joint trajectories again stay in the limits as prescribed in table (10). The joint 'Arm base' stays within the limit "$-175° \leq \theta \leq 60°$" during the 'Reach to pose" phase, This is the phase in which the arm again executes the circular move and the blue trajectory rises till the point it becomes constant. After this phase the trajectory remains constant since the arm moves down and up for the 'descend' and 'ascend' moves. Similarly, the shoulder joint also stays in the limit "$-90° \leq \theta \leq 0°$" during the 'descend' and 'ascend' phases indicate by the orange trajectory. The significant trajectory to note here is the one for the 'wrist_1' joint given in magenta. The trajectory remains in the limit "$-100° \leq \theta \leq 0°$" during the 'descend' and 'ascend' phases, which shows the box is kept upright by the gripper during its motion, both for the pick and drop moves. Similarly, the plots for both the 'wrist_1' joint and 'wrist_2' joint indicate that the orientation of the gripper is also preserved with respect to the "arm_base" during its motion for reaching the target pick or drop pose.

Chapter 5

Task Management

Analyzing the underlying elements, the inherent nature of the picking task encompasses a specific execution order of all the functional elements rather than a pure reactive control. Therefore, a sort of coordination mechanism is required for the execution of these elements in the right order. This execution of each functional elements can be regarded as a specific behavior, leading the robot from one stage to the next. The underlaying mechanism of the coordination framework must make use of some planning to generate the required sequence of actions (behaviors) to execute, while ensuring the synchronization of these actions with logical conditions and external as well as internal events. The resulting mechanism works as a supervisor to control the agent (robot) for accomplishing the task, while guaranteeing the safety properties of the system such as deadlock and collision avoidance, optimality and real-time control etc. While the previous chapters focused on detailing the functional elements involved in the task of picking, this chapter focuses on the coordination of these elements to produce a supervisory control of the robot for the accomplishment of the task. A reflection on the prominent supervisory control approaches is given in the beginning followed by the proposed strategy chosen for the supervisory control of the objective task, presented and discussed in detail.

5.1 Prominent approaches

For the robot control, the methodologies are fundamentally based on the particular task, the environment and the robot itself. The design of the control architecture is heavily influenced by the exact nature of the task, the level of efficiency required, the capability of the robot and the situatedness property. Situatedness refers to the existence of the robot in a complex and challenging environment, and the level of situatedness has a direct impact on the complexity of the control, which in turn strongly affects the global behavior of the robot. Robot control is a vast domain and while the designer is free to leverage the flexibility of existing elements and constraints and develop a methodology of choice, fundamentally there are four classes of robot control methods, which have developed (emerged) over the years by continuous evaluation of different approaches converging into final four ones [77].

The “Deliberate Control” is organized in a network of functional modules constituting the decision-making process. It involves ‘planning’ in advance to carry out the execution of required actions. The sensory process module, model update, planning, and execution module allow complex operations to be performed. The control involves a sequence of sense, plan and act stages, where the planning requires a symbolic representation of the world, i.e. the existence of a world model. This allows the agent (robot) to look ahead in the future and predict the outcome of certain actions in various states and plan in advance to mitigate the unforeseen circumstances. Planning requires a consistent updated word model, and the uncertainty in sensing, action and environment requires frequent planning at the cost of heavy computation, making real time reaction to sudden changes ineffective. Purely deliberate control under the constraints of time becomes infeasible in the existence of an inaccurate world model.

The “Reactive Control” is inspired by the biological notion of stimulus and response. Unlike the deliberate control, reactive control does not maintain a state of the world model, since it does not rely on the complex reasoning process utilized in the deliberate control. Reactive control architecture consists of a collection of preprogrammed concurrent condition action rules with minimal internal state, allowing the agent (robot) to respond rapidly to the changes in the environment. Since the reactive control systems apply a simple functional mapping between the stimuli and the appropriate response, they lack the capability to store information about the world state and do not have the ability to learn and improve over time, thus becoming ineffective for large complex problems. Pure reactive control strategies are only employed on smaller scales at the low-level control.

The “Hybrid Control” combines both the aspects of the reactive and deliberate control by employing a reactive system for low level control and a planner for high level decision making. The objective in this type of control is to leverage the real time response of pure reactivity and a rational planned output of deliberation. The architecture of hybrid control consists of at most three layers, with a reactive layer, a planning layer and a coordination layer providing the communication interface and conflicts resolution for the two control layers. The low-level reactive layer takes care of the immediate safety of the agent (robot) while the high-level deliberation layer utilizes a planner to select the optimal action sequences.

The “Behavior Based Control” consist of a system of purposively built perception action units called ‘behaviors’. Each behavior produces an immediate reaction to a particular sensory input to control the agent, fulfilling a specific objective. Behaviors acting as modules are distributed in the control architecture, interacting between each other to collectively achieve a desired goal of the agent. Each module receives input from a sensor or other modules in the

system and provides output to robot actuators or other modules. These modules are combined in a hierarchical fashion from low level reactive behaviors to high level complex process of reasoning and planning. The behaviors are implemented as control laws either in software or hardware, as processing element or as a procedure. The control architecture consists of a network of interacting behaviors maintaining no centralized world representation, but each module maintaining its own state and a model representation. Behaviors are designed manually by a process called behavior synthesis, and one of the central design challenges of behavior-based systems is action selection or behavior coordination, which is to choose a particular action or behavior from multiple behaviors. Behavior based systems are usually hybrid combining both the reactivity at the lower level and planning at the higher level. The interaction among the behaviors results in emergent behaviors of the robot in the environment.

Considering the development of behaviors and their selection mechanism, a considerable amount of effort has been dedicated in the past. Initially the research was done in the context of artificial intelligence [138] considering the behavior of agent only, but was not applicable to real world scenarios due to prevalent dynamics of the environment and the agents. Thus, the approach was resolved to take into account the dynamics of both the real world and the agent and formalize it into continuous and discrete states. This deduction was applied to behavior coordination for formulating action selection mechanisms. The action selection [139] deals with selecting the 'most appropriate' or the 'most relevant' next action to take at a particular moment, when facing a particular situation. This 'situation' refers to the current state of the system and based on which action selection is split into 'State based' and 'Continuous' mechanisms. Based on the state and continuous topology the ASMs are divided into two groups, "Arbitration" and "Command fusion". Arbitration ASMs allow one or a set of behaviors at a time to take control for a period of time until another set of behaviors is available. Command fusion allows multiple behaviors to contribute to the full control of the agent (robot).

5.1.1 Arbitration

The arbitration selection framework selects one behavior for the robot control from a group of competing behaviors in the current execution cycle. Arbitration is preferred when the control has to select an active behavior among multiple ones, which falls in accordance with the requirement and objective of the system. Arbitration mechanisms are further classified into three categories i.e. Priority based, State based, and Winner take all.

5.1.1.1 Priority based

In priority-based mechanism the action is selected by a central module on pre-assigned priorities. An example of such a framework is the 'Subsumption' architecture [140], in which behaviors with higher priorities in the hierarchy take control of the robot.

5.1.1.2 State based

In state-based framework the action is selected which is sufficiently apt to handle the situation corresponding to the given state. An example of such a framework is discrete event systems [141].

5.1.1.3 Winner take all

In winner take all the action is selected based on a competition among a set of distributed behaviors. The winner among all takes control of the robot. The example of such a framework is 'Activation Networks' [142].

5.1.2 Command Fusion

Command fusion framework allows a set of behaviors to share the control of the system in the current execution cycle. The action selection is a fusion of multiple behaviors to represent a consensual control, such that all the behaviors contribute to the control of the system in a cooperative manner. Based on the fusion of behaviors, the mechanisms classified into four categories i.e. Voting, Fuzzy, superposition, Multiple objective.

5.1.2.1 Voting fusion

In 'Voting' the output of each behavior is interpreted as votes and the action or behavior is selected which has received maximum number of votes. An example is DAMN [143].

5.1.2.2 Fuzzy fusion

In this framework, fuzzy inferencing rules are used to select behaviors. An inference engine uses a rule-based controller that produces a multivalued output of rules, which encodes the desirability of an action. Based on the grade of the desirability the action is selected. The example is fuzzy/multivalued logic approach [144].

5.1.2.3 Superposition

In superposition, behavior recommendations are fused together by linear combinations. Example is motor schemas [145].

5.1.2.4 Multiple objective fusion

This framework is based on an approach to select behaviors based on multiple objective decision theory. An example is the satisfying approach given in [146].

Also included in the behavior control approaches are the class of behavior control languages. Languages specifically designed for the description of behaviors on a conceptual level. The prominent examples are 'XABSL' [147] based on finite state machines, 'Colbert' [148] based on finite state machines supporting concurrent activities, Configuration Description Language (CDL) [149] to specify the configuration of behaviors, and the behavior language by Brooks [150] based on the Subsumption architecture.

5.2 Behavior related works

Behavior synthesis requires an exhaustive analysis of the requirement and the corresponding task allocation. Aspects concerning the nature of tasks i.e., long term, short term, repetitive, plausibility of subtask decomposition, number of allocations, speed of execution, safety of system and the capability of the platform, all have to be considered for designing an effective behavior control of the agent. This is crucial to develop an effective global control of the robot. To give the reader an idea of the variation of different methodologies conforming to the objective, it is worth mentioning a few works on the behavior composition and execution.

A behavior based approach is presented in [151] which utilizes a behavior based architecture called "CAPMPOUT" for autonomous construction tasks. CAMPOUT is a distributed control architecture based on multirobot cooperative control, in which the high-level functionality is composed by the coordination of basic behaviors using both arbitration and command fusion. It used state based and priority-based arbitration, whereas for coupled tasks requiring spatiotemporal coordination of activities, it uses voting and multiple objective behavior command fusion. The construction tasks include acquisition, and precision placement of components, where each task is decomposed into subtasks consisting of a general reusable complex behavior. The cooperative construction task was carried out in a leader follower configuration by two mobile robots, making use of grip force feedback for synchronized manipulation and movement.

Another work in [152] presents a particular behavior based approach for the control of a mobile robot which combines the state formalism of DES with the fuzzy decision making. The technique is based on behavior modulation using fuzzy discrete event systems, that combines both behavior arbitration (BA) and command fusion (CF) approach to coordinate

cooperative and competitive behaviors. The approach is peculiar since it is able to prioritize behaviors while employing both BA and CF. The arbitration is performed by the execution priority of each behavior and fusion is carried out by concurrent activation of all behaviors with varying membership grades using fuzzified functions. A reliable behavior selection is facilitated with the analysis of state-based observability and controllability. Cooperative behaviors after identification are weighed more, where the final action to control the robot is generated using weighted sum of all behavioral actions. The method was tested successfully with effective results on a mobile robot with three primitive behaviors for navigation.

The authors in [153] have worked on the navigation control of a mobile robot which consists of a hybrid control approach integrating partial motion planning with emotion-based behavior coordination. The behaviors are modeled based on the theory of non-linear dynamics and the authors have attempted to use an emotion model in which emotion plays a key role in the system management. Three primitive navigation behaviors are designed along with three primitive emotions to coordinate between these behaviors of the robot during motion. The goal to navigate is provided by a partial motion planner based solely on real-time sensor input, whereas the coordination between behaviors and the partial motion planner is done by a self-confidence emotion, reflecting the ability to fulfill the task. Partial motion planning provides the local target and behavior-based control with emotion mechanism makes the robot achieve this target with obstacle avoidance. The method was tested successfully on a differential drive robot.

The work in [154] presents the behavior arbitration by a learning-based approach in which the behaviors are encoded in hierarchical finite state machines termed as ‘skills’. The skills vary in strength and weakness and performance characteristics of each skill is different based on internal control policies to control the robot. The robot automatically determines the best ‘expert’ skill to use among multiple skills to achieve the same task in the given conditions. This is achieved by a learning algorithm which choses the best skill with maximized performance acquired by minimizing a total regret among all the skills. The algorithm also reorders the selection priority of the ‘expert’ based on change of relative performance in changing environmental conditions. The algorithm was tested successfully in a Segway robot for a soccer playing scenario.

The work in [155] presents the implementation of behavior arbitration with a behavior language based on hierarchical state machines (HSM). The language encompasses different classes of behaviors (states) in which behaviors can be passed as parameters, thus building a hierarchy. The behaviors are customizable template-based modules both for hardware control and independent actions. The behaviors are the fundamental building blocks of the language

with the capability to augment the HSM model. The functioning of the state machine and the passing of the behaviors as parameters is exhibited by simulating primitive navigation behaviors for the robot.

The authors in [156] make use of the finite state machines (FSM) for behavior arbitration for the supervisory autonomous vehicle control. The behavior framework is a hybrid state structure (HSS) with a discrete state system (DSS) on top for arbitration, and continuous feedback loop on the bottom as reactive control. The framework was constructed for the highly situational dependent nature of the DARPA urban challenge in which the high-level controller (HLC) controls the situation and events that are discrete in nature, while the low-level controller (LLC) regulates the continuous states such as vehicle velocity control. The progression of the task is carried out by the HLC and LLC interface, in which the DSS incorporates a hierarchy of meta states, with each meta state having its internal state machine to deal with a situation on the road.

Two behavior control frameworks have presented in [157] for controlling a robot with variable behavior requirements. The first one is based on a finite state machine library and the second one is an independent behavior control framework (BC) based on inhibition and hierarchical structure of dynamic transitioning between states. The state control library (SCL) is used for implementing low to mid-level complexity behaviors requiring structural sequences of actions, with one behavior active at one time only. This limitation is overcome by the implementation of BC which utilizes a behavior inhibition tree to prioritize behaviors, to control multiple aspects of an agent to be controlled simultaneously. Both the frameworks can be implemented independently or can work in tandem to handle a complex behavior application, in which the SCL is used to implement finite state machines within the individual behaviors of BC control framework.

In [158] Alejandro et al. authors have worked on the development of a layered architecture for behavior control utilizing behavior trees for action coordination and behavior execution. Behavior tees are an evolution of hierarchical finite state machines (HFSM), where arbitration is executed by replacing a state with a procedural action. In the proposed framework, the top layer is used for planning while the middle layer consists of behavior tress and the lowest layer consists of the low-level controllers for action execution. The top layer automatically generates the plan while the behavior trees coordinate the actions to realize the plan. The actions and their corresponding low-level controllers are coordinated by the lowest layer. The framework was implemented in the open source behavior tree library for ROS, and tested on a humanoid robot for grasping task by first walking to the object and then trying successive grasps to hold it.

To react to unforeseen scenarios and adapt by providing the system a flexibility to change the behavior on the fly is a fundamental functionality. For this the authors in [159] have presented high level approach to define and execute complex robot behaviors, that takes into account the operator interaction in the form of human supervision to make critical decisions within behaviors, thus leading to adaptive autonomy with several layers of autonomy in advance. The approach consists of a behavior engine 'FlexBE' which is based on hierarchical state machines, that takes into account factors of synchronization to avoid runtime failure, consistency across behavior modification, and collaborative autonomy by blocking critical decisions. The framework was effectively used with multiple robotic systems in the context of DARPA and ARGOS robotic challenge to deal with unforeseen scenarios and adapt the behaviors in runtime.

FlexBE[Ref] is an open source behavior framework implemented in ROS and was further used in [160] for the behavior control of an ATLAS robot for the DRC final challenges. As 'FlexBE' provides operator interaction, this collaborative autonomy feature was used to implement the state machines generated by the end to end approach, i.e. from specifications to automatic code generation. The behavior of the robot is controlled by the use of state machines encoding a synthesized plan in the software, based on a formal specification of linear temporal logic (LTL), encoding system capabilities, the constraints, the task and the desired reaction to low level failures. A reactive mission plan satisfying the formal specifications of LTL is synthesized to automatically generate executable state machine that meet the desired goals, i.e. achieve the task goal or correctly react to failure.

5.3 Strategy and framework

When analyzing the dynamics of the global system, the dynamics of the picking task is event driven rather continuous, requiring the system to progress from one state to the next to successfully arrive at the objective. This highlights the fact that every functional element or stage corresponds to a specific state of the system. A successful execution of the global task requires the progression of the system from one state to the next. This necessitates the requirement of the system to be modelled in the discrete domain. In addition, the behavior of the robot to suffice each state requires a flexible control approach rather depending only on a basic reactive control. The flexibility can be induced in terms of a hybrid approach employing a behavior framework on top for action coordination, and reactivity on bottom for action execution. Keeping in view the requirement to manage the global execution of the task and the available frameworks, the approach adopted in this thesis is to consider the global system in the "Discrete Event System (DES)" domain and model the execution of all the functional elements within the system with 'Petri nets'.

5.3.1 Petri Nets

Petri nets were introduced by Carl Adam Petri [161] in 1962. This was a new modeling technique in the domain of discrete event systems, which was based on the concept of asynchronous and concurrent operations of the parts of the system and their relationships represented by a graph. In this modeling framework, the dynamic behavior of the system is represented by a graphical representation supported by a well-defined mathematical theory. These frameworks have been widely used to model dynamic systems, due to their ability to model concurrency, parallelism, synchronization, and decision making. These aspects make them good candidates to model task execution in robotics, where the task represents the sequence of actions the system must carry out to accomplish the global goal. Research in this domain with application to robotics is ubiquitous employing Petri nets for basic behavior control of a robot to cooperative behavior coordination of multi-robot systems. An overview of some significant research works is given as under.

The authors in [162] employed Petri nets for task decomposition and execution in a Tripodal control architecture of a mobile service robot. The architecture is based on a hybrid approach composed of a deliberate, sequencing and reactive layers. Each layer consists of a configuration specifying the relation among tasks, processes and behavior. For a given configuration the task is decomposed into individual process modules consisting of a Petri net for execution. The proposed architecture was tested successfully on a service robot for a package transportation task consisting of a docking and manipulation maneuvers. The flexibility of the architecture was further tested in [163] in terms of reusability and scalability by adding a new task to perform on a completely different platform.

In [164] Petri nets were used for the supervisory control of robots for safe navigation. An online controller was implemented for the coordination of multiple robots in a semi structured environment, with the requirement of replanning and deadlock resolution. The plan was initially defined by the operator and then compiled into a Petri net to be forwarded to the plan executer. Similarly, in [165] the authors have used Petri nets to design and implement an execution control module to control the navigation and guidance of a remotely operated vehicle (ROV) in deep sea. The Petri net based control module generates sequences of task activation and deactivation operations to execute the desired command to maintain the system in admissible configurations. The execution control module automatically detects the conflicts between the tasks and reconfigure the execution level using only the information embedded in the input / output.

Chung et al, have presented [166] the modulation of multiple navigation behaviors for a service robot with the Petri net formalism. The model under consideration consists of two navigation behaviors and the framework quantitively computes the most desirable navigation behavior in runtime according to events fired by the navigation components. The selection criterion is based on the behavior which leads the robot to the goal faster with guaranteed localization. This is decided by comparing the frequencies of the transitions fired to execute specific behavior under corresponding scenario. The robot thus selects the behavior which has a higher frequency than the other. The framework was tested on a service robot for mobile navigation in a museum in crowded environments. The same authors used the framework to test two different navigation behaviors in their further work in [167].

A robotic soccer scenario was tested [168] with the implementation of Petri net task plan. The framework consists of a multilayer task model with different degrees of abstraction from the lowest to the highest, consisting of an environment model, action model, and action coordinator model with the task plan is embedded within it. Several different models were used to test the influence of realism of actions and environment models. The framework was used to test the impact of goal achievement with different task plans. Petri nets have also been employed for the coordinated motion control of a robot with high degree of freedom. In [169] the global task given to the humanoid robot is decomposed into several subtasks, representing the sequence of actions the subsystems must carry out to accomplish the task goal. Each subtask is modelled by a Petri net and the coordinated execution of both parallel and sequential subtask is carried out by Petri nets.

Further the work in [170] presents a GUI based framework utilizing hierarchical Petri nets to coordinate the activity of modules implementing primitive actions to control a mobile robot. The framework called 'Robograph' consists of Petri nets as a high-level task application programming language to execute tasks in real time. The tasks are defined using a Petri net editor while a dispatcher executes the different Petri nets on user request. This dispatcher coordinates the execution of control modules and sequences their function according to the plan defined as a Petri net. The work in [171] consists of a Petri net (PN) based decentralized approach for the coordination control of multirobot systems. The behavior of each robot is modeled by a Petri net and the coordination between the robots is handled by a set of linear temporal logic (LTL) rules. The LTL formulas define the events and changes in state and automatically augment the PN model of each robot to handle the communication of changes in the environment, and execution of actions required for coordination between the robots. Thus, the augmented PNs and the LTL formulation allow to form local PN supervisors to coordinate the collaborative actions between the robots. Last but not least [172] the authors have used the concept of "pages" to develop a hierarchical Petri net, modeling the activities of

a robotic system decomposed into five layers. Each page represents the Petri nets for that specific layer. The overall specification of HPN results in the concurrent execution of activities describing the interaction of the subsystems and lower lever control components.

While reflecting back on the review, perhaps [162] and [166] are the adjacent to the work presented in this thesis with a relative difference of the environment i.e. industrial versus domestic, the type of Petri nets used and the difference of scenarios for switching the behavior of the robot.

5.3.2 Theoretical framework

A 'Petri-Net' is bipartite graph consisting of nodes and arcs. The nodes are categorized into two types consisting of places and transitions and they are connected to each other via arcs. Each arc is directed from an element of one set (place or transition) to an element of the other set (place or transition) producing a directed graph. Directness refers to the fact that the arcs run from a place to a transition or vice versa and never between places or between transitions. The node connected by the arc 'to' the transition is called the input place denoted by $I(t_i, p_j)$ and the node connected by the arc 'from' a transition is called the output place of the transition denoted by $O(t_i, p_j)$. That is an arc is directed from a place $'p_j'$ to a transition $'t_j'$ if the place is an input of the transition and an arc is directed from a transition $'t_j'$ ' to a place $'p_j'$ if the place is an output of the transition. The set of input places for the transitions forms an input function $I(t_i)$. Likewise, the set of output places forms the output function $O(t_i)$.

For the dynamic analysis, the places model 'conditions' represented by a circle, the transitions model events represented by a bar in the graph. The arcs are represented by a directed arrow from a place to a transition and vice versa. The Petri net is executed by firing transitions. The transitions are fired when an event occurs and the state of the system changes. The transitions are fired only when they are 'enabled'. A transition is enabled when each of its input place has at least one token. Technically the execution of the Petri net takes place by the assignment of 'tokens' to the places and is called the marking $'M'$ of the Petri net Tokens reside in the places and are represented by a 'dot'. The tokens are removed from the input place of the transition to the output place when a transition gets fired. The number of tokens removed or deposited depends on the weights assigned to the arcs.

The marking $'M'$ of the Petri net gives the assignment of tokens to the places in the net. The number of and positions of tokens in the net may change during the execution of the

net. The number of tokens in each place are given by the vector $m=(m_1,m_2.....m_n)$. The number of tokens in place 'p_j' is 'm_i' where $i=1.....n$. A marking function $m:P\rightarrow N$ from the set of places to the set of natural numbers $N=\{0,1,2,....\}$ allows to use the notation '$m(p_i)$' to specify the number of tokens in places 'p_i'. For a marking 'm', $m(p_i)=m_i$. Firing a transition changes the marking of the net from 'M' to 'M''. Regarding the change, since only enabled transitions are fired, the number of tokens in each place always remains non-negative upon firing of a transition. Mathematically the change to the new marking is given by

$$M'(p)=M(p)-I(t,p)+O(t,p) \qquad (5.1)$$

5.3.2.1 Mathematical definition

Formally a Petri net is defined as a 5 tuple, $PN=(P,T,F,W,M_o)$ where :

$P=\{p_1,p_2,.....,p_n\}$ is a finite set of places.

$T=\{t_1,t_2,....,t_n\}$ is a finite set of transitions.

$I=P\times T$ represents the arc connections from places to transitions.

$O=T\times P$ represents the arc connections from transitions to places.

$F\subseteq(P\times T)\cup(T\times P)$ is a set of arcs denoting the flow relation.

$W:F\rightarrow\{1,2,3,....\}$ is a weight function denoting the weight of an edge (arc).

$M_o:P\rightarrow\{0,1,2,3,....\}$ is the initial marking.

$P\cap T=\varnothing$ and $P\cup T\neq\varnothing$

A Petri net with the marking 'M_j' is called a 'Marked' Petri net and is called an ordinary net if all of its arc weights are '1'. The figure (5.1) shows a basic Petri net with two places and one transition.

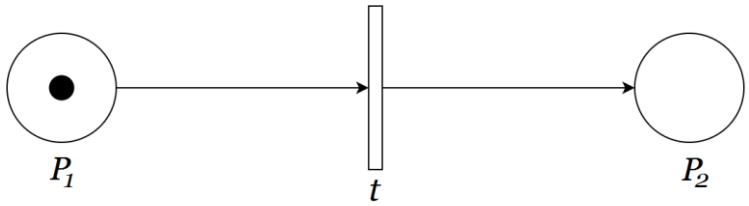


Figure 5.1 : A simple Petri net

A transition without any input place is called a source transition, and one without any output place is called a sink transition. In the above figure 'P_1' and 'P_2' represents the conditions or states of the system and 't' the triggering of an event. 'P_1' can be visualized to represent a processor running and 'P_2' the processor idle. As soon as a processing job is finished by the processor, the transition 't' is triggered and the token is shifted from 'P_1' to 'P_2' making the system achieve idle state.

5.3.2.2 State space

The state of a petri net is defined by its marking. The firing of a transition changes the state of the net, thus state of the system. The state of the net changes according to the following rules.

1. A transition 't' is enabled if each of the input places 'p_i' of 't' contains at least the number of tokens equal to the weight of the directed arc connecting 'p_i' to 't'.

2. An enabled transition may or may not fire depending on the related event to occur. The firing of the enabled transition 't' 'removes' from each input place 'p_i', the number of tokens equal to the weight of the arc connecting 'p_i' to 't', and 'deposit' equivalently the number of tokens in the output place 'p_j'.

The state space of a Petri net with 'n' places is the set of all markings 'N^n'. The change in the state caused by the firing of a transition is given by a partial function 'δ' called the next state function. Application of this function to a marking 'm' and transition 't_j' yields the value of the marking that results from the firing of transition 't_j' in marking 'm'. Petri nets possess a number of properties and in the context of a modeled system, they allow the system designer to identify the presence or absence of the application specific functional properties of the system under design. Two types of properties can be distinguished, behavioral and structural ones. The behavioral properties are those which depend on the initial state or marking of a Petri net. The structural properties do not depend on the initial marking of a Petri net but rather on the topology or structure of a Petri net. For the properties overview the interested readers can refer to [173].

5.3.2.3 Analysis

In addition to the modeling properties of the petri nets, the analysis of the Petri nets, and hence the overall system is carried out by three approaches. The reachability analysis, the matrix-equation approach, and overall system simulation. The reachability analysis is conducted with the construction of a reachability or coverability tree. This tree consists of nodes representing the markings, and arcs representing the transitions. For a given node 'x', additional nodes are added to the tree for all markings that are directly reachable from 'x'. For each transition 't_j' which is enabled in the marking for node 'x', a new node with marking '$\delta(x,t_j)$' is created and an arc labelled 't_j' is directed from the node 'x'to this new node. The process is repeated for all the new nodes and the tree is evolved. For the simulation analysis, a simulation tool is used to run an execution algorithm to run a Petri net. The simulation is carried out by deciding the initial marking of the net and the set of all enabled transitions in that marking. Then the simulation is executed to see the evolution of the net by firing of transitions and state changes. The simulation is stopped on a certain stopping criterion or occurrence of a deadlock marking due to disabling of all transitions.

5.3.2.4 Incidence matrix and state equation

Another approach for analyzing the system modeled with Petri nets is the incidence matrix equation. The matrix equations [173] govern the dynamic behavior of concurrent systems modeled by Petri nets with the assumption that the net under consideration is a pure one. For a Petri net 'N' with 'n' transitions and 'm' places, the incidence matrix '$A=[a_{ij}]$' is a '$n\times m$' matrix of integers and its typical entry is given by

$$a_{ij} = a_{ij}^{+} - a_{ij}^{-} \tag{5.2}$$

where '$a_{ij}^{+} = w(i,j)$' is the weight of the arc from transition 'i' to its output place 'j' and $a_{ij}^{-} = w(i,j)$' is the weight of the arc to the transition 'i' from its input place 'j'. From the transition firing rule explained in section (5.3.2.2) it can be seen that 'a_{ij}^{-}', 'a_{ij}^{+}', and 'a_{ij}' respectively represent the number of tokens removed, added and changed in place 'j' when transition 'i' fires once. Transition 'i' is enabled at marking 'M'if

$$a_{ij}^{-} \le M(j) \qquad j = 1,2,\ldots..,m \tag{5.3}$$

In the matrix equations the marking 'M_k' is given as '$m\times1$' column vector. The 'j_{th}' entry of 'M_k' denotes the number of tokens in place 'j' immediately after the 'k_{th}' firing in some firing sequence. The 'k_{th}' firing or the control vector 'u_k' is an '$n\times1$' column vector of '$n-1$' 0's and one nonzero entry, a 1 in the 'j_{th}' position indicating that transition 'i' fires at the 'k_{th}' firing. Since the 'i_{th}' row of the incidence matrix 'A' denotes the change of the marking as the result of firing transition 'i', the state equation for the Petri net is then given by

$$M_k = M_{k-1} + A^T u_k \qquad k = 1,2, \ldots \tag{5.4}$$

If a destination marking 'M_d' is reachable from 'M_o' through a firing sequence $\{u_1, u_2, \ldots\ldots, u_d\}$, then the above state equation for $i = 1,2,\ldots\ldots d$ becomes

$$M_d = M_o + A^T \sum_{k=1}^{d} u_k \qquad \Rightarrow \qquad A^T x = \Delta M \tag{5.5}$$

where '$\Delta M = M_d - M_o$' and '$x = \sum_{k=1}^{d} u_k$'. Here 'x' is a '$n\times1$' column vector of nonnegative integers and is called the firing count vector. The 'i_{th}' entry of 'x' denotes the number of times that transition 'i' must fire to transform 'M_o' to 'M_d'.

5.3.3 Implementation

For the implementation of the framework for robot behavior control and task management, the standard approach has been adopted which consists of three sequential steps as given below in the figure.

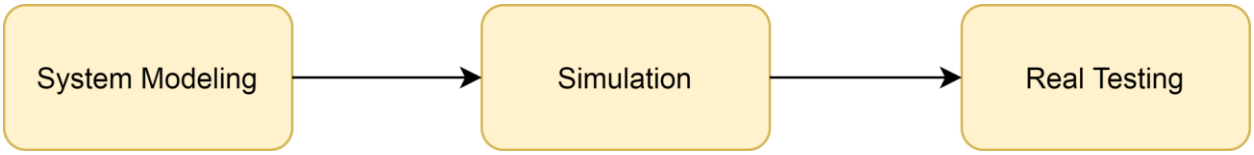


Figure 5.2 : System analysis approach

The system was modelled with Petri nets for representing discrete event dynamics. The complete Petri net graph was transformed to the mathematical representation, using equations of incidence matrix and state space and was simulated in Simulink (MATLAB) to observe the evaluation of states in the net and identify dead locks if there exist any. The conclusion of the simulation enabled to implement the framework of Petri net on the real robot for behavior

management. The real testing consisted of the implementation of modeling framework on the software platform of C++ in Robot Operating System (ROS).

5.3.3.1 System Modeling

The global task of the robots consists of three primitive behaviors which can be considered as global behaviors. These are localization, navigation and manipulation. Each of these behaviors consists of sub-behaviors encoded in each state of the Petri net. For modeling the behavior of the robot, a Petri net is created for each of the global behaviors and referred to as the state machine. The three Petri nets corresponding to localization, navigation and manipulation are connected to form a complete net to manage the full behavior of the robot. The nets are connected according to the 'client-server' topology given in section (5.3.3.2). In the following sections the graph for each global component (behavior) is presented separately.

5.3.3.1.1 Localization state machine

The localization state machine is responsible for managing the localization of the robot at each instant. The state machine uses the information from the markers and the current position of the robot to extract the position of the robot in the global map. It manages the control of the cameras installed on the robot used for scanning and detecting the markers. The state machine is invoked both by the navigation state machine and manipulation state machines. Before the start of the navigation the robot requires to localize itself in the global map and correct its pose with respect to global coordinates. At the end of the navigation it has to repeat the process. This is acquired by detecting the markers on the racks. During manipulation the robot requires to localize the package to pick from the pallet. The cameras have to be scanned consistently and sequentially in each localization request. The control flow of the cameras and extraction of the marker coordinates is presented by the petri net below. There are in total four cameras installed on the robot. One RGB (Axis) camera at the front right side and one RGB (Axis) camera at the rear left side. Two RGB depth cameras (PTU left and PTU right) are installed at the front face of the robot. The RGB (Axis) cameras are used for detecting the markers of the racks and the pallets inside the environment whereas the RGB depth cameras are used to detect the packages on the pallets. The states in the petri net are designated for the execution control of each camera separately. Place '3.0' consists of executing both 'Axis' cameras at the same in scan mode to detect a marker on either side if there is any in the environment. Places '4.0 – 4.1' are reserved for the 'Axis' front and '5.0 – 5.1' for Axis rear.

5.3.3.1.1.1 Petri Net

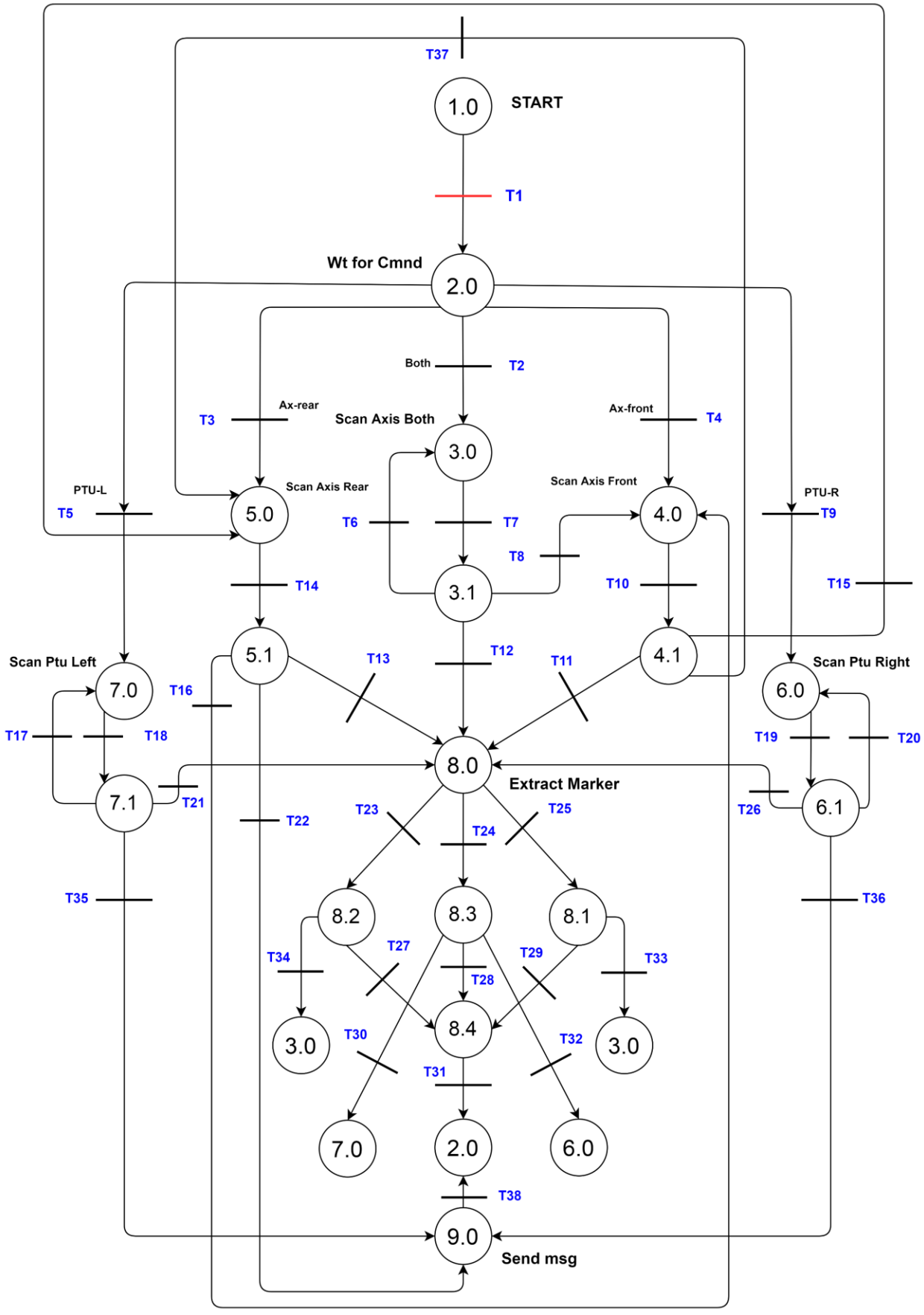


Figure 5.3 : Petri net Graph of Localization state machine

Likewise, places '6.0 – 6.1' are reserved for control and scanning of 'PTU right' and places '7.0 – 7.1' are for 'PTU left'. A comprehensive detail of the rest of the places i.e. states along with their corresponding transitions is given below.

Places

P 1.0: Start

P 2.0: Wait for command

P 3.0: Scan with both RGB cams

P 3.1: Evaluate marker detection

P 4.0: Scan with front RGB cam

P 4.1: Evaluate marker detection

P 5.0: Scan with rear RGB cam

P 5.1: Evaluate marker detection

P 6.0: Scan with 3D depth cam right

P 6.1: Evaluate marker detection

P 7.0: Scan with 3D depth cam left

P 7.1: Evaluate marker detection

P 8.0: Extract marker ID and coordinates

P 8.1: Evaluate marker ID for Pallet

P 8.2: Evaluate marker ID for Rack

P 8.3: Evaluate marker ID for Box/Package

P 8.4: Publish marker ID on topic

P 9.0: Send message no marker found

Transitions

T 1 : OK

T 2 : Scan both cam = true

T 3 : Scan rear cam = true

T 4 : Scan front cam = true

T 5 : Scan left PTU = true

T 6 : Marker det = false & cntr = 1

T 7 : Both_scan done = true

T 8 : Marker det = false & cntr > 1

T 9 : Scan right PTU = true

T 10 : Front_scan done = true

T 11 : Marker detected front cam = true

T 12 : Marker detected both cam = true

T 13 : Marker detected rear cam = true

T 14 : Rear_scan done = true

T 15 : Marker det front = false & cntr = 1

T 16 : Marker det rear = false & cntr = 1

T 17 : Marker det ptu_left = false & cntr = 1

T 18 : Ptu_left_scan done = true

T 19 : Ptu_right_scan done = true

T 20 : Marker det ptu_right = false & cntr = 1

T 21 : Marker detected ptu_left = true

T 22 : Marker det rear = false & cntr > 1

T 23 : Rack Marker detected = true

T 24 : Box Marker detected = true

T 25 : Pallet Marker detected = true

T 26 : Marker detected ptu_right = true

T 27 : Rack ID = true

T 28 : Box ID = true

T 29 : Pallet ID = true

T 30 : Box ID = false & cam = Ptu_left

T 31 : Marker published = true

T 32 : Box ID = false & cam = Ptu_right

T 33 : Rack ID = false

T 34 : Pallet ID = false

T 35 : Marker det ptu_left= false & cntr > 1

T 36 : Marker det ptu_right = false & cntr > 1

T 37 : Marker det front = false & cntr > 1

T 38 : message_sent = true

5.3.3.1.1.2 State Space

There are 18 places (states) in the petri net and 38 transitions. According to equation (5.2), the incidence matrix '$A = [a_{ij}]$' for the petri net is of the size 18 x 38, the initial marking 'Mo' is '18 x 1' and the vector 'x' is of the size '38 x 1'.

5.3.3.1.1.3 Simulation 1

A simulation corresponding to different scenarios was performed in 'MATLAB' for verifying the firing of the transitions and correct evolution of the resulting states. The first simulation was carried out for the localization of the robot inside the environment that consists of detecting a correct marker on the racks. For this, the 'Axis' RGB cameras are executed in the petri net i.e place '3.0 – 3.1'. The evolution of states resulting in firing of corresponding transitions was extracted in the form of a chrono graph to show the time spent and change in the states. A second chronograph was extracted for the transitions as well to show their firing with respect to time. In figure (5.3) the system is in state '1.0' for first 10 seconds. Then after 10 seconds it is waiting for an external command to execute any cameras. After 1 second 'T1' is fired and it receives a command to execute both RGB cameras to scan for a marker. The system enters state '3.0' and does a quick scan and 'T7' is fired, does not detect a marker and 'T6' is fired. It then goes back to the previous state '3.0' and restarts scanning in step mode. It performs the scan for 5 seconds and detects the markers and 'T12' is fired. After detection the system enters state '8.0' to extract the coordinates of the marker and check the right ID. If the exact required marker is detected in the global list, transition 'T23' fires entering state '8.2'. It then publishes the marker in state '8.4' on the ROS topic for the other state machines, which called the localization state machine. After publishing the system goes back to state '2.0' and waits for the next command to arrive.

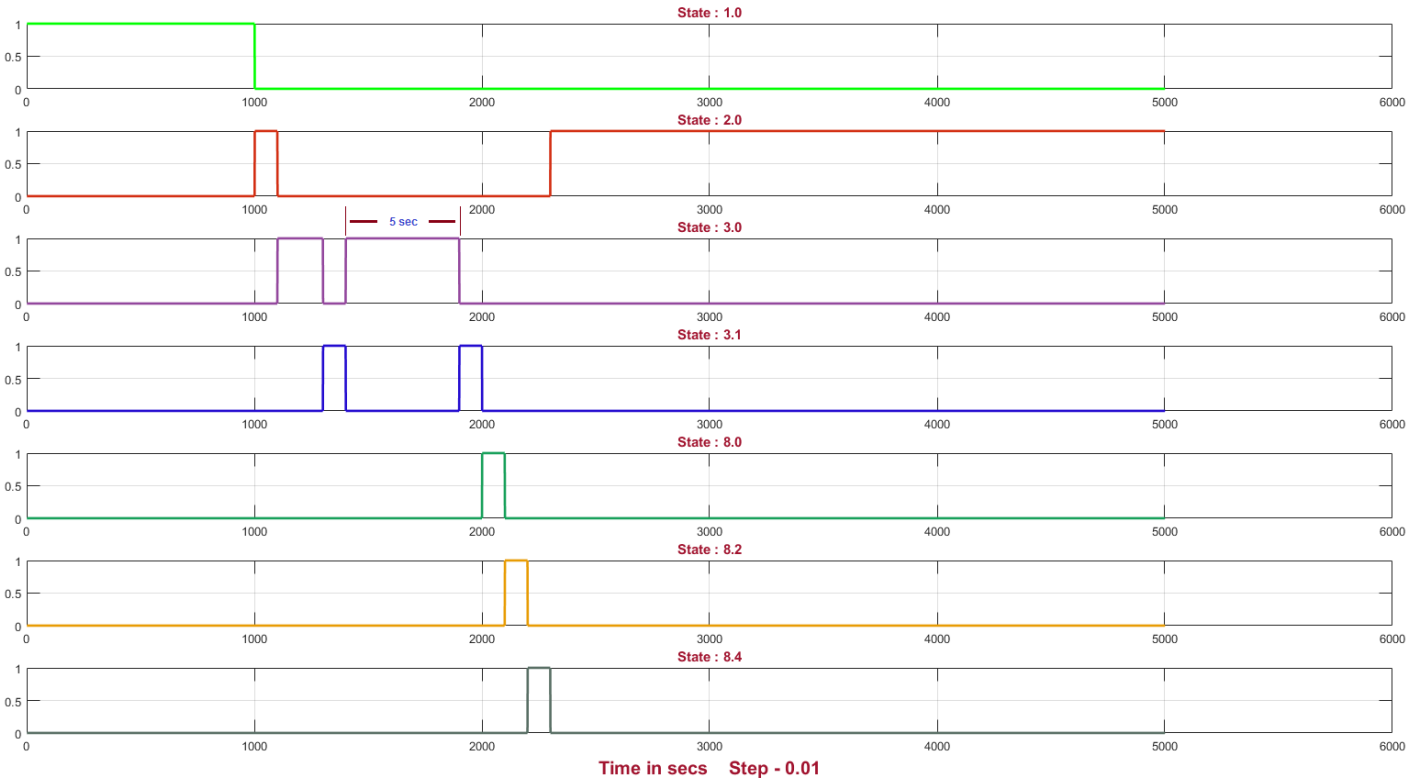


Figure 5.4 : Evolution of states for localization process

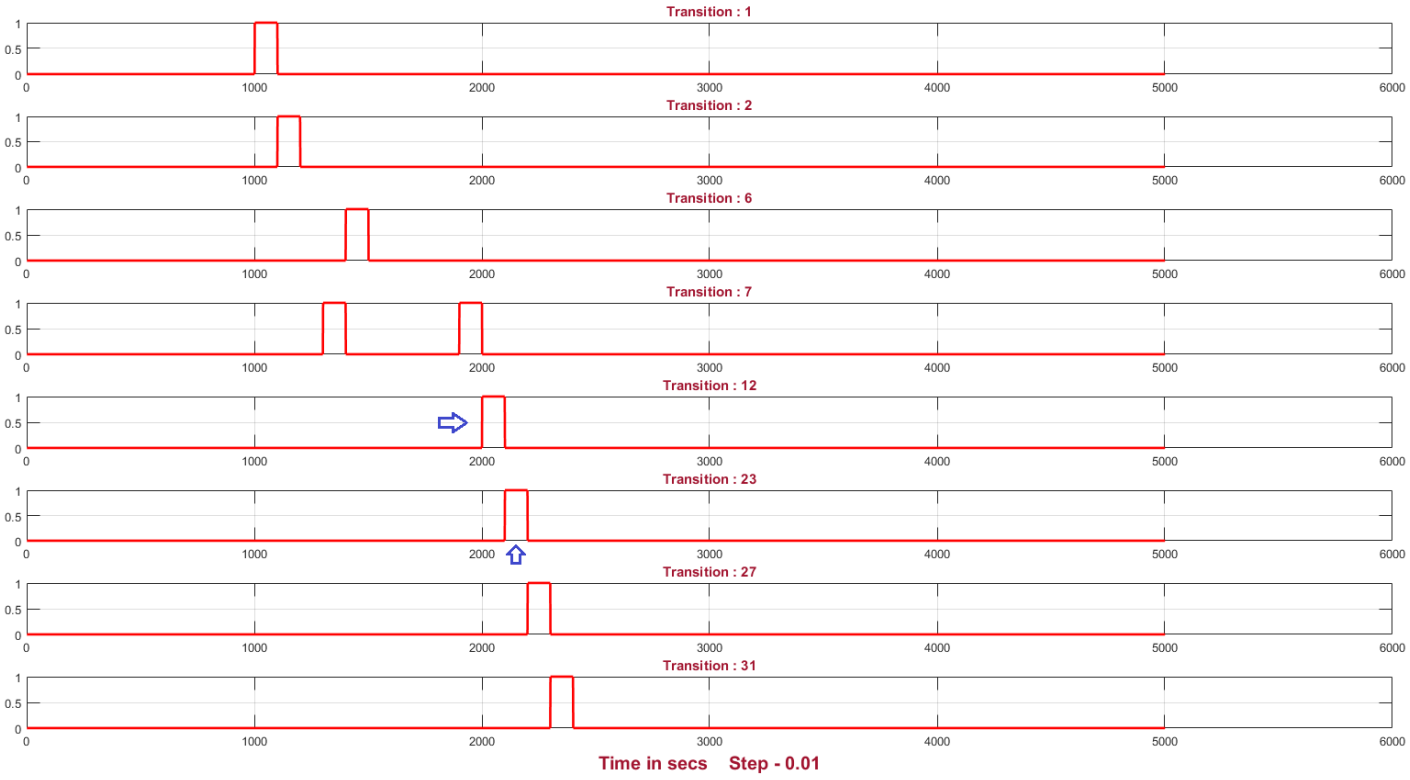


Figure 5.5 : Firing of transitions corresponding to states

5.3.3.1.1.4 Simulation 2

The next simulation was carried out to detect a box / package on the pallet once the robot has arrived to the right pallet. For this either 'PTU right' or 'PTU left' is used, depending on which side of the robot the pallet is. For this simulation it was assumed that the pallet is on the right side and the robot has already aligned itself to the pallet. Again, as before the system waits for the command. Once the command to execute the camera is received, the system enters state '6.0' and camera starts scanning the pallet area. Based on the actual camera speed (rotation) for scanning, the time for a one rotation scan was noted to be 5 seconds. In the figure after 5 seconds the scan is completed, transition 'T26' is fired and the system enters state '6.0' for evaluating if any marker is detected. If one or more markers are detected then transition 'T24' is fired and the system enters state '8.0' to extract the ID and coordinates of the markers. Once the exact marker has been extracted 'T28' fires, and the system publishes the marker ID and coordinates on the ROS topic so that the 'Arm' state machine can use this information to plan the motion of the arm to the target coordinates of the box / package. The system after publishing the marker returns to state '2.0', i.e. it waits for the next command to arrive.

The localization state machine is called either by the navigation state machine or arm state machine. The 'Navigation' state machine uses localization state machine to detect a marker on the racks for localizing the robot, or detect a marker on the pallet once it has arrived to the picking location. It has to be noted here that the simulation '1' was performed for robot localization, the same can be used for pallet localization, by using the same states as before but activating transition 'T25' and using state '8.1'. Once the marker ID and coordinates are published onto the topic, the navigation state machine aligns the robot close to the pallet using the x and y coordinates. Then the 'Arm' state machine will again call the localization state machine to detect and extract the coordinates of the box / package. Also, before each call of the localization state machine, either for detecting a rack marker, a pallet marker, or a box marker, the respective markers ID's or the list of the markers is sent by the other state machines, so that the localization state machine can extract the right marker from a global list in states '3.1', '8.0', '8.1' and '8.2'.

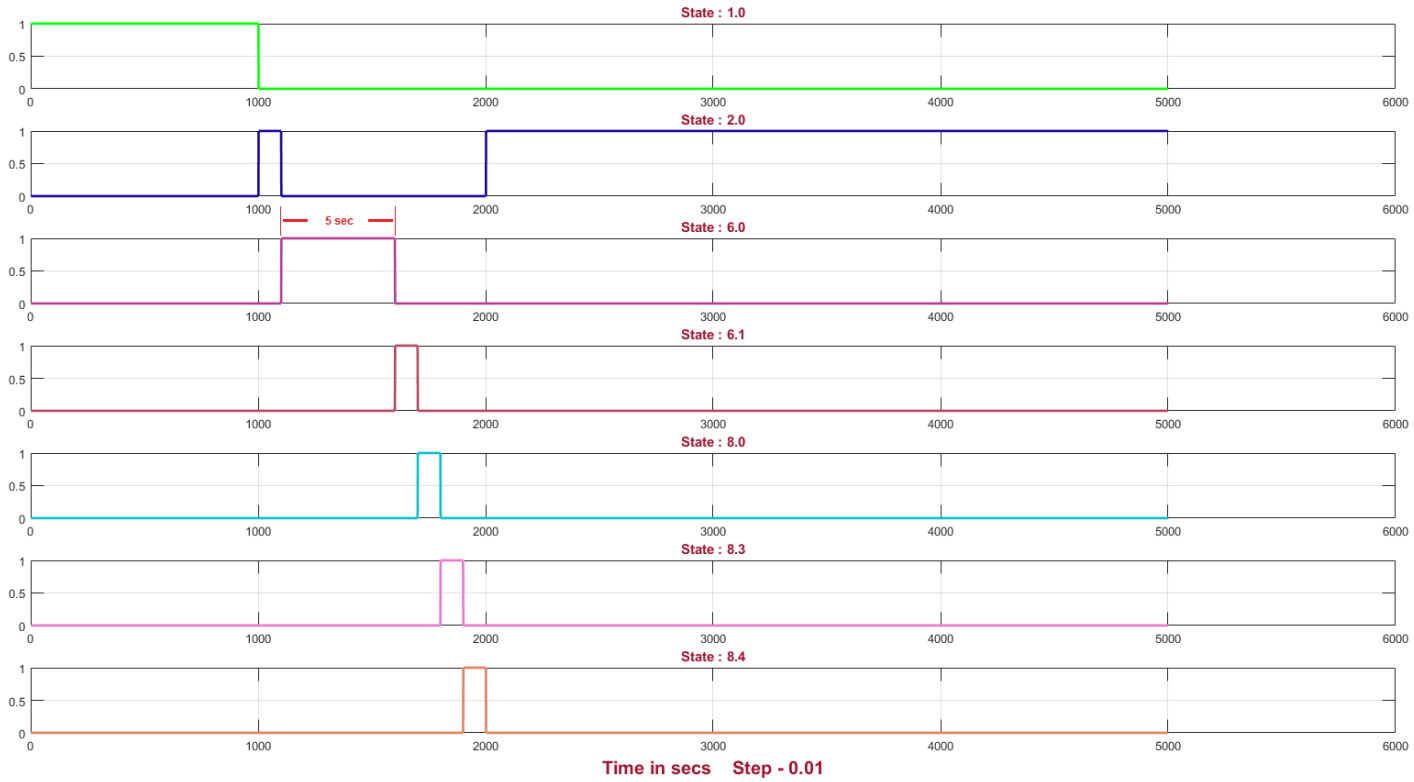


Figure 5.6 : Evolution of states for localization process

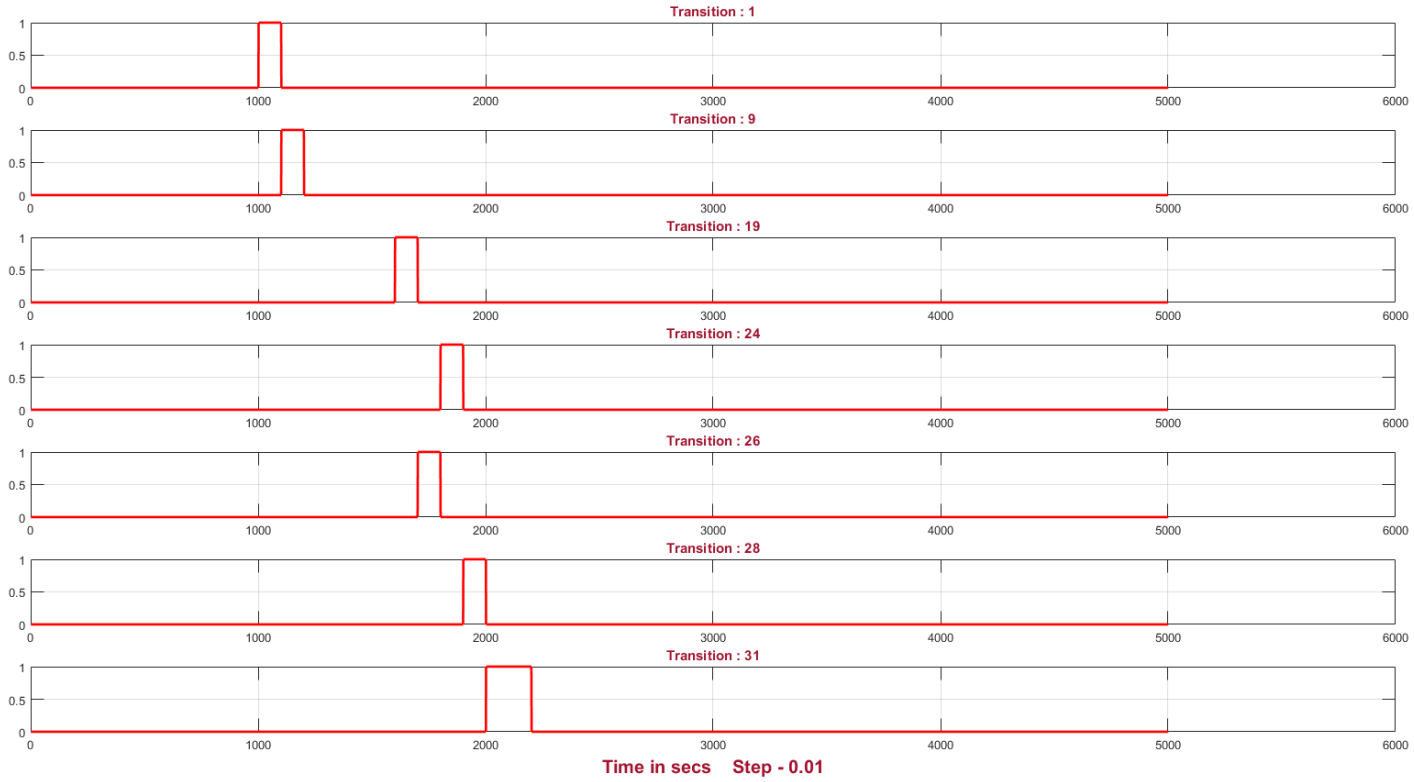


Figure 5.7 : Firing of transitions corresponding to states

5.3.3.1.2 Arm state machine

The arm state machine is responsible for managing all the elements involved in the pick and drop of a package. These elements consist of localizing the package, executing the motion of the UR5 arm, activating and deactivating the pump for pick and drop, and robot position adjustment in case the pick or drop target point is out of reach of the robot. The overall picking task is achieved by executing motion of the arm in specific sequences for both pick and drop respectively. Each sequence corresponds to a specific state of motion, where the execution of the state is controlled by the state machine. For each motion sequence the arm executes a specific trajectory, where the trajectory generation has already been explained in section (4.4). Significantly the arm state machine is conceived to manage the pick and drop of the package and the requirement of a retry in case of a failure and a successful pick and drop. A failure might occur in the condition the package is not grasped properly, or the pick point is out of reach of the robot. In that case the robot position has to be adjusted and arm motion has to be executed again.

Prior to executing motion of the arm, the target pick point or drop point has to be acquired. This means the package or a drop point has to be localized before each motion. For this the arm state machine uses the previous localization state machine to acquire the marker coordinates of the package or a drop location. Once the coordinates are acquired, they are verified if they are in pick range or drop range. If not, then the arm state machine uses the navigation state machine to adjust the pose of the robot to acquire the feasible range. Then the motion of the arm is planned and executed to reach the pick or drop point. The control flow for pick /drop localization, arm execution, pump activation, and robot motion are managed by specific states given as places in the petri net below. Places (5.0.0) and (6.0.0) are designated for localizing the pick and drop point, places (5.0.2),(5.0.3),(6.0.2) and (6.0.3) are designated for making a request to move the robot and adjust the pose both for pick and drop respectively, and places (5.0.1), (6.0.1), (7.0) and (9.0) are reserved for executing arm motion. A comprehensive detail of the rest of the places i.e. states along with their corresponding transitions is given below.

5.3.3.1.2.1 Petri net

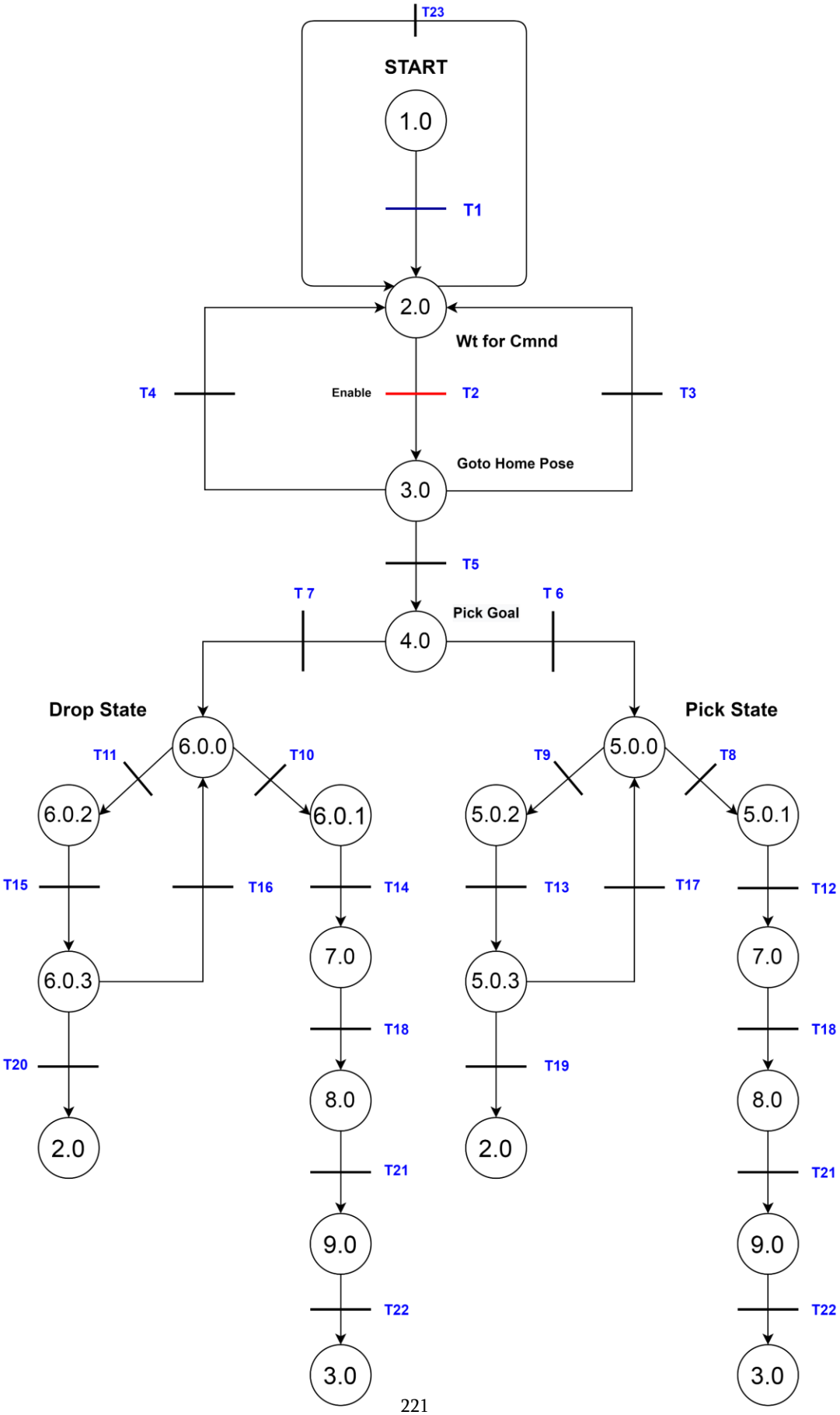


Figure 5.8 : Petri net graph for Manipulation state machine

Places

P 1.0: Start

P 2.0: Wait for command

P 3.0: Goto Home Pose

P 4.0: Pick Goal

P 5.0.0: Extract Coordinates for Pick

P 5.0.1: Execute Pick Motion*

P 5.0.2: Send message move robot

P 5.0.3: Monitor robot move for pick

P 6.0.0: Extract Coordinates for Drop

P 6.0.1: Execute Drop Motion*

P 6.0.2: Send message move robot

P 6.0.3: Monitor robot move for drop

P 7.0: Descend*

P 8.0: Run Pump

P 9.0: Ascend*

*(Execute trajectory)

Transitions

T 1 : Joints read OK

T 2 : Enable = true

T 3 : Pick_done = true

T 4 : Drop_done = true

T 5 : (not(Pick_done) || not (Drop_done)) && enable

T 6 : Pick = true

T 7 : Drop = true

T 8 : Pick Coordinates OK = true

T 9 : Pick Coordinates not OK = true

T 10 : Drop Coordinates OK = true

T 11 : Drop Coordinates not OK = true

T 12 : Pick Traj Execute = true

T 13 : Message mv robot for pick = true

T 14 : Drop Traj Execute = true

T 15 : Message mv robot for drop = true

T 16 : Robot moved for drop = true

T 17 : Robot moved for pick = true

T 18 : Descend = true

T 19 : Robot not moved && time > 3 min

T 20 : Robot not moved && time > 3 min

T 21 : Run pump = true

T 22 : Ascend = true

T 23 : Enable = false

5.3.3.1.2.2 State space

There are 15 places (states) in the petri net and 23 transitions. According to equation (5.2), the incidence matrix '$A = [a_{ij}]$' for the petri net is of the size 15 x 23, the initial marking 'Mo' is '15 x 1' and the vector 'x' is of the size '23 x 1'.

5.3.3.1.2.3 Simulation 1

A simulation corresponding to two scenarios of pick and drop of a package respectively, was performed in 'MATLAB' for verifying the firing of the transitions and correct evolution of the resulting states. The first simulation was carried out for detecting a package and moving the arm to pick. The time for simulation was two minutes. Once the robot aligns itself to the pallet for a pick / drop operation, it calls the arm state machine to commence the process by sending a pick or drop request. Referring to the petri net of figure (5.8), the system after starting waits for a command. As soon as the command is received, transition 'T2' is fired and the state machine brings the arm to the home position i.e. place '3.0'. It then checks the request received in state 'Pick Goal' to either select a pick or drop. Transition 'T6' is fired for this simulation and the system goes to the localization state. In this place (5.0.0) it requests the localization state machine to extract the coordinates of the marker of the target pick point. It takes 5 seconds to extract the coordinates (x, y and z) of the pick point and once they are verified to be in the range, transition 'T8' is fired (indicated by the blue arrow in the figure below), and the state machine executes the motion of the arm to the pick point at height of 20 cm above the package. Then transition 'T12' is fired and the arm executes a descend motion is state '7.0' to place the gripper on top of the package / box properly. Once the gripper is placed firmly, transition 'T18' is fired and the vacuum pump is activated to grasp the box. Then the arm executes an ascend motion to lift the package up and transition 'T22' is fired which brings the arm to home position. The state machine then comes back to state '2.0' (dark green) after 36 seconds to wait for the next command to arrive. The various states of the state machine are shown in different colors in figure (5.9).

5.3.3.1.2.4 Simulation 2

The next simulation was carried out to drop the box picked previously. The execution sequence of states for this scenario is almost same to the previous one, but with the difference that in this scenario it was assumed that the drop coordinates extracted are out of reach of the arm. For this the arm state machine has to go in the reaction mode and request the navigation state machine to move the robot for the specified reach coordinates. This reaction was simulated and the evolution of corresponding states was verified.

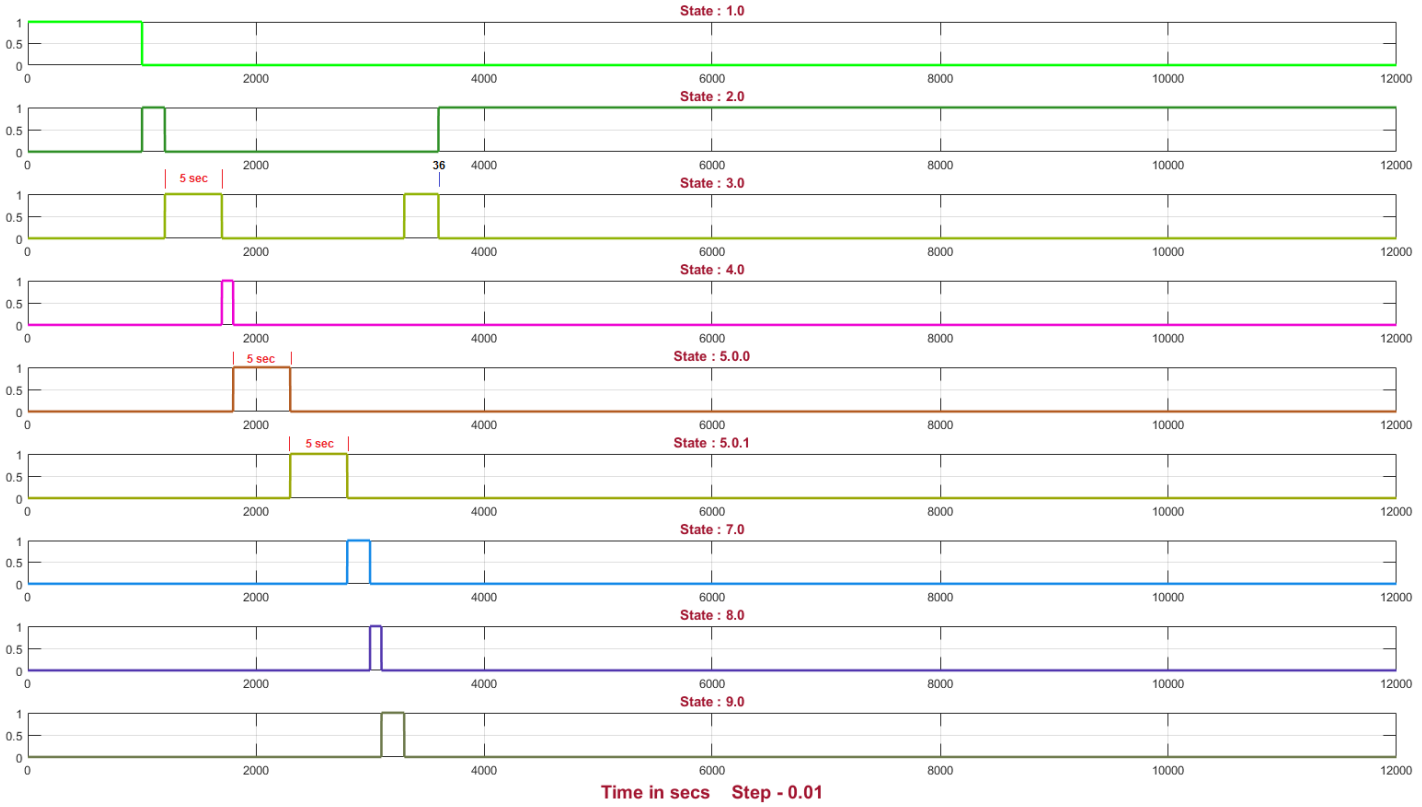


Figure 5.9 : Evolution of states for pick and drop tasks for Arm state machine

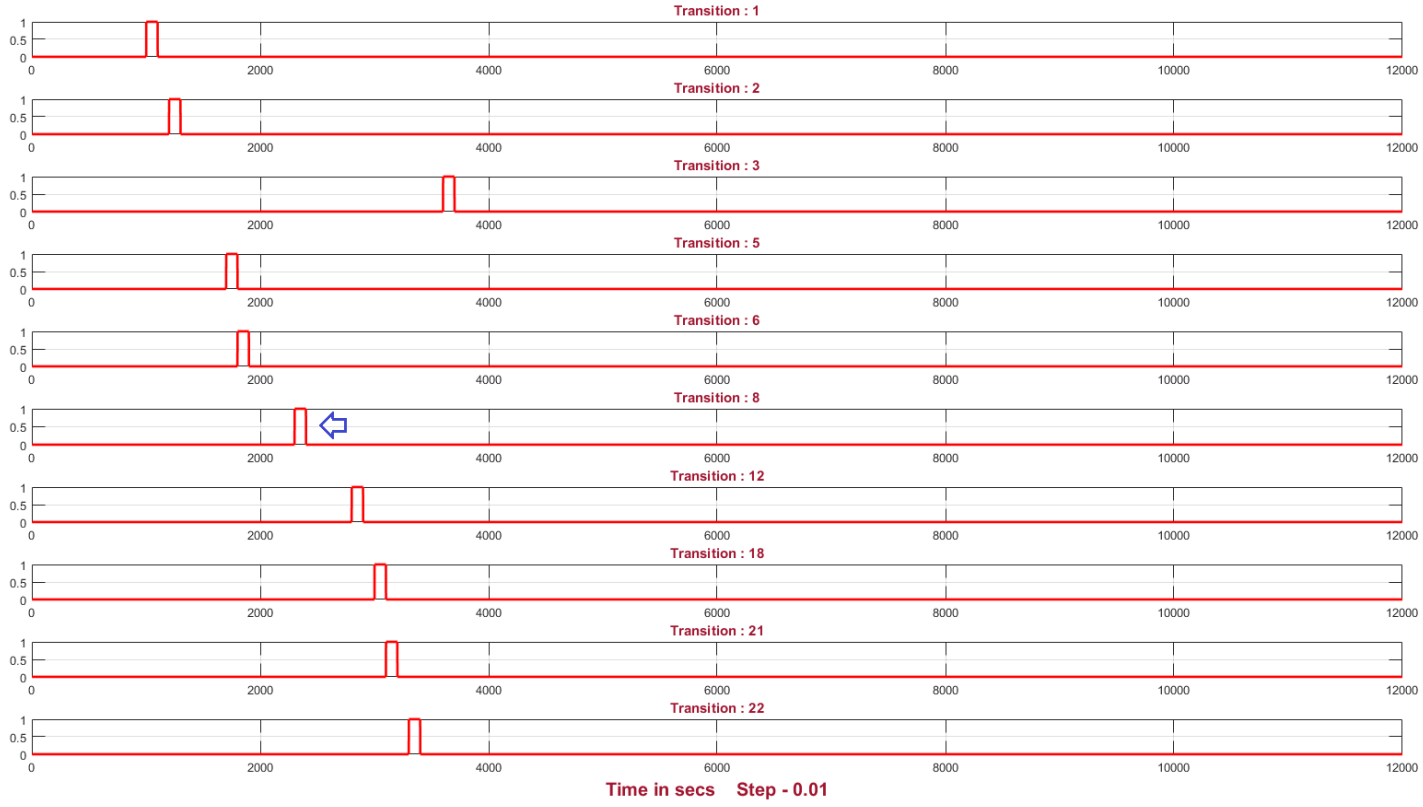


Figure 5.10 : Firing of transitions corresponding to pick and drop process

As before, once the drop item command is received the state machine reaches state ‘6.0.0’ and requests the localization state machine to extract the drop point marker coordinates. The drop coordinates are extracted in 5 seconds and it is assumed that the drop point is a bit further away from the robot, i.e. at the maximum limit of the arm reach. Correspondingly transition ‘T11’ is fired in the petri net, and the arm state machine reaches the state ‘request robot move’. In place ‘6.0.2’ a message is published on a ‘rostopic’ to notify the navigation state machine to move the robot. After the message is published the system enters state ‘6.0.3’ and monitors the move of the robot. This state is an instantiation of the notion of concurrency, since once state is waiting for the other state to finish. When the robot has moved the required distance, a ‘done’ message is published to the arm state machine, and transition ‘T16’ is fired. It can be seen in the state evolution diagram that the system remains in state ‘6.0.3’ for at most one minute until the robot has finished the move. After this the system again goes back to localization state and the marker coordinates are extracted and they are verified to be in range since the robot has already moved closer. The respective next states are then executed to move the arm to the drop point, descend to place the box, deactivate the pump, ascend and then come back to home position hence finishing the drop.

Figure 5.11 : Execution of a concurrent process in state 6.0.3

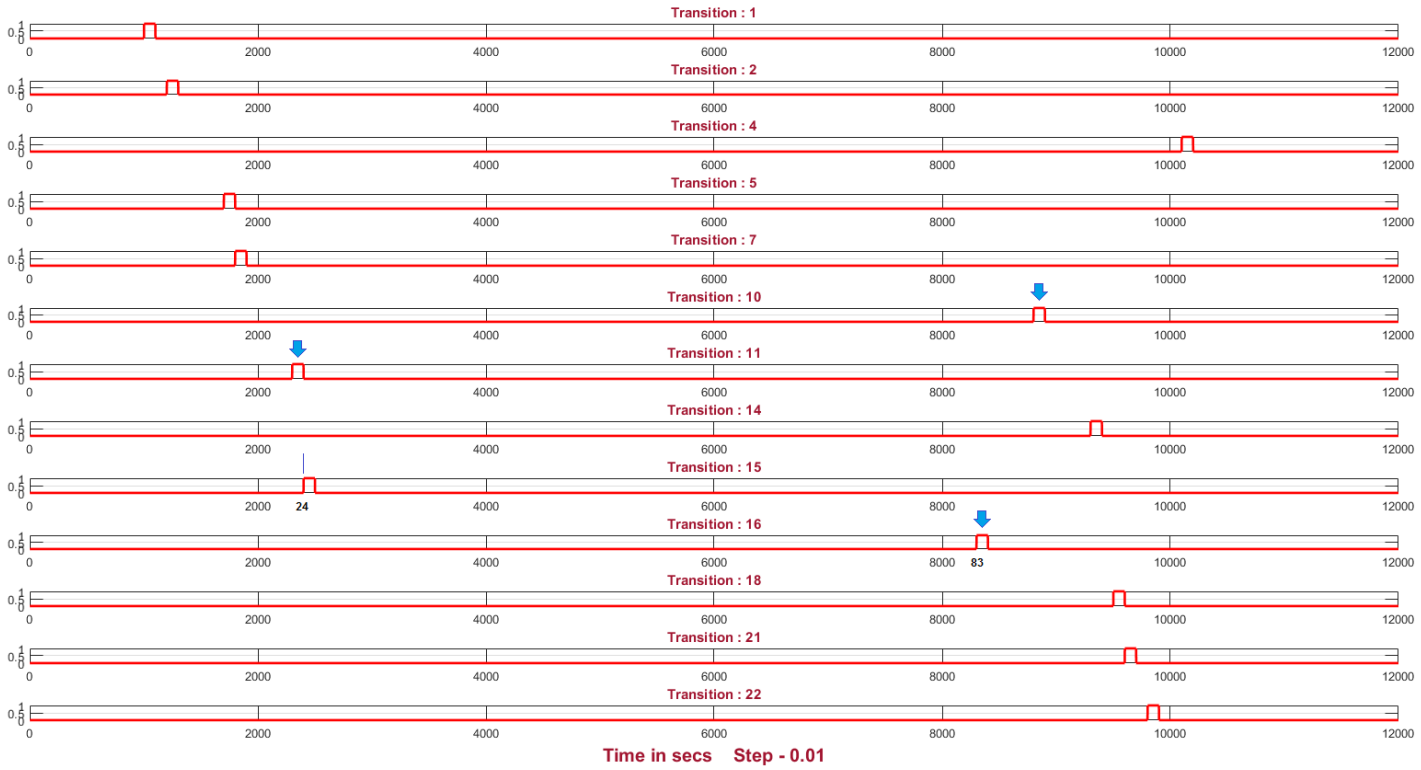


Figure 5.12 : Firing of transitions corresponding to execution and completion of a concurrent process

5.3.3.1.3 Navigation state machine

The objective of the state machine is to manage and control the global behavior of the robot. This is achieved by executing the respective states of the navigation state machine, while at the same time provide service to other state machines to coordinate the behavior activity. In actuality the structure of the picking task required for the enterprise consists of the use of one robot to retrieve the package and place it on another robot to make up a pallet. But this requires a large system to manage the coordination between two robots and their respective behaviors. To adhere to the time constraints and avoid the complexity of coordinating a large system, the state machine given in petri net has been developed to manage the behavior of just one robot and manage a picking task consisting of package retrieval and depositing on pallets. This navigation state machine acts as the global supervisor and manages the picking operation and complete motion of the robot. It localizes the robot in the global map, plans the motion of the robot to the goal point, executes the global navigation and Hough navigation motions, align the robot to a pallet for picking and then dealign when a job is finished. Before commencing a picking operation, the state machine will receive a request from the warehouse management system (WMS) for a picking job. The request will consist of the target point to pick and drop, and the number of items to pick. It will then plan the motion of the robot in the global map, localize and move the robot from a certain start point and bring it to the

5.3.3.1.3.1 Petri net

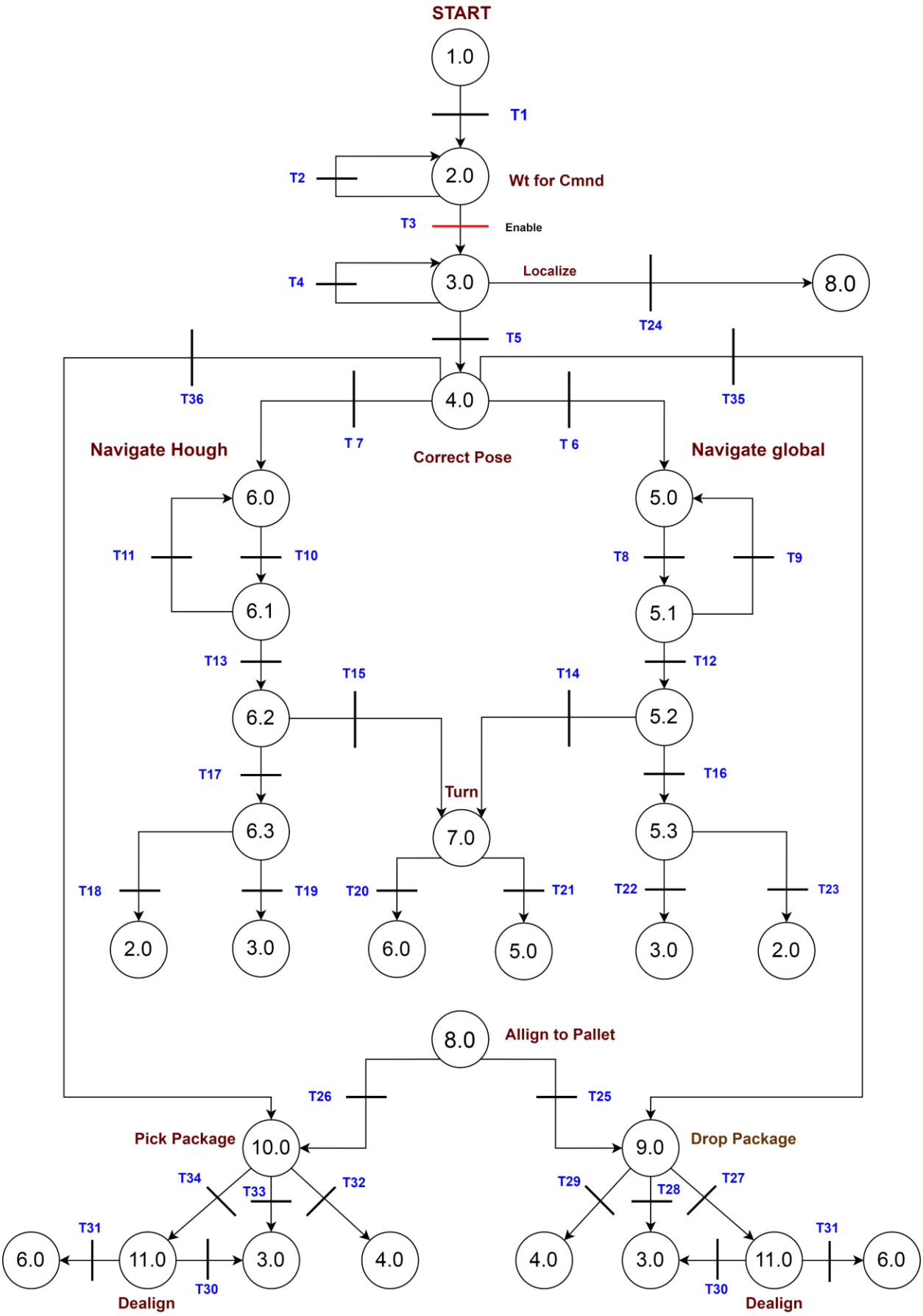


Figure 5.13 : Petri net graph for Navigation state machine

respective rack to the pick location. Then it will start the picking operation and transport the number of items requested to the desired location or drop point. The desired location can be the same rack or another one, but for each move the state machine utilizes the global map and plans the global trajectory inside the map, while at the same time keeping track of the current distance travelled with respect to the odometry of the robot. For managing the motion of the robot and item retrieval / placement, it calls the localization and arm state machines respectively at specific instants when required. At the same time, it also provides a service to arm state machine if it is required to adjust the pose of the robot for pick or drop. This service call is an instantiation of the notion of concurrency.

Places

P 1.0: Start

P 2.0: Wait for command

P 3.0: Localize

P 4.0: Correct pose

P 5.0: Set Goal for Global navigation

P 5.1: Check Goal and Start point

P 5.2: Check Heading

P 5.3: Execute robot motion

P 6.0: Set Goal for Hough navigation

P 6.1: Check Goal and Start point

P 6.2: Check Heading

P 6.3: Execute robot motion

P 7.0: Turn

P 8.0: Align to pallet

P 9.0: Drop package

P 10.0: Pick package

P 11.0: Dealign from pallet

Transitions

T 1 : start OK

T 2 : Enable = false

T 3 : Enable = true && job_cntr < no_jobs

T 4 : Marker_detect = false

T 5 : Marker_detect = true && locate_rack = true

T 6 : Pose_ok = true && move_global = true

T 7 : Pose_ok = true && move_hough = true

T 8 : Goal_point && start_point set = true

T 9 : Goal_ok = false && start_point_ok = false

T 10 : Goal_point && start_point set = true

T 11 : Goal_ok = false && start_point_ok = false

T 12 : Goal_ok = true && start_point_ok = true

T 13 : Goal_ok = true && start_point_ok = true

T 14 : Heading_ok = false

T 15 : Heading_ok = false

T 16 : Heading_ok = true

T 17 : Heading_ok = true

T 18 : Goal_reached = false

T 19 : Goal_reached = true

T 20 : Turn = true && move_hough = true

T 21 : Turn = true && move_global = true

T 22 : Goal_reached = true

T 23 : Goal_reached = true & goto_home

T 24 : Marker_detect = true && locate_pallet = true

T 25 : Aligned = true && drop = true

T 26 : Aligned = true && pick = true

T 27 : Item_droped = true

T 28 : Item_droped = false

T 29 : Item_droped = false && move_robot = true

T 30 : Dealign = true && job_cntr > no_jobs

T 31 : Dealign = true && job_cntr < no_jobs

T 32 : Item_picked = true

T 33 : Item_picked = false

T 34 : Item_picked = false && move_robot = true

T 35 : Pose_ok = true && drop_again = true

T 36 : Pose_ok = true && pick_again = true

In the above description of the petri net the places '3.0', '4.0', '5.0', '6.0', '7.0', '8.0' and '11.0' are designated for the states that control the localization and motion of the robot, while states '9.0' and '10.0' are reserved for making a request to the arm state machine. Before the start of navigation, the robot has to localize itself in the global map. Once the pose is acquired, then using it as the start point the state machine calls the path planner to produce a path to the goal point. Then using the controller in section (4.15-4.16), the navigation is executed by the significant waypoints of the global path. Once arrived at the location, the arrival (respective rack) is again verified by localization. The Hough controller (4.33-4.35) is invoked to execute the motion inside the rack to go to the pick point and once arrived at the pick point, again verification is done by localizing the pallet. After retrieving the package, the robot disengages (dealing) from the pallet and the path planner is utilized to plan the path again to the drop point. If the drop point is close in the same rack then the Hough motion is executed to reach the target, otherwise the controller in section (4.15-4.16) is executed to go to the other rack to reach the drop point.

5.3.3.1.3.2 State space

There are 17 places (states) in the petri net and 36 transitions. According to equation (5.2), the incidence matrix $'A = [a_{ij}]'$ for the petri net is of the size 17 x 36, the initial marking 'Mo' is '17 x 1' and the vector 'x' is of the size '36 x 1'.

5.3.3.1.3.3 Simulation

A simulation was performed for a complete picking operation for one job in MATLAB, to verify the firing of the transitions and correct evolution of the resulting states. Referring to the petri net of figure (5.13), the system after starting waits for a command. As soon as the command is received for a navigation goal, transition 'T3' is fired and the system proceeds to localize the robot. Transition 'T3' is fired on an external command. Once the robot is localized

it corrects it's position in the global map in state ‘4.0’. Since the robot has to navigate globally to the pick location, transition ‘T6’ is fired and system proceeds to state ‘5.0’. The path planner is called, the start point is set and in the following states, robot heading is verified and motion is executed. Once the robot reaches the goal point transition ‘T22’ is fired and the system again localizes the robot to verify the arrival point. The robot corrects its pose i.e state ‘4.0’, and then it proceeds to execute the Hough navigation motion. The system enters state ‘6.0’, the start point and goal point (pick point) is set and the heading is verified. The heading is verified in state ‘6.2’ to check if the robot is in the good direction to go with respect to the global map. Here it is assumed that the robot at the moment is in the opposite direction to the required motion to execute, so the system proceeds to state ‘7.0’ and the robot is turned 180°. After turning the system again goes back to state ‘6.0’ to assign the start and goal and the heading is verified. Then the robot executes Hough motion in state ‘6.3’. Once it reaches the estimated goal point, the system goes to state ‘3.0’ to localize a pallet to verify the correct arrival. After detection of the pallet, the state is ‘8.0’ and the robot aligns itself to the pallet. In state ‘10.0’ the state machine requests the service of the ‘Arm’ state machine to pick the package. The system remains in this state as long as the other state machine has not finished the pick process. As soon as the ‘Arm’ state machine finishes, it publishes the message on a ROS topic and the system goes out of state ‘10.0’ and proceeds to state ‘6.0’ to navigate to the drop point. Using the Hough motion, the robot arrives at the estimated desired location, and again localization is invoked to detect the pallet for drop.

Once the pallet is detected the system again goes into state ‘8.0’ to align to pallet. Then the ‘Arm’ service is utilized the second time in state ‘9.0’ to place the package at the desired drop point. After finishing the system goes to state ‘11.0’ and the robot dealings from the pallet. At this point one job of pick and drop has completed. Now the system again goes to localization state so that the robot can correct its position in the global map for its journey back to home position. The home position for this simulation is the starting point when the command to execute was received by firing transition ‘T3’. It has to be noted that the system after finishing a job, will invoke the path planner to plan the global path to go to the next point. The next point can be a home position or any other pick point in another rack, and the system will repeat the process. The evolution of the states is numbered in the figures below to show the sequence of states followed for a complete pick operation as explained above. The numbering can be matched to the states of petri net in figure (5.13) to visualize the complete sequence of state execution to realize the complete operation. The time of the simulation was specified to be 7 minutes approximately, while assuming ideal conditions of every process that the robot completes.

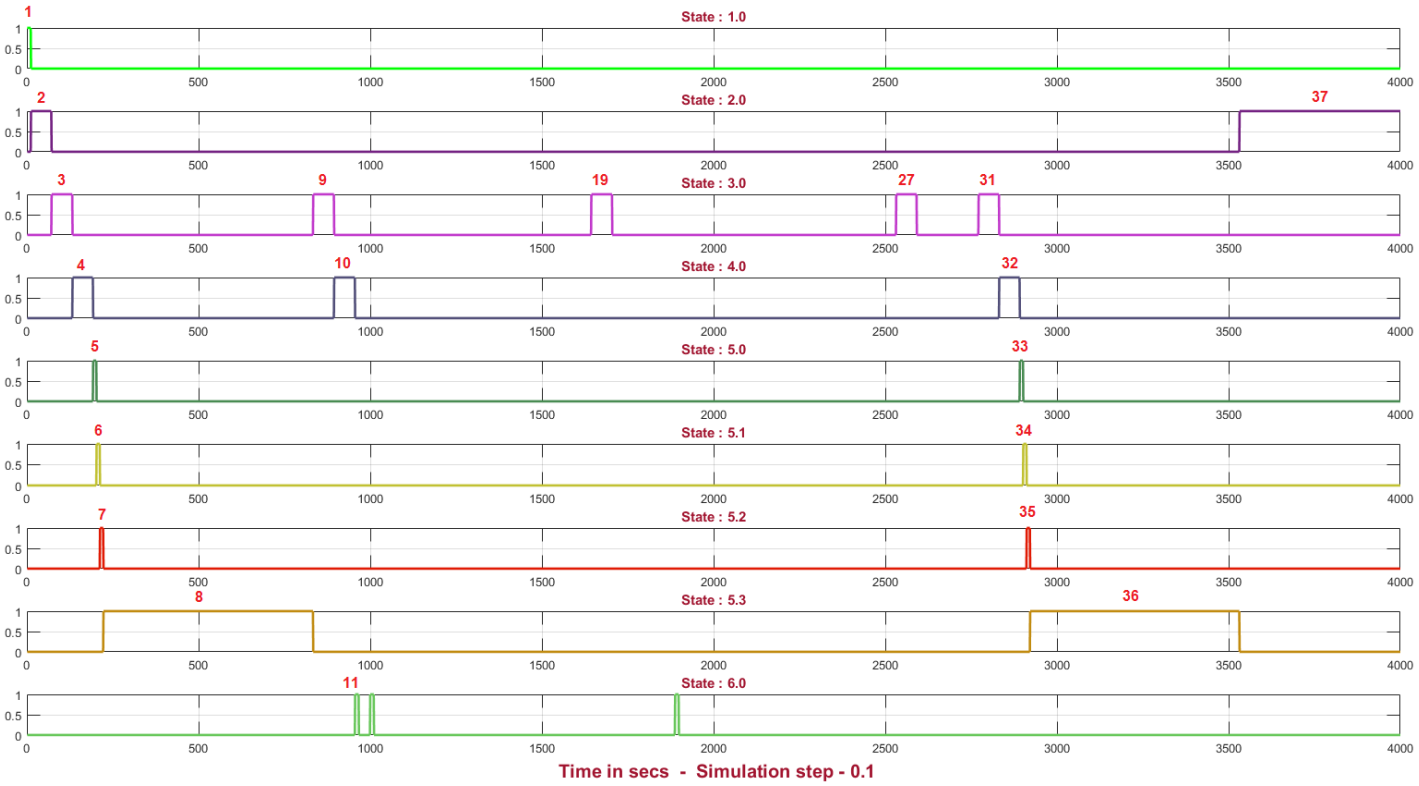


Figure 5.14 : Evolution of states for a complete picking process

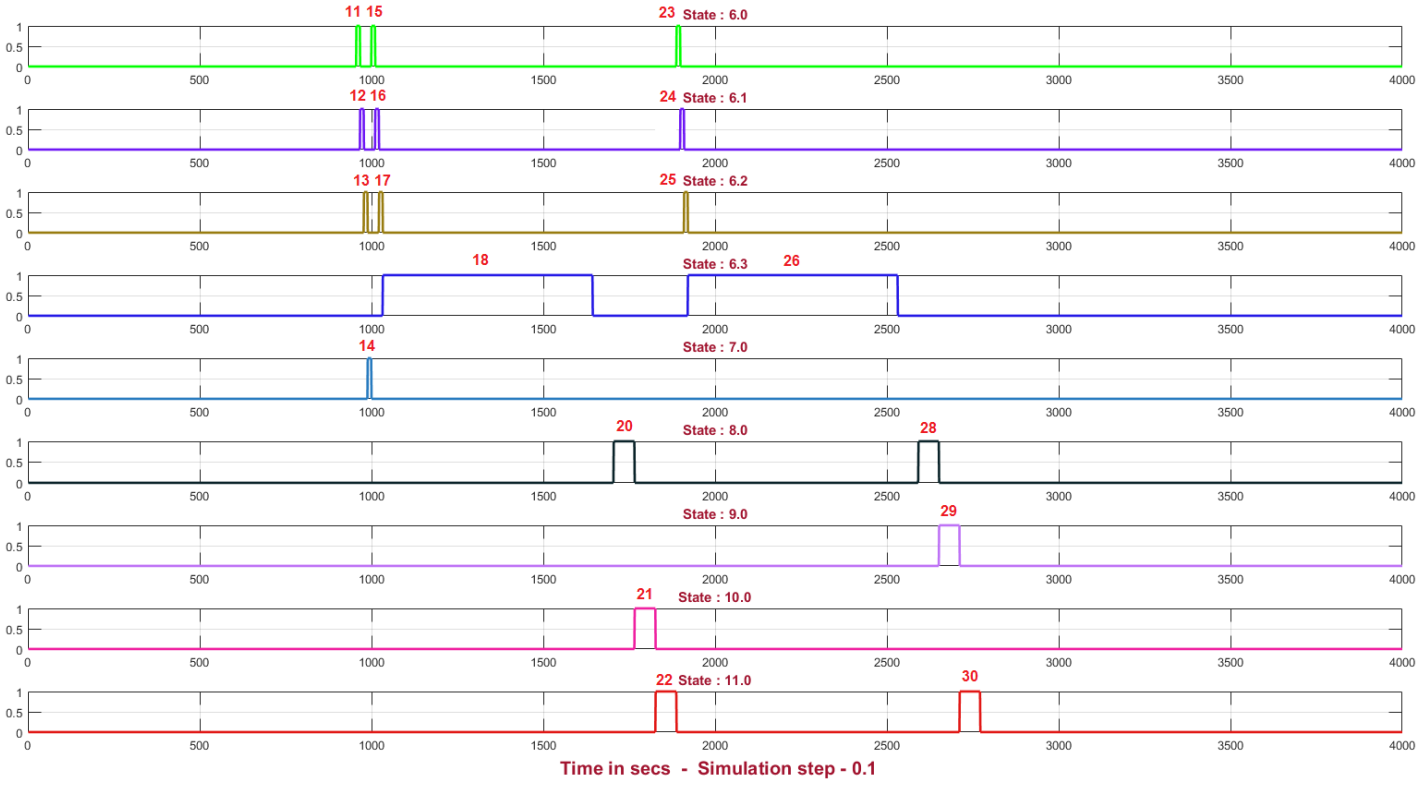


Figure 5.15 : Evolution of states for a complete picking process

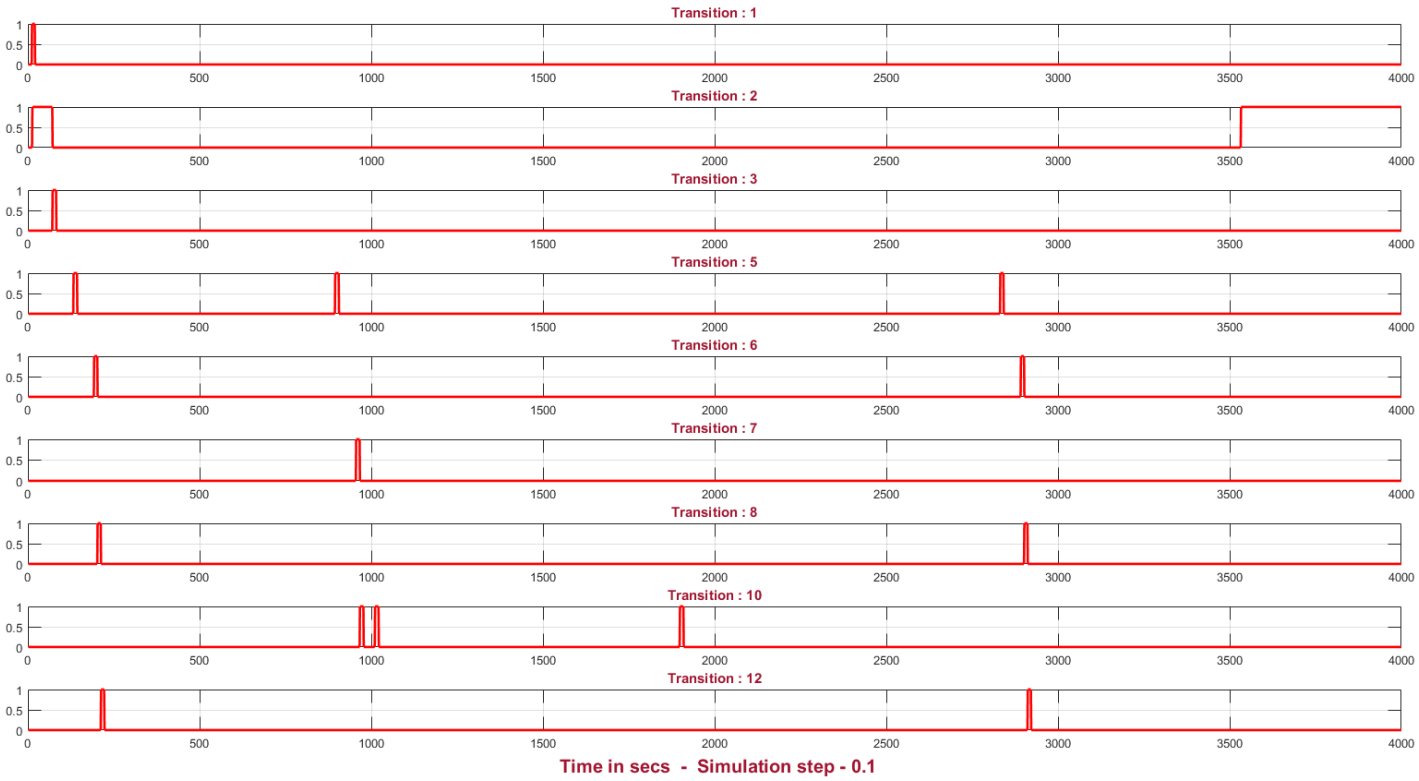


Figure 5.16 : Firing of transitions for a complete picking process

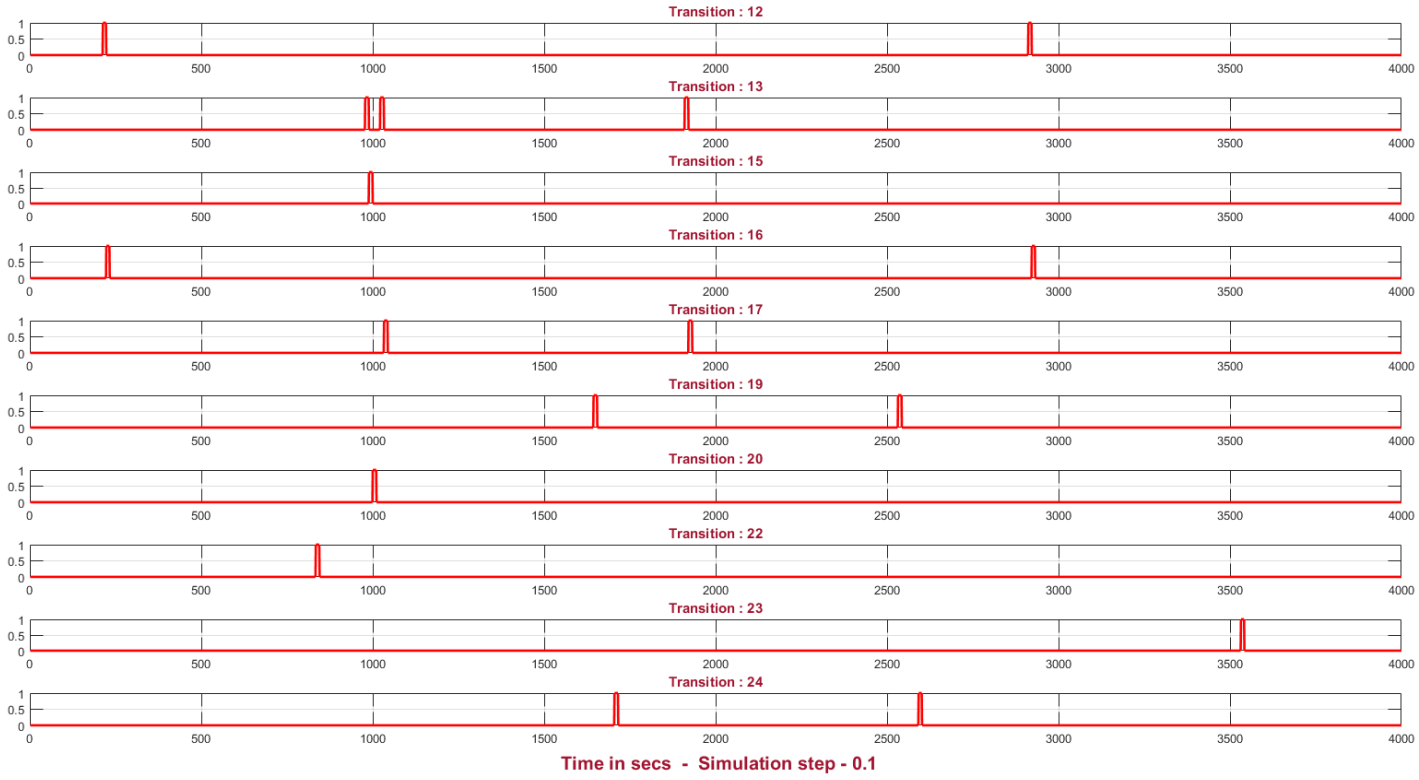


Figure 5.17 : Firing of transitions for a complete picking process

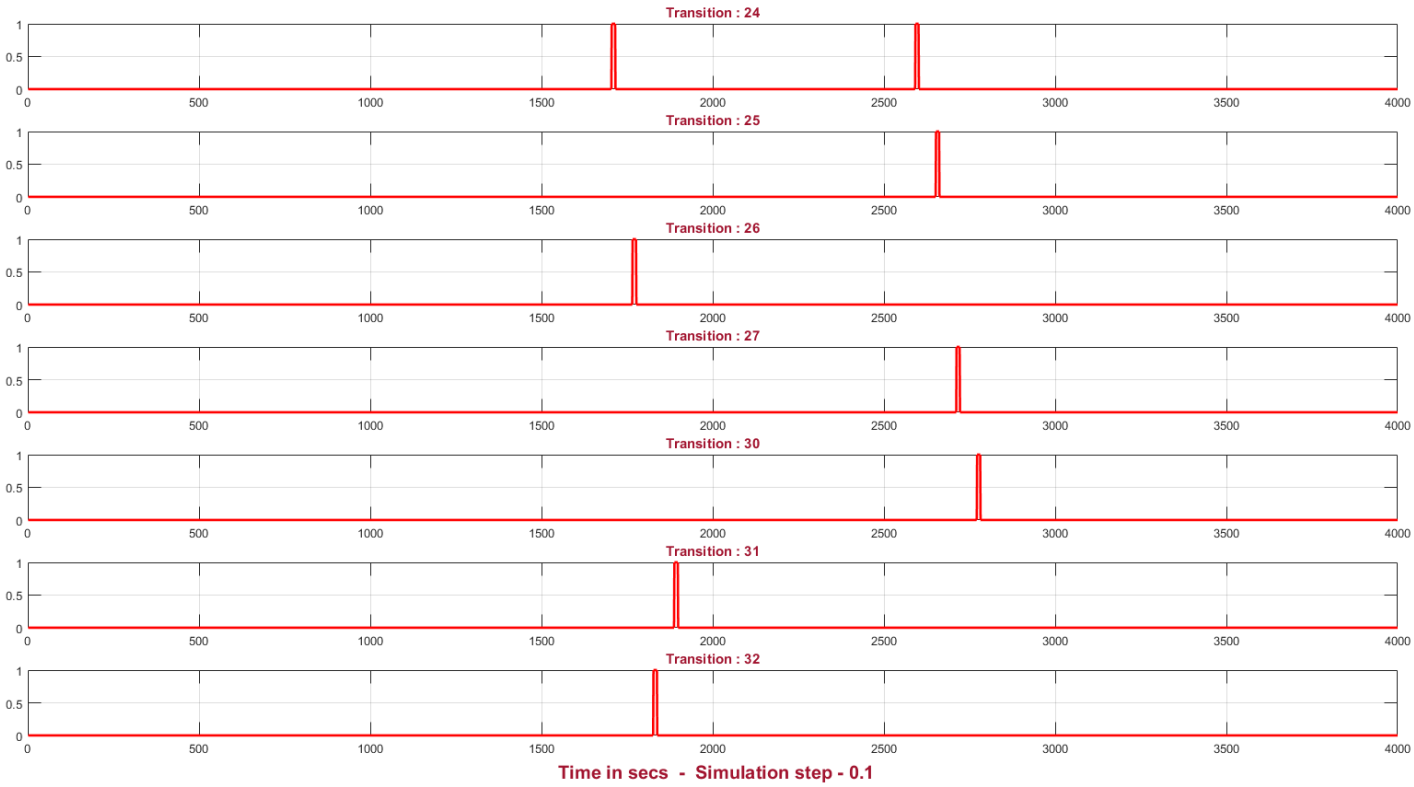


Figure 5.18 : Firing of transitions for a complete picking process

5.3.3.2 Real time testing performance

After validating the performance of each petri net independently in simulation, a real time test was performed to validate the performance of the three state machines executing in unison as a global system. The objective was to identify deadlocks and verify concurrency in the form of successful message passing, along with identification of blind spots where more states or conditions might be required to resolve a conflict. The test was performed to execute a full picking scenario and complete 2 jobs. The test environment was setup as a section of warehouse consisting of two aisles with one containing a rack with pallets as seen in figure. Markers were placed both on the racks and the boxes to facilitate localization during operation. Environment layout can be seen in figure (5.19) in which the process stages are numbered to indicate the sequence of task progression. The robot starts (1) from a point in the in first aisle and navigates using the path planner to arrive at the goal point (2) in the next aisle in front of the rack. It then proceeds to move inside the rack using Hough controller and detect the pick pallet. Once it detects the correct pallet, it aligns (3) itself to it and picks (5) up the box and dealigns itself to go to the drop location. After travelling an approximate distance using the path planner and odometry, it again localizes the drop pallet and aligns itself to drop the first box (6).

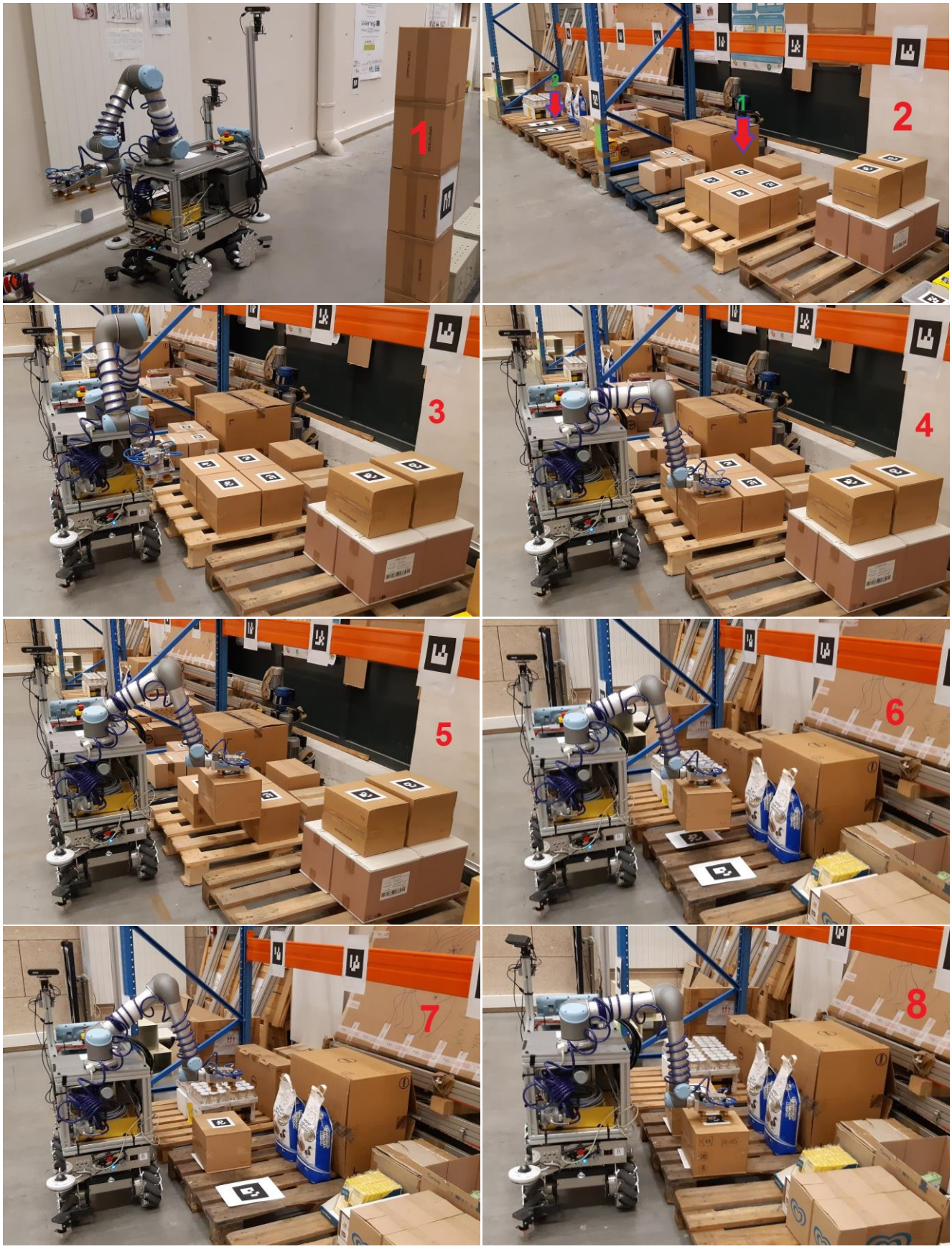


Figure 5.19 : Execution sequence of a picking task in a warehouse

As it detects the drop location, it evaluates the drop point to be outside the reach of the arm. Therefore the 'Arm' state machine invokes states (5.0.2-5.0.3) to request the 'Navigation' state machine to move the robot a bit forward. Once this is executed, the box is dropped on the right spot and the robot dealings itself from the pallet thus finishing job '1' (7). It then turns and navigates back to the same pick spot (pallet) to retrieve the second package (box). The same sequence of align, pick and then dealign is repeated and the robot transports the second box to the previous drop location. Once the drop is finished (8), job number '2' is over and the robot localizes itself to go back to the home position or the start point in the first aisle. On successful localization the robot starts its journey back to home and successfully arrives at the starting point. For a successful execution of the process step by step, the evolution of states during execution can be viewed in the "**video**[ref] " for more detail.

It is imperative to mention here that the global supervisory system consisting of the three state machines made use of the "client-server" architecture in ROS. Each state machine is designed as a service which can be called by other state machines in a sub-state to execute a concurrent process in the called state machine. The message passing and validation is acquired with ROS topics, with all the state machines running as parallel nodes in the same ROS core environment.

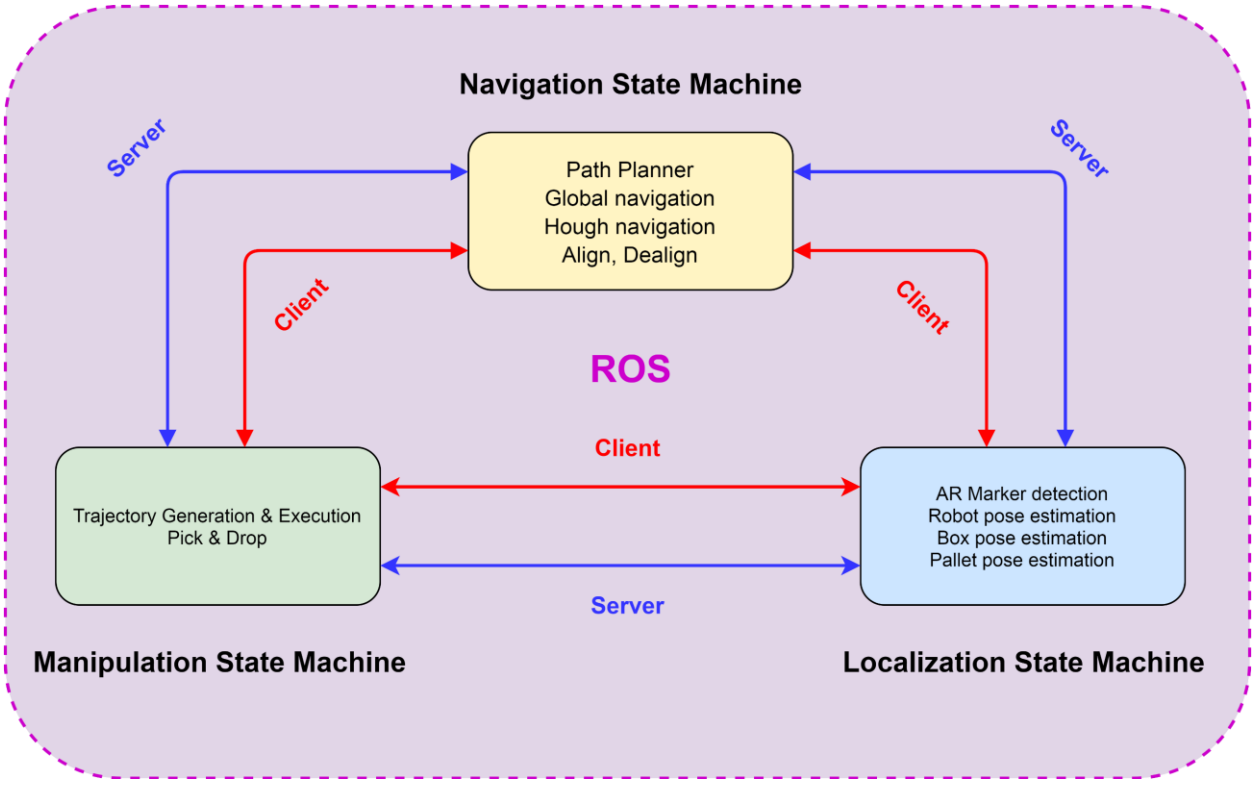


Figure 5.20 : Architecture of the Global Supervisory System

25 ref : https://youtu.be/bwvEnpOTHj4

5.4 Future strategy

The global framework presented in the previous sections has successfully demonstrated a well-structured and organized approach to control the behavior of the robot. The framework facilitates analysis and incremental improvement by adding a state for a conflict resolution, or by identifying how a certain behavior or execution can be improved with the addition of one or more states. Hence it is evolutive. Even though the framework has been demonstrated successfully, however one of the issues with the current implementation is that each state machine has to be manually coded. It would be more advantageous to reduce the labor of hand coding and make the system intelligent to develop a state machine on its own. This can be achieved by developing a system which utilizes a library of individual states and utilize machine learning.

First a library of states pertaining to numerous behaviors can be developed. Then the system can be trained for the selection of states for one or multiple scenarios. Once the system gets trained, it can develop its own state machines and produce a network of global supervisory system. Taking this idea and putting it in the context of current system at hand, the 'Arm' state machine can be made intelligent to detect a product and infer which states to choose to pick a specific type. The system should be able to detect and differentiate the type of products and the plan accordingly (select states). The selection of different states will then govern the motion planning of the manipulator on how the product is picked, e.g. if a certain type and size of package is encountered, the system should be able to decide the feasibility of grasping the box from the side or the top. The concept is given in figure below.

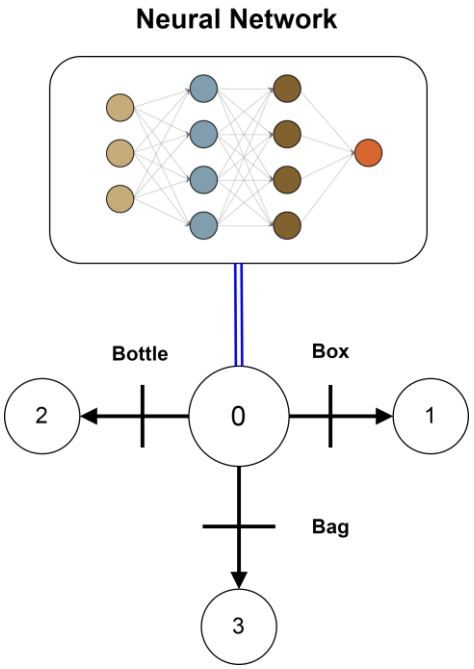


Figure 5.21 : Proposed concept of an intelligent system

A similar kind of approach has already been developed and tested in our research facility for a different project. The work in [174] is focused on adaptive grasping involving object detection and motion planning based on a specified grasping mode. The grasping mode consists of adapting the fingers of the gripper to the exposed face. The framework used in the approach consist of a deep neural network 'Yolo' (You look only once). Yolo [175] is an object detector which works by detecting each object in an image. A neural network is applied to the image which divides it into regions, to predict bounding boxes and probabilities for each region. The bounding boxes are weighted by the predicted probabilities. In the presented work 'Yolo' is trained separately for object detection and object face classification. In each training a different data set is provided. For pose estimation a data set consisting of the shape of the object with various configurations, and for face classification the data set of all the faces of the object is given. Once the training is complete the system after detecting an object is able to estimate the pose while at the same time infer the face of the object to decide a grasping mode. All this information is then fed to the motion planner to produce a grasping trajectory.

The same framework can be used for detecting different type of products e.g. boxes, bags, bottles etc. While the number and type of products in a warehouse is very large, the idea is to first select a very limited number of products and then train the system first for at most two or three boxes / packages. The training requires data set and similar to the previous approach two type of data sets can be prepared. The first type will consist of the shape of the boxes in various configurations, while the second data set will contain all the faces for face classification. Once the system is trained, it should be able to detect different types of boxes based on face classification, and then estimate their poses to decide on a grasping strategy for motion planning. The proposed strategy is given in the figure below.

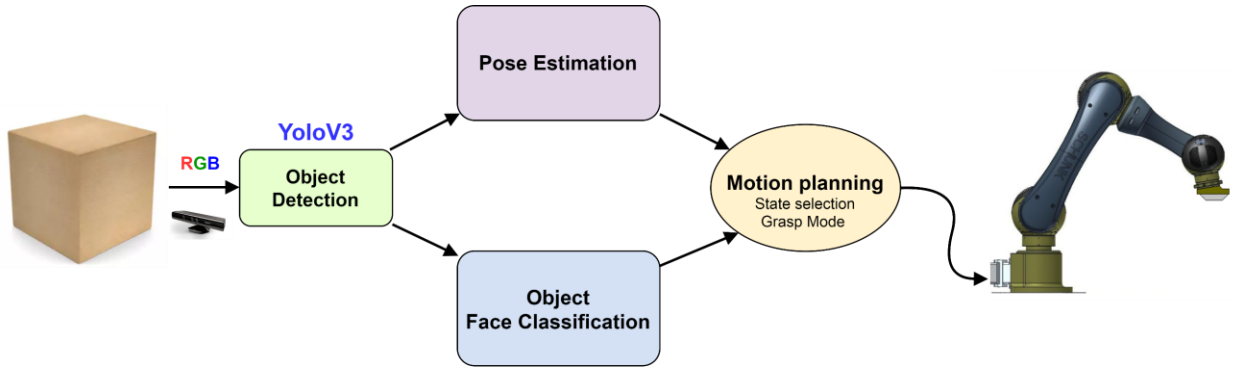


Figure 5.22 : Proposed deep learning framework

Chapter 6

Conclusion and Perspective

6.1 Conclusion

The goal of this research project was to achieve autonomous pallet production. The research conducted in this context helped to evaluate and identify the limitations and key components involved in the accomplishment of such an endeavor. The objective was to use a mobile manipulator and perform picking inside a warehouse. To fulfill this goal, the fundamental elements involved in realizing this concept were identified and a state of the art was prepared. Mobile manipulation in warehouses involves navigation, localization, manipulation and grasping. The state of the art concluded by studying various grippers and grasping strategies, motion planning for navigation, indoor localization techniques, cobotic arms for collaboration, and behavior design strategies for global task management.

A prototype consisting of a mobile base and a collaborative arm was developed by integrating existing technologies. Existing grasping technologies were studied and vacuum gripping was employed with the development of a custom gripper and installation of a vacuum system on the prototype. Each brick of the concept was tackled sequentially, and first a navigation strategy was developed for moving inside the racks, and then for point to point motion inside the warehouse. It was required for the robot to come to the pick point from a random location inside the warehouse. This was achieved by the use of a global path planner and a motion controller that enabled the robot to reach the goal from any start point. For motion inside the racks, a corridor following strategy was employed. For this the conventional Hough transform was used to develop a motion controller to align the robot to the racks and execute its motion. The controllers in both cases were tested and demonstrated in a real industrial setting with successful navigation.

Due to the extremity of dynamic setting of a warehouse environment, SLAM was not pursued in this thesis and a simple strategy was adopted for localization. Synthetic markers were used in the environment and also on the products, and a readily available library in ROS was used for their detection. The framework provided localization with good accuracy enabling the robot to localize itself in the environment and detect the products to pick successfully in every execution. For manipulation a custom motion planner was developed and

tested successfully for pick and place. The motion planner consisted of planning and executing point to point trajectories consisting of linear and circular moves in the available workspace.

To demonstrate a complete picking process, it was required to use a framework that could combine all the essential elements and execute them in a coordinated way to achieve the task. Hence the task execution or behavior control of the robot was modelled with discrete event systems approach and Petri nets were used to create state machines for each component required for the global task. The execution of state machines was evaluated in simulation and the full framework of task management was tested in real time, by demonstrating a complete picking task in an industrial warehouse. The task was carried out successfully validating the use of the employed strategies and the final framework

For the enterprise FM logistic, this research endeavor was a first step towards autonomous palletization and its further evolution. The added value that the company has acquired is the significant and fundamental knowledgebase, a successful proof of concept as a working prototype, identification of limitations and constraints, analysis of difficulties, and accomplishment of the objective by the technological demonstrator executing a subtask of the major goal. The comprehensive knowledge base provides an introspect of what already exists and how it can be leveraged to produce something new, conceiving the essential bricks of developing a framework conforming to the objective at hand. The development of the proof of concept and its implementation highlighted the limitations of the methods employed, constraints of hardware and environment, difficulties in implementation and the room for improvement. The deliverables at the end provided an assessment of the extent of enhancement required in the current framework to make it more efficient for deployment on a commercial scale in time to come.

One of the key objectives of the company in this research was to assess if the goal can be achieved or not and if so, then to what extent and how. If the concept is feasible then what are the essential bricks to develop a framework to achieve this goal. The type of solution that can be conceived to evaluate the timelines, return of investment, and its commercial utilization in the future. The goal of this project was not to deliver a final product, but propose and work on a proof of concept that can be enhanced based on the results acquired and lessons learnt in the research. This research experience has provided a comprehensive overview of how much effort is still required to reach the apex and deliver an efficient commercial solution to the market. Further, it has established the foundation for FM Logistic to conduct further endeavors of research and engineering for the evolution of autonomous palletization.

6.2 Future Perspective

Today logistics is a very promising industry for the future of robotics and automation technology. A lot of effort and research is being dedicated in this domain for the improvement of operations to increase the productivity. The goal is to make most of the processes autonomous to reduce fatigue, labor costs and increase production by collaborative technology. Companies ranging from small enterprises to well renowned ones such as Boston Dynamics[26] are investing a considerable amount of resource and capital for the improvement and autonomy of logistic processes. One of the core operations in logistics is order picking and palletization and the key focus of the robotic research community is to achieve autonomous palletization. The reason is twofold, i.e. reduce human fatigue and increase productivity. The objective is to have a shared environment in which robots need not replace the workers but rather collaborate with them to augment the production. This requires the robots to work at some satisfactory level of decision making to execute tasks autonomously for a specific operation. As a consequence, robots having the situational awareness and capability for mobile manipulation is the proposed strategy for autonomous palletization.

This research endeavor has led to the conclusion that mobile manipulation in itself and autonomous palletization as a whole, is an arduous task and there is ample room for improvement in every aspect, until a commercial solution is fully realizable. For the task of autonomous palletization, a robust navigation requires to take into account all the aspects of a dynamic environment. The motion strategies need to be flexible, incorporating all kinds of motions required globally or locally with obstacle detection and avoidance. SLAM in dynamic industrial environments is still an open problem, and for the moment localization is resolved by augmenting the environment with artificial landmarks (markers), but it does not solve the problem of blind spots which degrades the long-term localization and autonomy. Both navigation and localization should be augmented with a framework to make the robot fully aware and cognizant of its environment for a long-term autonomy. Also, for the global task management of the task of autonomous palletization, a fleet management framework is required which is able to control and coordinate all the mobile manipulators for the successful operation of picking. This framework can function as a standalone supervisor, or can integrate with the warehouse management system as a submodule.

The contribution in this thesis on the navigation aspect was primarily focused on moving inside the racks. The motion inside the racks was realized as a corridor following problem and Hough transform was utilized in the control to track the reference of virtual walls of the corridor or the rack. To reduce the computational time for the real time control, a filtration process was employed making use of a limit threshold to extract the optimal control

26 https://youtu.be/5iV_hB08Uns

references or Hough lines. In the current implementation, to select the control references or the lines, this limit is fixed manually. To account for noise, sparse or excess data the current framework using Hough transform is required to be adaptive, for automatically tuning the efficiency of control for optimality, based on the available data. Thus, the 'limit' thresholding needs to be adaptive to account for the variation of data available, to ultimately produce the control references.

Currently in the industrial sector vacuum grasping is delivering a major chunk of application thus meeting the production goals, but it is still not able to handle products with large sizes, irregular shapes and inconsistent surfaces. This requires the development of a versatile gripper which can grasp almost any product with variation of shape, size and irregular surfaces. For handling the object (package), its manipulation requires a motion planner which takes into account the constraints of speed, collision avoidance and safe trajectories guaranteeing precise displacement of products. Therefore, a robust motion planning framework for manipulation should leverage the coupled modeling and control of the mobile manipulator, to ensure the dynamic stability of the platform as well as the dynamic stability of the product being grasped.

The detection and correct inference of a product on a pallet is still an open challenge. A homogenous pallet consists of the same type of products, while heterogenous pallets have different ones. The detection requires the correct inference of the type of product in presence of similar or different products, and also its pose within the whole stack. For the moment this is solved by placing a marker on the product, which is encoded with specific data giving out the type and its pose on detection. A marker free true detection can only be acquired by the application of advanced image processing framework.

To manage the execution of a complicated task such as autonomous picking, a task management or behavioral engine is required which is able to coordinate the whole activity without any deadlocks. At the moment such frameworks are manually coded with states for executing specific behaviors. It is more advantageous to develop a framework which is able to produce a chain of behaviors on its own corresponding to a specific scenario, and is able to learn and behave the same way a human picker performs decision making. To acquire autonomy, while the current focus is on the development of primitive control and motion planning strategies, a significant and primary area of contribution is the development of a behavior management framework which can have the capacity and capability to perform at the same level of reasoning and decision making as a human operator.

The power of human decision making and reasoning is the ability to correct itself and learn. The decision-making cycle utilizes past experience, error correction and learning to adapt in the next cycle. Higher level robust decision making can only be acquired by imitating and implementing human form cognitive models and learning methods. For the moment machine learning is the answer to develop and train a system, that may behave relative to a human level of decision making and achieve reasonable results. But to have the exact power or similar kind of decision-making ability, the answer is to develop and use a parallelism of human cognitive psychology[27] and an understanding of the processes to execute. A thorough understanding of the processes to perform and the required behaviors to execute with a robust and reactive decision-making paradigm is a research challenge for the future generations to come.

27 Wiki: https://en.wikipedia.org/wiki/Cognitive_psychology

References

[1] De Koster, R., Roodbergen, K.J. and Van Voorden, R., 1999. Reduction of walking time in the distribution center of De Bijenkorf. In New trends in distribution logistics (pp. 215-234). Springer, Berlin, Heidelberg.

[2] Calzavara, M., Glock, C.H., Grosse, E.H., Persona, A. and Sgarbossa, F., 2017. Analysis of economic and ergonomic performance measures of different rack layouts in an order picking warehouse. Computers & Industrial Engineering, 111, pp.527-536.

[3] Hvilshøj, M., Bøgh, S., Skov Nielsen, O. and Madsen, O., 2012. Autonomous industrial mobile manipulation (AIMM): past, present and future. *Industrial Robot: An International Journal*, *39*(2), pp.120-135.

[4] Hvilshøj, M. and Bøgh, S., 2011. "Little Helper"—An Autonomous Industrial Mobile Manipulator Concept. International Journal of Advanced Robotic Systems, 8(2), p.15.

[5] Dubowsky, S. and Vance, E.E., 1989, May. Planning mobile manipulator motions considering vehicle dynamic stability constraints. In Proceedings, 1989 International Conference on Robotics and Automation (pp. 1271-1276). IEEE.

[6] Dubowsky, S., Gu, P.Y. and Deck, J.F., 1991, August. The dynamic analysis of flexibility in mobile robotic manipulator systems. In VIII World Congress on the Theory of Machines and Mechanics, Prague, Czechoslavakia, Aug (pp. 26-31).

[7] Pin, F.G. and Culioli, J.C., 1992. Optimal positioning of combined mobile platform-manipulator systems for material handling tasks. Journal of intelligent and Robotic Systems, 6(2-3), pp.165-182.

[8] Huang, Q., Sugano, S. and Kato, I., 1994, September. Stability control for a mobile manipulator using a potential method. In Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94) (Vol. 2, pp. 839-846). IEEE.

[9] Pin, F.G., Culioli, J.C. and Reister, D.B., 1994. Using minimax approaches to plan optimal task commutation configurations for combined mobile platform-manipulator systems. IEEE Transactions on Robotics and Automation, 10(1), pp.44-54.

[10] Yamamoto, Y. and Yun, X., 1992, December. Coordinating locomotion and manipulation of a mobile manipulator. In [1992] Proceedings of the 31st IEEE Conference on Decision and Control (pp. 2643-2648). IEEE.

[11] Yamamoto, Y. and Yun, X., 1995, May. Coordinated obstacle avoidance of a mobile manipulator. In Proceedings of 1995 IEEE International Conference on Robotics and Automation (Vol. 3, pp. 2255-2260). IEEE.

[12] Nagatani, K. and Yuta, S., 1996, April. Designing strategy and implementation of mobile manipulator control system for opening door. In Proceedings of IEEE International Conference on Robotics and Automation (Vol. 3, pp. 2828-2834). IEEE.

[13] Khatib, O., Yokoi, K., Chang, K., Ruspini, D., Holmberg, R. and Casal, A., 1996, November. Vehicle/arm coordination and multiple mobile manipulator decentralized cooperation. In Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS'96 (Vol. 2, pp. 546-553). IEEE.

[14] Papadopoulos, E.G. and Rey, D.A., 1996, April. A new measure of tipover stability margin for mobile manipulators. In Proceedings of IEEE International Conference on Robotics and Automation (Vol. 4, pp. 3111-3116). IEEE.

[15] Lee, J.K. and Cho, H.S., 1997. Mobile manipulator motion planning for multiple tasks using global optimization approach. Journal of Intelligent and Robotic Systems, 18(2), pp.169-190.

[16] Osumi, H., Terasawa, M. and Nojiri, H., 1998, May. Cooperative control of multiple mobile manipulators on uneven ground. In Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146) (Vol. 4, pp. 3198-3203). IEEE.

[17] Sugar, T. and Kumar, V., 1998, May. Decentralized control of cooperating mobile manipulators. In Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146) (Vol. 4, pp. 2916-2921). IEEE.

[18] Huang, Q., Sugano, S. and Tanie, K., 1998, May. Motion planning for a mobile manipulator considering stability and task constraints. In Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146) (Vol. 3, pp. 2192-2198). IEEE.

[19] Peterson, L., Austin, D. and Kragic, D., 2000, October. High-level control of a mobile manipulator for door opening. In Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)(Cat. No. 00CH37113) (Vol. 3, pp. 2333-2338). IEEE.

[20] Gardner, J.F. and Velinsky, S.A., 2000. Kinematics of mobile manipulators and implications for design. Journal of Robotic Systems, 17(6), pp.309-320.

[21] Papadopoulos, E. and Poulakakis, J., 2000, October. Planning and model-based control for mobile manipulators. In Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)(Cat. No. 00CH37113) (Vol. 3, pp. 1810-1815). IEEE.

[22] Huang, Q., Tanie, K. and Sugano, S., 2000. Coordinated motion planning for a mobile manipulator considering stability and manipulation. The International Journal of Robotics Research, 19(8), pp.732-742.

[23] Papadopoulos, E. and Rey, D.A., 2000. The force-angle measure of tipover stability margin for mobile manipulators. Vehicle System Dynamics, 33(1), pp.29-48.

[24] Inoue, F., Muralami, T. and Ihnishi, K., 2001. A motion control of mobile manipulator with external force. IEEE/ASME transactions on mechatronics, 6(2), pp.137-142.

[25] Kim, J., Chung, W.K., Youm, Y. and Lee, B.H., 2002, May. Real-time ZMP compensation method using null motion for mobile manipulators. In Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292) (Vol. 2, pp. 1967-1972). IEEE.

[26] Nagatani, K., Hirayama, T., Gofuku, A. and Tanaka, Y., 2002, October. Motion planning for mobile manipulator with keeping manipulability. In IEEE/RSJ International Conference on Intelligent Robots and Systems (Vol. 2, pp. 1663-1668). IEEE.

[27] Yamamoto, Y. and Fukuda, S., 2002, May. Trajectory planning of multiple mobile manipulators with collision avoidance capability. In Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292) (Vol. 4, pp. 3565-3570). IEEE.

[28] Tomizawa, T., Ohya, A. and Yuta, S.I., 2003, September. Remote book browsing system using a mobile manipulator. In 2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422) (Vol. 1, pp. 256-261). IEEE.

[29] Shin, D.H., Hamner, B.S., Singh, S. and Hwangbo, M., 2003, October. Motion planning for a mobile manipulator with imprecise locomotion. In Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453) (Vol. 1, pp. 847-853). IEEE.

[30] Furuno, S., Yamamoto, M. and Mohri, A., 2003, September. Trajectory planning of mobile manipulator with stability considerations. In 2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422) (Vol. 3, pp. 3403-3408). IEEE.

[31] Stroupe, A., Huntsberger, T., Okon, A., Aghazarian, H. and Robinson, M., 2005, August. Behavior-based multi-robot collaboration for autonomous construction tasks. In 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 1495-1500). IEEE.

[32] Katz, D., Horrell, E., Yang, Y., Burns, B., Buckley, T., Grishkan, A., Zhylkovskyy, V., Brock, O. and Learned-Miller, E., 2006, August. The umass mobile manipulator uman: An experimental platform for autonomous mobile manipulation. In Workshop on manipulation in human environments at robotics: science and systems. Citeseer.

[33] Thibodeau, B.J., Deegan, P. and Grupen, R., 2006, May. Static analysis of contact forces with a mobile manipulator. In Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006. (pp. 4007-4012). IEEE.

[34] Najjaran, H. and Goldenberg, A., 2007. Real-time motion planning of an autonomous mobile manipulator using a fuzzy adaptive Kalman filter. Robotics and Autonomous Systems, 55(2), pp.96-106.

[35] Hamner, B., Koterba, S., Shi, J., Simmons, R. and Singh, S., 2009, October. Mobile robotic dynamic tracking for assembly tasks. In 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 2489-2495). IEEE.

[36] Hamner, B., Koterba, S., Shi, J., Simmons, R. and Singh, S., 2010. An autonomous mobile manipulator for assembly tasks. Autonomous Robots, 28(1), p.131.

[37] Chitta, S., Cohen, B. and Likhachev, M., 2010, May. Planning for autonomous door opening with a mobile manipulator. In 2010 IEEE International Conference on Robotics and Automation (pp. 1799-1806). IEEE.

[38] Jain, A. and Kemp, C.C., 2010. EL-E: an assistive mobile manipulator that autonomously fetches objects from flat surfaces. Autonomous Robots, 28(1), p.45.

[39] Holmberg, R. and Khatib, O., 2000. Development and control of a holonomic mobile robot for mobile manipulation tasks. The International Journal of Robotics Research, 19(11), pp.1066-1074.

[40] Bogh, S., Schou, C., Rühr, T., Kogan, Y., Dömel, A., Brucker, M., Eberst, C., Tornese, R., Sprunk, C., Tipaldi, G.D. and Hennessy, T., 2014, June. Integration and assessment of multiple mobile manipulators in a real-world industrial production facility. In ISR/Robotik 2014; 41st International Symposium on Robotics (pp. 1-8). VDE.

[41] Monkman, G.J., 1993. Automated handling of packaging materials. Industrial robot, 20(3), pp.16-19.

[42] Derby, S. and Connolly, C., 2007. Robots at the heart of Schubert packaging machinery lead to great flexibility. Industrial Robot: An International Journal.

[43] Karbassi, J & Mraz, S.J., 2009. Getting a grip with suction cups. Mach. Des. 81. 52-54.

[44] Yu, S. and Gil, M., 2012. Manipulator handling device for assembly of large-size panels. Assembly Automation, 32(4), pp.361-372.

[45] Angelillo, D.J., Park, J., 2002. Efficient robotic packing speeds soft drinks manufacture. Ind. Rob. 29, pp.272–274.

[46] McKeown, C. and Webb, P., 2011. A reactive reconfigurable tool for aerospace structures. Assembly Automation, 31(4), pp.334-343.

[47] Kolluru, R., Valavanis, K.P. and Hebert, T.M., 1998. Modeling, analysis, and performance evaluation of a robotic gripper system for limp material handling. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 28(3), pp.480-486.

[48] Cotugno, G., Turchi, D., Russell, D. and Deacon, G., 2018, October. SecondHands: A Collaborative Maintenance Robot for Automated Warehouses. Implications for the Industry and the Workforce. In International Conference on Inclusive Robotics for a better Society (pp. 195-200). Springer, Cham.

[49] Chow, D.L., Xu, P., Tuna, E., Huang, S., Çavuşoğlu, M.C. and Newman, W., 2017, September. Supervisory control of a DaVinci surgical robot. In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 5043-5049). IEEE.

[50] Ogure, T., Nakabo, Y., Jeong, S. and Yamada, Y., 2009, October. Risk management simulator for low-powered human-collaborative industrial robots. In 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 49-54). IEEE.

[51] Kishi, Y., Yamada, Y. and Yokoyama, K., 2012, October. The role of joint stiffness enhancing collision reaction performance of collaborative robot manipulators. In 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 376-381). IEEE.

[52] Zanchettin, A.M. and Rocco, P., 2013, November. Path-consistent safety in mixed human-robot collaborative manufacturing environments. In 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 1131-1136). IEEE.

[53] Sloth, C. and Petersen, H.G., 2018, October. Computation of Safe Path Velocity for Collaborative Robots. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 6142-6148). IEEE.

[54] Chen, J.H. and Song, K.T., 2018, May. Collision-Free Motion Planning for Human-Robot Collaborative Safety under Cartesian Constraint. In 2018 IEEE International Conference on Robotics and Automation (ICRA) (pp. 1-7). IEEE.

[55] Casalino, A., Bazzi, D., Zanchettin, A.M. and Rocco, P., 2019, May. Optimal Proactive Path Planning for Collaborative Robots in Industrial Contexts. In 2019 International Conference on Robotics and Automation (ICRA) (pp. 6540-6546). IEEE.

[56] Raiola, G., Cardenas, C.A., Tadele, T.S., De Vries, T. and Stramigioli, S., 2018. Development of a Safety-and Energy-Aware Impedance Controller for Collaborative Robots. IEEE Robotics and automation letters, 3(2), pp.1237-1244.

[57] Unhelkar, V.V., Lasota, P.A., Tyroller, Q., Buhai, R.D., Marceau, L., Deml, B. and Shah, J.A., 2018. Human-aware robotic assistant for collaborative assembly: Integrating human motion prediction with planning in time. IEEE Robotics and Automation Letters, 3(3), pp.2394-2401.

[58] Heo, Y.J., Kim, D., Lee, W., Kim, H., Park, J. and Chung, W.K., 2019. Collision Detection for Industrial Collaborative Robots: A Deep Learning Approach. IEEE Robotics and Automation Letters, 4(2), pp.740-746.

[59] Ende, T., Haddadin, S., Parusel, S., Wüsthoff, T., Hassenzahl, M. and Albu-Schäffer, A., 2011, September. A human-centered approach to robot gesture based communication within collaborative working processes. In 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 3367-3374). IEEE.

[60] Moon, A., Parker, C.A., Croft, E.A. and Van der Loos, H.M., 2011, September. Did you see it hesitate?-Empirically grounded design of hesitation trajectories for collaborative robots. In 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 1994-1999). IEEE.

[61] Dumora, J., Geffard, F., Bidard, C., Brouillet, T. and Fraisse, P., 2012, October. Experimental study on haptic communication of a human in a shared human-robot collaborative task. In 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 5137-5144). IEEE.

[62] Campeau-Lecours, A. and Gosselin, C., 2016, October. An anticipative kinematic limitation avoidance algorithm for collaborative robots: Two-dimensional case. In 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 4232-4237). IEEE.

[63] LeBel, P., Gosselin, C. and Campeau-Lecours, A., 2017, September. An anticipative kinematic limitation avoidance algorithm for collaborative robots: Three-dimensional case. In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 3075-3080). IEEE.

[64] Mainprice, J. and Berenson, D., 2013, November. Human-robot collaborative manipulation planning using early prediction of human motion. In 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 299-306). IEEE.

[65] Bestick, A.M., Burden, S.A., Willits, G., Naikal, N., Sastry, S.S. and Bajcsy, R., 2015, September. Personalized kinematics for human-robot collaborative manipulation. In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 1037-1044). IEEE.

[66] El Makrini, I., Merckaert, K., Lefeber, D. and Vanderborght, B., 2017, September. Design of a collaborative architecture for human-robot assembly tasks. In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 1624-1629). IEEE.

[67] Brooks, C., Atreya, M. and Szafir, D., 2018, October. Proactive Robot Assistants for Freeform Collaborative Tasks Through Multimodal Recognition of Generic Subtasks. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 8567-8573). IEEE.

[68] Kruse, D., Radke, R.J. and Wen, J.T., 2015, May. Collaborative human-robot manipulation of highly deformable materials. In 2015 IEEE international conference on robotics and automation (ICRA) (pp. 3782-3787). IEEE.

[69] Schillinger, P., Kohlbrecher, S. and von Stryk, O., 2016, May. Human-robot collaborative high-level control with application to rescue robotics. In 2016 IEEE International Conference on Robotics and Automation (ICRA) (pp. 2796-2802). IEEE.

[70] Paxton, C., Hundt, A., Jonathan, F., Guerin, K. and Hager, G.D., 2017, May. CoSTAR: Instructing collaborative robots with behavior trees and vision. In 2017 IEEE International Conference on Robotics and Automation (ICRA) (pp. 564-571). IEEE.

[71] Munzer, T., Toussaint, M. and Lopes, M., 2017, May. Preference learning on the execution of collaborative human-robot tasks. In 2017 IEEE International Conference on Robotics and Automation (ICRA) (pp. 879-885). IEEE.

[72] Johannsmeier, L. and Haddadin, S., 2016. A hierarchical human-robot interaction-planning framework for task allocation in collaborative industrial assembly processes. IEEE Robotics and Automation Letters, 2(1), pp.41-48.

[73] Hu, B. and Chen, J., 2017. Optimal task allocation for human–machine collaborative manufacturing systems. IEEE Robotics and Automation Letters, 2(4), pp.1933-1940.

[74] El Zaatari, S., Marei, M., Li, W. and Usman, Z., 2019. Cobot programming for collaborative industrial tasks: An overview. Robotics and Autonomous Systems, 116, pp.162-180.

[75] Siegwart, R., Nourbakhsh, I.R. and Scaramuzza, D., 2011. Introduction to autonomous mobile robots. MIT press.

[76] Tzafestas, S.G., 2013. Introduction to mobile robot control. Elsevier.

[77] Siciliano, B. and Khatib, O. eds., 2016. Springer handbook of robotics. Springer.

[78] Röhrig, C., Heß, D. and Künemund, F., 2017, August. Motion controller design for a mecanum wheeled mobile manipulator. In 2017 IEEE Conference on Control Technology and Applications (CCTA) (pp. 444-449). IEEE.

[79] Kamewaka, S. and Uemura, S., 1987. A magnetic guidance method for automated guided vehicles. ieee transactions on magnetics, 23(5), pp.2416-2418.

[80] Reinke, C. and Beinschob, P., 2013, September. Strategies for contour-based self-localization in large-scale modern warehouses. In 2013 IEEE 9th International Conference on Intelligent Computer Communication and Processing (ICCP) (pp. 223-227). IEEE.

[81] Beinschob, P. and Reinke, C., 2014, September. Advances in 3d data acquisition, mapping and localization in modern large-scale warehouses. In *2014 IEEE 10th International Conference on Intelligent Computer Communication and Processing (ICCP)* (pp. 265-271). IEEE.

[82] Beinschob, P., Meyer, M., Reinke, C., Digani, V., Secchi, C. and Sabattini, L., 2017. Semi-automated map creation for fast deployment of AGV fleets in modern logistics. Robotics and Autonomous Systems, 87, pp.281-295.

[83] Ronzoni, D., Olmi, R., Secchi, C. and Fantuzzi, C., 2011, May. AGV global localization using indistinguishable artificial landmarks. In 2011 IEEE International Conference on Robotics and Automation (pp. 287-292). IEEE.

[84] Vasiljevic, G., Petric, F. and Kovacic, Z., 2014, June. Multi-layer mapping-based autonomous forklift localization in an industrial environment. In 22nd Mediterranean Conference on Control and Automation (pp. 1134-1139). IEEE.

[85] Vasiljević, G., Miklić, D., Draganjac, I., Kovačić, Z. and Lista, P., 2016. High-accuracy vehicle localization for autonomous warehousing. Robotics and Computer-Integrated Manufacturing, 42, pp.1-16.

[86] Saarinen, J., Andreasson, H., Stoyanov, T. and Lilienthal, A.J., 2013, November. Normal distributions transform Monte-Carlo localization (NDT-MCL). In 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 382-389). IEEE.

[87] Borenstein, J., Everett, H.R., Feng, L. and Wehe, D., 1997. Mobile robot positioning: Sensors and techniques. Journal of robotic systems, 14(4), pp.231-249.

[88] Huang, S. and Dissanayake, G., 1999. Robot localization: An introduction. Wiley Encyclopedia of Electrical and Electronics Engineering, pp.1-10.

[89] Durrant-Whyte, H. and Bailey, T., 2006. Simultaneous localization and mapping: part I. IEEE robotics & automation magazine, 13(2), pp.99-110.

[90] Aulinas, J., Petillot, Y.R., Salvi, J. and Lladó, X., 2008. The slam problem: a survey. CCIA, 184(1), pp.363-371.

[91] Montemerlo, M., Thrun, S., Koller, D. and Wegbreit, B., 2002. FastSLAM: A factored solution to the simultaneous localization and mapping problem. Aaai/iaai, 593598.

[92] Xue, Y., Tian, G., Song, B. and Zhang, T., 2012. Distributed environment representation and object localization system in intelligent space. Journal of Control Theory and Applications, 10(3), pp.371-379.

[93] Betke, M. and Gurvits, L., 1997. Mobile robot localization using landmarks. IEEE transactions on robotics and automation, 13(2), pp.251-263.

[94] Shimshoni, I., 2002. On mobile robot localization from landmark bearings. IEEE Transactions on Robotics and Automation, 18(6), pp.971-976.

[95] Bekris, K.E., Argyros, A.A. and Kavraki, L.E., 2004, April. Angle-based methods for mobile robot navigation: Reaching the entire plane. In IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 (Vol. 3, pp. 2373-2378). IEEE.

[96] Yuen, D.C. and MacDonald, B.A., 2005. Vision-based localization algorithm based on landmark matching, triangulation, reconstruction, and comparison. IEEE Transactions on robotics, 21(2), pp.217-226.

[97] Alcantarilla, P.F., Oh, S.M., Mariottini, G.L., Bergasa, L.M. and Dellaert, F., 2010, May. Learning visibility of landmarks for vision-based localization. In 2010 IEEE International Conference on Robotics and Automation (pp. 4881-4888). IEEE.

[98] Loevsky, I. and Shimshoni, I., 2010. Reliable and efficient landmark-based localization for mobile robots. Robotics and Autonomous Systems, 58(5), pp.520-528.

[99] Sert, H., Kökösy, A. and Perruquetti, W., 2011, May. A single landmark based localization algorithm for non-holonomic mobile robots. In 2011 IEEE International Conference on Robotics and Automation (pp. 293-298). IEEE.

[100] Mitterer, T., Gietler, H., Faller, L.M. and Zangl, H., 2019. Artificial Landmarks for Trusted Localization of Autonomous Vehicles Based on Magnetic Sensors. Sensors, 19(4), p.813.

[101] Schaff, C., Yunis, D., Chakrabarti, A. and Walter, M.R., 2017, September. Jointly optimizing placement and inference for beacon-based localization. In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 6609-6616). IEEE.

[102] Beinhofer, M., Müller, J. and Burgard, W., 2013. Effective landmark placement for accurate and reliable mobile robot navigation. Robotics and Autonomous Systems, 61(10), pp.1060-1069.

[103] Rosenfeld, A. and Pfaltz, J.L., 1966. Sequential operations in digital picture processing. Journal of the ACM (JACM), 13(4), pp.471-494.

[104] Jarvis, R.A., 1985. Collision-free trajectory planning using distance transforms. Transactions of the Institution of Engineers, Australia. Mechanical engineering, 10(3), pp.187-191.

[105] Fabbri, R., Costa, L.D.F., Torelli, J.C. and Bruno, O.M., 2008. 2D Euclidean distance transform algorithms: A comparative survey. ACM Computing Surveys (CSUR), 40(1), pp.1-44.

[106] Martelli, A., 1972. Edge detection using heuristic search methods. Computer graphics and image processing, 1(2), pp.169-182.

[107] Ziou, D. and Tabbone, S., 1998. Edge detection techniques-an overview. Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii, 8, pp.537-559.

[108] Hueckel, M.H., 1971. An operator which locates edges in digitized pictures. Journal of the ACM (JACM), 18(1), pp.113-125.

[109] Pavlidis, T. and Horowitz, S.L., 1974. Segmentation of plane curves. IEEE transactions on Computers, 100(8), pp.860-870.

[110] Gonzalez, J., Ollero, A. and Reina, A., 1994, May. Map building for a mobile robot equipped with a 2D laser rangefinder. In Proceedings of the 1994 IEEE International Conference on Robotics and Automation (pp. 1904-1909). IEEE.

[111] Vandorpe, J., Van Brussel, H. and Xu, H., 1996, April. Exact dynamic map building for a mobile robot using geometrical primitives produced by a 2D range finder. In Proceedings of IEEE International Conference on Robotics and Automation (Vol. 1, pp. 901-908). IEEE.

[112] Zhang, L. and Ghosh, B.K., 2000, April. Line segment based map building and localization using 2D laser rangefinder. In Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065) (Vol. 3, pp. 2538-2543). IEEE.

[113] Sack, D. and Burgard, W., 2004, July. A comparison of methods for line extraction from range data. In Proc. of the 5th IFAC symposium on intelligent autonomous vehicles (IAV) (Vol. 33).

[114] Nguyen, V., Gächter, S., Martinelli, A., Tomatis, N. and Siegwart, R., 2007. A comparison of line extraction algorithms using 2D range data for indoor mobile robotics. Autonomous Robots, 23(2), pp.97-111.

[115] Hough, P.V., 1962. Method and means for recognizing complex patterns. U.S. Patent 3,069,654.

[116] Duda, R.O. and Hart, P.E., 1972. Use of the Hough transformation to detect lines and curves in pictures. Communications of the ACM, 15(1), pp.11-15.

[117] Walsh, D. and Raftery, A.E., 2002. Accurate and efficient curve detection in images: the importance sampling Hough transform. Pattern Recognition, 35(7), pp.1421-1431.

[118] Berger, A.D. and Khosla, P.K., 1990. The modified adaptive Hough transform (MAHT). Journal of Robotic Systems, 7(2), pp.277-290.

[119] Ziou, D. and Tabbone, S., 1998. Edge detection techniques-an overview. Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii, 8, pp.537-559.

[120] Xu, L., Oja, E. and Kultanen, P., 1990. A new curve detection method: randomized Hough transform (RHT). Pattern recognition letters, 11(5), pp.331-338.

[121] Vaughn, D.L. and Arkin, R.C., 1991, April. Workstation recognition using a constrained edge-based hough transform for mobile robot navigation. In Sensor Fusion III: 3D Perception and Recognition (Vol. 1383, pp. 503-514). International Society for Optics and Photonics.

[122] Forsberg, J., Larsson, U., Ahman, P. and Wernersson, A., 1993, May. The Hough transform inside the feedback loop of a mobile robot. In [1993] Proceedings IEEE International Conference on Robotics and Automation (pp. 791-798). IEEE.

[123] Forsberg, J., Larsson, U. and Wernersson, A., 1995. Mobile robot navigation using the range-weighted Hough transform. IEEE Robotics & Automation Magazine, 2(1), pp.18-26.

[124] Larsson, U., Forsberg, J. and Wernersson, A., 1996. Mobile robot localization: integrating measurements from a time-of-flight laser. IEEE Transactions on Industrial Electronics, 43(3), pp.422-431.

[125] Jensfelt, P. and Christensen, H., 1998, May. Laser based position acquisition and tracking in an indoor environment. In International Symposium on Robotics and Automation-ISRA (Vol. 98).

[126] Anousaki, G.C. and Kyriakopoulos, K.J., 1999. Simultaneous localization and map building for mobile robot navigation. IEEE Robotics & Automation Magazine, 6(3), pp.42-53.

[127] Großmann, A. and Poli, R., 2001. Robust mobile robot localization from sparse and noisy proximity readings using Hough transform and probability grids. Robotics and Autonomous Systems, 37(1), pp.1-18.

[128] Surmann, H., Lingemann, K., Nüchter, A. and Hertzberg, J., 2001, April. A 3D laser range finder for autonomous mobile robots. In Proceedings of the 32nd ISR (International Symposium on Robotics) (Vol. 19, No. 21, pp. 153-158).

[129] Iocchi, L. and Nardi, D., 2002. Hough localization for mobile robots in polygonal environments. Robotics and Autonomous Systems, 40(1), pp.43-58.

[130] Pfister, S.T., Roumeliotis, S.I. and Burdick, J.W., 2003, September. Weighted line fitting algorithms for mobile robot map building and efficient data representation. In 2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422) (Vol. 1, pp. 1304-1311). IEEE.

[131] Ravankar, A.A., Hoshino, Y., Ravankar, A., Jixin, L., Emaru, T. and Kobayashi, Y., 2015. Algorithms and a framework for indoor robot mapping in a noisy environment using clustering in spatial and hough domains. International Journal of Advanced Robotic Systems, 12(3), p.27.

[132] Lu, X. and Manduchi, R., 2005, April. Detection and localization of curbs and stairways using stereo vision. In Proceedings of the 2005 IEEE International Conference on Robotics and Automation (pp. 4648-4654). IEEE.

[133] Åstrand, B. and Baerveldt, A.J., 2005. A vision based row-following system for agricultural field machinery. Mechatronics, 15(2), pp.251-269.

[134] Ronnback, S., Hyyppa, K. and Wernersson, A., 2005, August. On passing a doorway with an autonomous Internet connected wheelchair using MATLAB. In 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 1532-1537). IEEE.

[135] Choi, Y.H., Lee, T.K. and Oh, S.Y., 2008. A line feature based SLAM with low grade range sensors using geometric constraints and active exploration for mobile robot. Autonomous Robots, 24(1), pp.13-27.

[136] Fernández, C., Moreno, V., Curto, B. and Vicente, J.A., 2010. Clustering and line detection in laser range measurements. Robotics and Autonomous Systems, 58(5), pp.720-726.

[137] Winterhalter, W., Fleckenstein, F.V., Dornhege, C. and Burgard, W., 2018. Crop row detection on tiny plants with the pattern hough transform. IEEE Robotics and Automation Letters, 3(4), pp.3394-3401.

[138] Russell, S.J. and Norvig, P., 1995. Artificial Intelligence: A Modern Approach Prentice Hall. New Jersey.

[139] Pirjanian, P., 1999. Behavior coordination mechanisms-state-of-the-art. Technical report, University of Southern California, Institute for Robotics and Intelligent Systems Technical Report IRIS-99-375.

[140] Brooks, R., 1986. A robust layered control system for a mobile robot. IEEE journal on robotics and automation, 2(1), pp.14-23.

[141] Košecká, J. and Bajcsy, R., 1994. Discrete event systems for autonomous mobile agents. Robotics and Autonomous Systems, 12(3-4), pp.187-198.

[142] Maes, P., 1989. How to do the right thing. Connection science, 1(3), pp.291-323.

[143] Rosenblatt, J.K., 1997. DAMN: A distributed architecture for mobile navigation. Journal of Experimental & Theoretical Artificial Intelligence, 9(2-3), pp.339-360.

[144] Saffiotti, A., Konolige, K. and Ruspini, E.H., 1995. A multivalued logic approach to integrating planning and control. Artificial intelligence, 76(1-2), pp.481-526.

[145] Arkin, R.C., 1989. Motor schema—based mobile robot navigation. The International journal of robotics research, 8(4), pp.92-112.

[146] Pirjanian, P., 2000. Multiple objective behavior-based control. Robotics and Autonomous Systems, 31(1-2), pp.53-60.

[147] Loetzsch, M., Risler, M. and Jungel, M., 2006, October. XABSL-a pragmatic approach to behavior engineering. In 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 5124-5129). IEEE.

[148] Konolige, K., 1997, September. Colbert: A language for reactive control in sapphira. In Annual Conference on Artificial Intelligence (pp. 31-52). Springer, Berlin, Heidelberg.

[149] MacKenzie, D.C., 1996. A design methodology for the configuration of behavior-based mobile robots (Doctoral dissertation, College of Computing, Georgia Institute of Technology).

[150] Brooks, R.A., 1990. The Behavior Language; Users' Guide, Massachusetts Institute of Technology. Artificial Intelligence Laboratory, Al. Memo, (1227).

[151] Stroupe, A., Huntsberger, T., Okon, A., Aghazarian, H. and Robinson, M., 2005, August. Behavior-based multi-robot collaboration for autonomous construction tasks. In 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 1495-1500). IEEE.

[152] Huq, R., Mann, G.K. and Gosine, R.G., 2006. Behavior-modulation technique in mobile robotics using fuzzy discrete event system. IEEE Transactions on Robotics, 22(5), pp.903-916.

[153] Zhang, H., Liu, S. and Yang, S.X., 2006, October. A hybrid robot navigation approach based on partial planning and emotion-based behavior coordination. In 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 1183-1188). IEEE.

[154] Argall, B., Browning, B. and Veloso, M., 2007, April. Learning to select state machines using expert advice on an autonomous robot. In Proceedings 2007 IEEE International Conference on Robotics and Automation (pp. 2124-2129). IEEE.

[155] Tousignant, S., Van Wyk, E. and Gini, M., 2012, May. Xrobots: A flexible language for programming mobile robots based on hierarchical state machines. In 2012 IEEE International Conference on Robotics and Automation (pp. 1773-1778). IEEE.

[156] Kurt, A. and Özgüner, Ü., 2013. Hierarchical finite state machines for autonomous mobile systems. Control Engineering Practice, 21(2), pp.184-194.

[157] Allgeuer, P. and Behnke, S., 2018. Hierarchical and state-based architectures for robot behavior planning and control. arXiv preprint arXiv:1809.11067.

[158] Marzinotto, A., Colledanchise, M., Smith, C. and Ögren, P., 2014, May. Towards a unified behavior trees framework for robot control. In 2014 IEEE International Conference on Robotics and Automation (ICRA) (pp. 5420-5427). IEEE.

[159] Schillinger, P., Kohlbrecher, S. and von Stryk, O., 2016, May. Human-robot collaborative high-level control with application to rescue robotics. In 2016 IEEE International Conference on Robotics and Automation (ICRA) (pp. 2796-2802). IEEE.

[160] Maniatopoulos, S., Schillinger, P., Pong, V., Conner, D.C. and Kress-Gazit, H., 2016, May. Reactive high-level behavior synthesis for an atlas humanoid robot. In 2016 IEEE International Conference on Robotics and Automation (ICRA) (pp. 4192-4199). IEEE.

[161] Petri, C.A., 1962. Kommunikation mit Automaten./(1966). Communication with Automata. Technical Report RADC-TR-65-377, Rome Air Dev. Center, New York.

[162] Kim, G., Chung, W., Kim, M. and Lee, C., 2003, September. Tripodal schematic design of the control architecture for the service robot PSR. In 2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422) (Vol. 2, pp. 2792-2797). IEEE.

[163] Kim, G., Chung, W., Kim, M. and Lee, C., 2004, April. Implementation of multi-functional service robots using tripodal schematic control architecture. In IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 (Vol. 4, pp. 4005-4010). IEEE.

[164] King, J., Pretty, R.K. and Gosine, R.G., 2003. Coordinated execution of tasks in a multiagent environment. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 33(5), pp.615-619.

[165] Caccia, M., Coletta, P., Bruzzone, G. and Veruggio, G., 2005. Execution control of robotic tasks: a Petri net-based approach. Control Engineering Practice, 13(8), pp.959-971.

[166] Kim, G. and Chung, W., 2007. Navigation behavior selection using generalized stochastic Petri nets for a service robot. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 37(4), pp.494-503.

[167] Moon, C.B. and Chung, W., 2010, May. Design of navigation behaviors and the selection framework with generalized stochastic petri nets toward dependable navigation of a mobile robot. In 2010 IEEE international conference on robotics and automation (pp. 2989-2994). IEEE.

[168] Costelha, H. and Lima, P., 2007, October. Modelling, analysis and execution of robotic tasks using petri nets. In 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 1449-1454). IEEE.

[169] Asfour, T., Ly, D.N., Regenstein, K. and Dillmann, R., 2004. Coordinated task execution for humanoid robots. In The 9th International Symposium on Experimental Robotics (ISER 04.

[170] Fernández, J.L., Sanz, R., Paz, E. and Alonso, C., 2008, May. Using hierarchical binary Petri nets to build robust mobile robot applications: RoboGraph. In 2008 IEEE International Conference on Robotics and Automation (pp. 1372-1377). IEEE.

[171] Lacerda, B. and Lima, P.U., 2011, September. LTL-based decentralized supervisory control of multi-robot tasks modelled as Petri nets. In 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 3081-3086). IEEE.

[172] Figat, M. and Zieliński, C., 2019, May. Methodology of designing multi-agent robot control systems utilising hierarchical Petri nets. In 2019 International Conference on Robotics and Automation (ICRA) (pp. 3363-3369). IEEE.

[173] Murata, T., 1989. Petri nets: Properties, analysis and applications. Proceedings of the IEEE, 77(4), pp.541-580.

[174] SEBBATA, W. and KENK, M., 2020, September. An adaptive robotic grasping with a 2-finger gripper based on deep learning network. In 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA) (pp.--). IEEE.

[175] Redmon, J. and Farhadi, A., 2017. YOLO9000: better, faster, stronger. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7263-7271).

[176] Monkman, G.J., Hesse, S., Steinmann, R. and Schunk, H., 2007. Robot grippers. John Wiley & Sons.

[177] Amend, J.R., Brown, E., Rodenberg, N., Jaeger, H.M. and Lipson, H., 2012. A positive pressure universal gripper based on the jamming of granular material. IEEE transactions on robotics, 28(2), pp.341-350.