ENPM662 – INTRODUCTION TO ROBOT MODELLING

PROJECT - 2

VERSA-BOT V 1.0 – A SHOP -FLOOR MOBILE MANIPULATOR

A PROJECT REPORT BY

HARIHARASUDAN MURALIDHARAN – 120172656

SAI JAGADEESH MURALIKRISHNAN- 120172243

TABLE OF CONTENTS

ABSTRACT

This report presents the comprehensive design and simulation of a mobile manipulator with a 7-degree-of-freedom robotic arm, meticulously modeled in SolidWorks software and then exported to a URDF for compatibility with ROS 2. Using ROS 2, a package was built, including URDF, launch files, and controllers, to the manipulator's operation within the Gazebo simulation environment. A significant focus was placed on the integration of Rviz for the real-time visualization of lidar data. The computation and validation of both forward and inverse kinematics, ensuring accurate control and movement of the manipulator. The report includes detailed simulations demonstrating the controller within the Gazebo environment, showcasing the system's operational capabilities. Finally, the report shows the underlying assumptions and challenges faced during the process.

CHAPTER – 1: INTRODUCTION

In future, shop floor and retail floor robots, particularly mobile manipulators, are set to revolutionize the way businesses operate, significantly elevating the market landscape. These advanced robots will transform retail and manufacturing environments into highly efficient, smart, and customer-focused spaces [2]. Some main significance of these robots can be realized with their ability to navigate autonomously, handle products, and interact with customers, these robots will streamline operations, drastically reduce the time taken for tasks like restocking shelves or assembling products. This will lead to a significant increase in productivity and a decrease in operational costs [3].

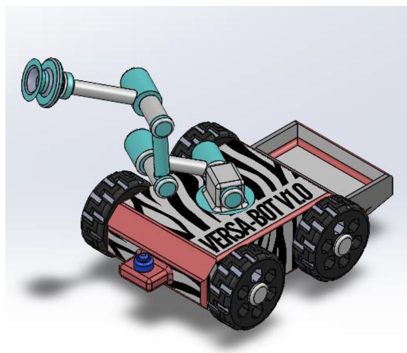


Fig 1.1: Versa-BOT V 1.0 Model

This report explains the goals of the project with every detail involved in the process for modelling and controlling the robot using the defined controller. Kinematics calculation and validation is done using MATLAB software using the Peter Corke toolbox and Inverse kinematics is found using the algorithm used in homework 4 in finding the trajectory of the end-effector.

Goals of the Project:

- Modelling the Versa-BOT in the SolidWorks software with 7 degrees of freedom
- Exporting the 3-D model into URDF.
- Moving the mobile manipulator URDF to the Ubuntu OS to work with ROS 2.
- Spawning the robot in the Gazebo environment in a empty world like did in project 1.
- Including LIDAR in the robot and RViz visualization of the sensor.
- Creating an algorithm for controlling the robot in the gazebo environment.

CHAPTER – 2: APPLICATIONS

Shop Floor Application

- Handling and Sorting applications
- Machine Tending Operation
- Retail Stocking Efficiency

Retail Floor Applications

- Innovative Customer Assistance.
- Robust Inventory Management

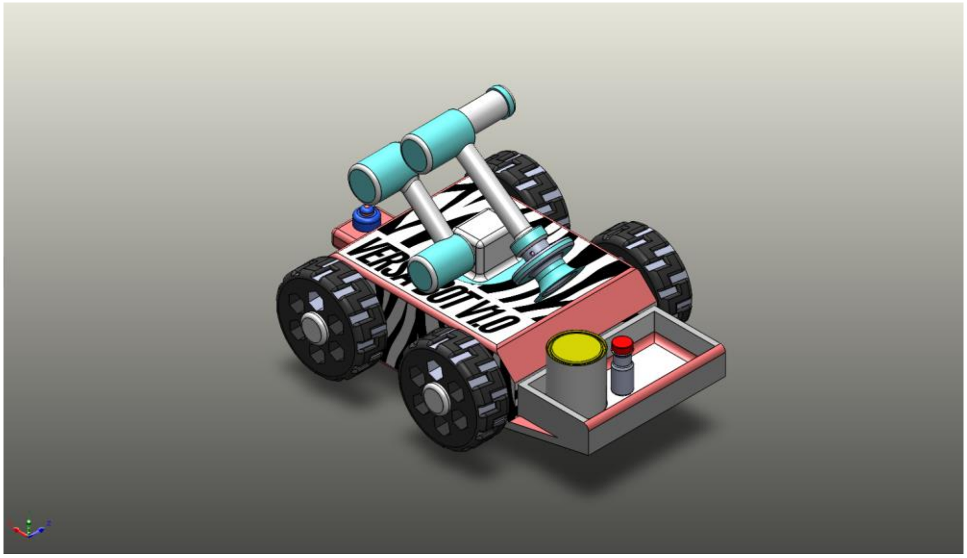


Fig 2.1: The Robot Involved In The Application

CHAPTER – 3: VERSA-BOT V 1.0

3.1. Robot Type and Degrees of Freedom

The type of robot is a mobile manipulator. The robot design has 7 degrees of freedom in total. The mobile base with 3 degrees of freedom and the manipulator has 4 degrees of freedom on top of the mobile base. The robot with revolute joints for the manipulator operation.

The two primary components: the mobile base and the manipulator arm.

3.2. Dimensions of the Robot

MOBILE BASE	
Length	762.00 mm
Width	381.00 mm
Height	157.85 mm
Wheelbase	304.80 mm
Wheel Radius	108.00 mm
MANIPULATOR	
Arm Base Length	100.00 mm
First Link Length	190.50 mm
Second Link Length	190.50 mm
Third Link Length	328.93 mm
Radius of the Link	38.00 mm
End-Effector	Vacuum Gripper

3.3. CAD Models

The CAD models with the 2-D drawing with dimensions are shown below for both mobile base and manipulator.

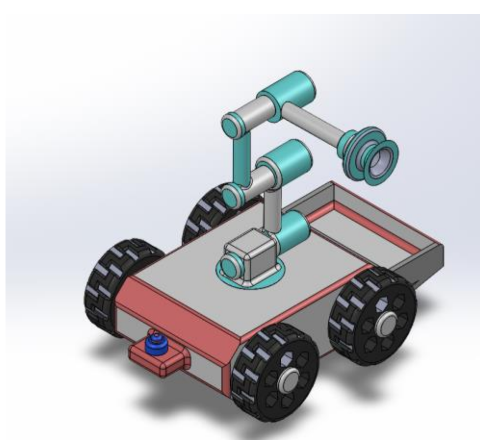


Fig 3.1: Isometric view of the Model

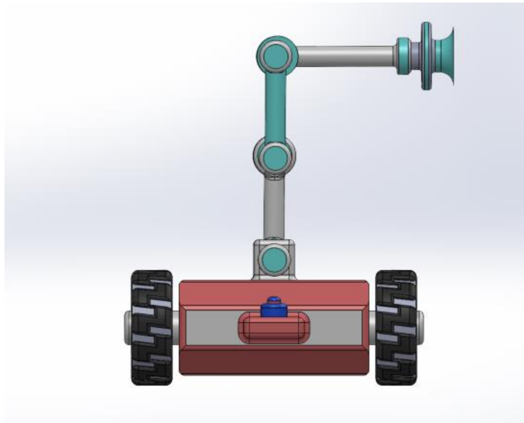


Fig 3.2: Front-View of the Robot

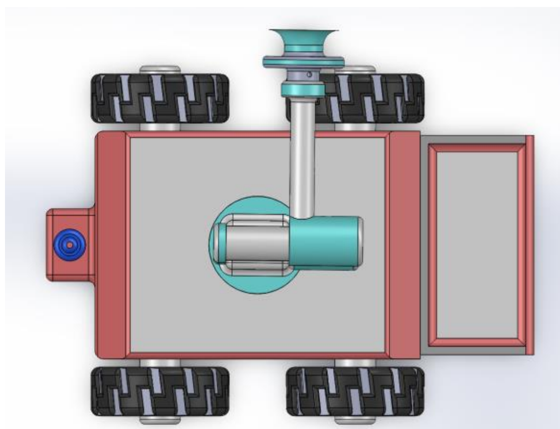


Fig 3.3: Top-View of the Robot

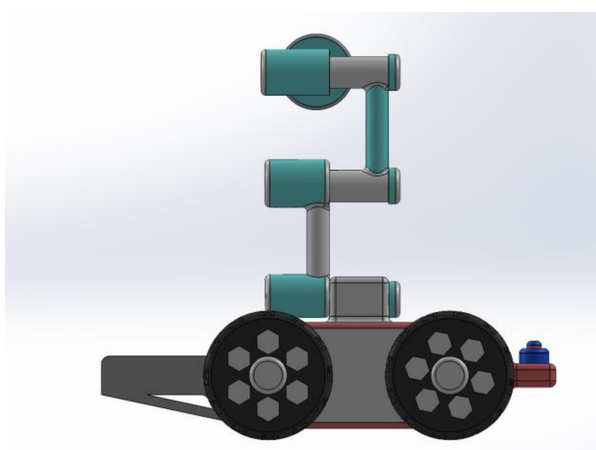


Fig 3.4: Right-View of the Robot

Mobile Base

The mobile base is designed with 3 degrees of freedom. The mobile base has 4 wheels with differential drive.

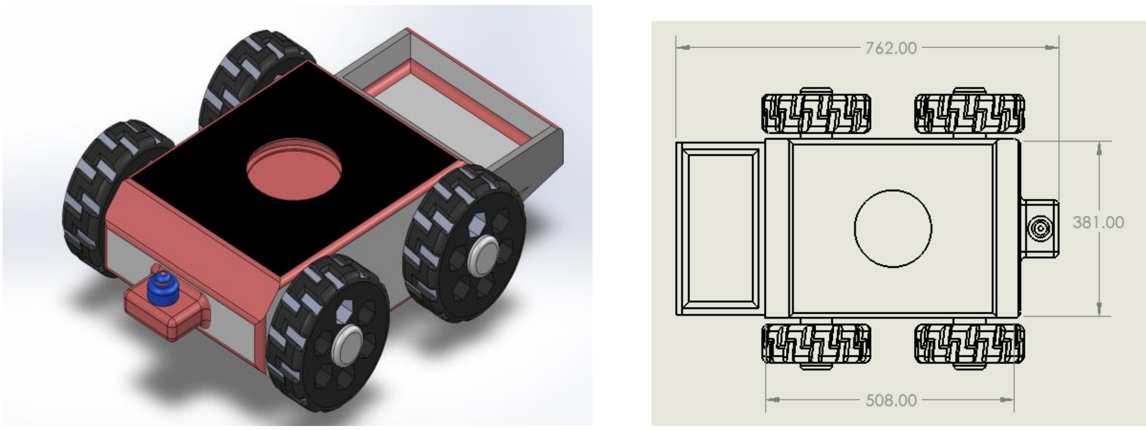


Fig 3.5: Mobile Base Model with 2-D CAD

Manipulator Arm

Mounted top of the mobile base is the manipulator arm, which is equipped with 4 degrees of freedom m. This configuration of the arm with revolute joints, mimicking a human arm, allows for complex manipulative tasks in industry.

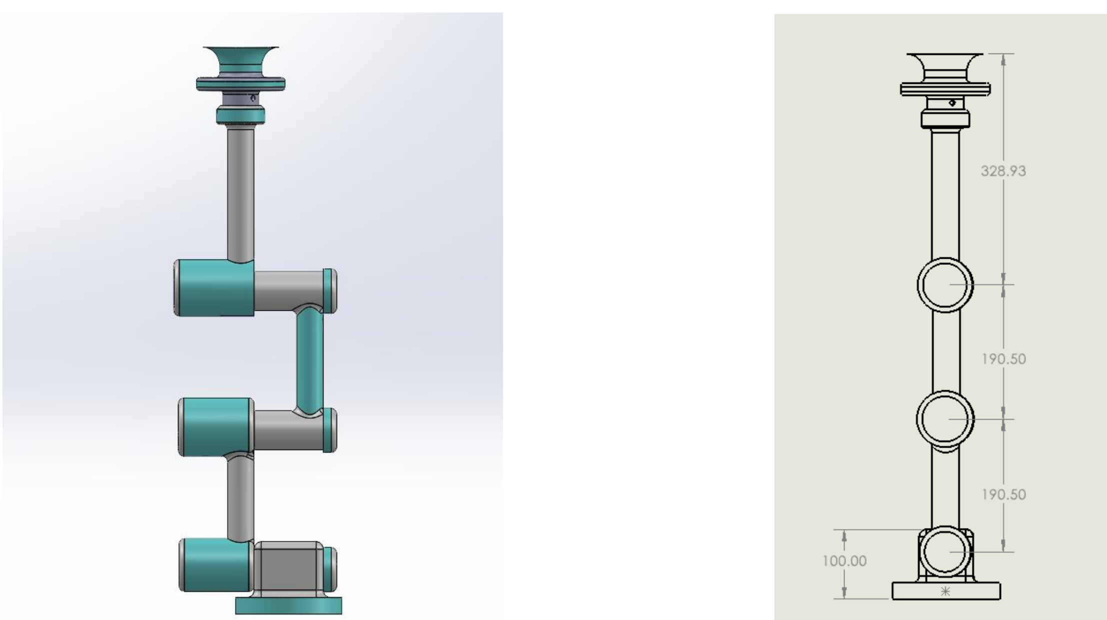
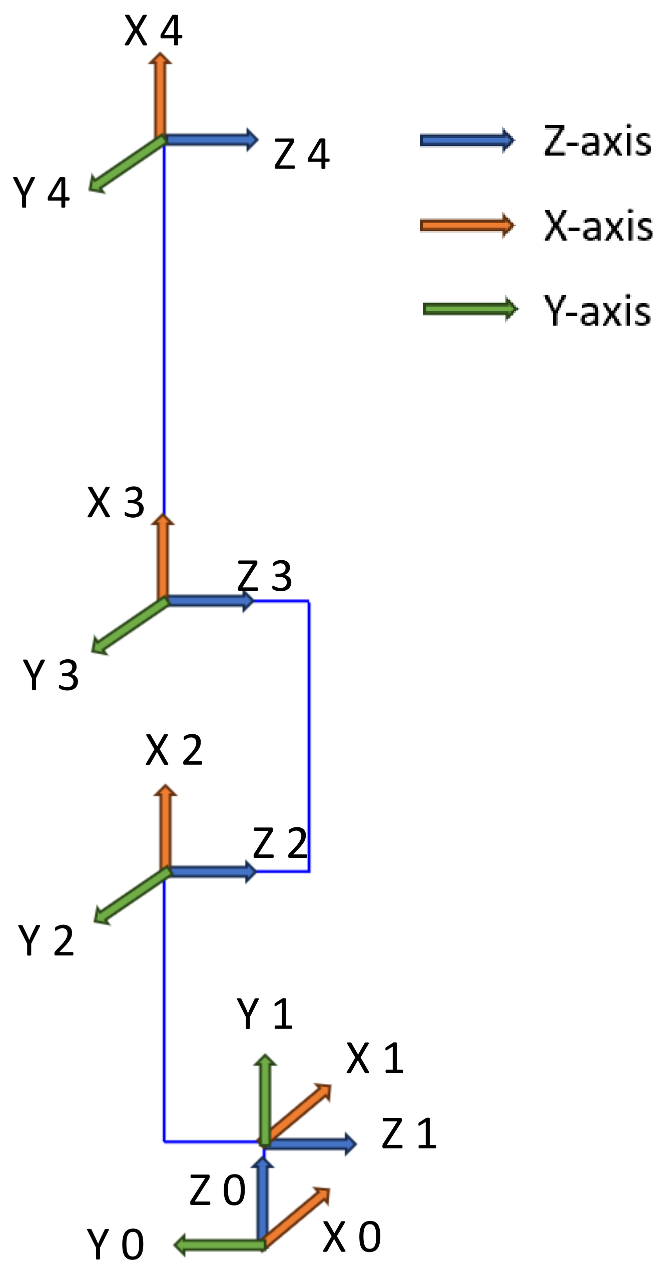


Fig 3.6: Manipulator Arm Model with 2-D CAD

CHAPTER – 4: D-H PARAMETERS

Frame Assignment for the Robot: .



D-H Table:

Transformations	θ_i	α_i	a_i	d_i
0 -> 1	θ_1	90°	0	38.1 mm
1 -> 2	$\theta_2 - 90°$	0	-190.5 mm	-70.60 mm
2 -> 3	θ_3	0	-190.5 mm	0
3 -> 4	θ_4	0	-328.93 mm	0

Table 4.1: D-H Table of the Transformations

CHAPTER – 5: KINEMATICS

5.1. Forward Kinematics

The Forward Kinematics is computed for the 4 degrees of freedom manipulator without considering the mobile base, Since the mobile base is stationary while the manipulator does its operation. Steps need to follow to calculate the forward kinematics of manipulator is,

- Finding the D-H parameters after assigning the frames for the manipulator.
- The algorithm is framed to calculate the matrix using previous homework.
- Using the D-H table we can find the Final transformation matrix.
- Finally, the pose of the end effector is known by using final transformation matrix T_0^4

Algorithm To Compute Forward Kinematics:

- Individual transformation matrices can be found using the transformation matrix below.

$$\begin{bmatrix} cos\theta_i & -sin\theta_i cos\alpha_i & sin\theta_i sin\alpha_i & a_i cos\theta_i \\ sin\theta_i & cos\theta_i cos\alpha_i & -cos\theta_i sin\alpha_i & a_i sin\theta_i \\ 0 & sin\alpha_i & cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- The final transformation matrix T_0^4 can be found by finding the induvial transformation matrix and multiplying them accordingly.

$$T_0^4 = T_0^1 * T_1^2 * T_2^3 * T_3^4$$

- In the final column in the transformation matrix shows $P_x, P_y, and\ P_z$, which is the pose in the X, Y and Z- axis.
- The final transformation matrix T_0^4 is shown below,

```
the transformation matrix is:
⎡(-sin(θ₂)⋅sin(θ₃)⋅cos(θ₁) + cos(θ₁)⋅cos(θ₂)⋅cos(θ₃))⋅sin(θ₄) + (sin(θ₂)⋅cos(θ
⎢
⎢(-sin(θ₁)⋅sin(θ₂)⋅sin(θ₃) + sin(θ₁)⋅cos(θ₂)⋅cos(θ₃))⋅sin(θ₄) + (sin(θ₁)⋅sin(θ
⎢
⎢                (sin(θ₂)⋅sin(θ₃) - cos(θ₂)⋅cos(θ₃))⋅cos(θ₄) + (sin(θ₂)⋅cos(θ₃
⎢
⎣                                                              0

₁)⋅cos(θ₃) + sin(θ₃)⋅cos(θ₁)⋅cos(θ₂))⋅cos(θ₄)  (-sin(θ₂)⋅sin(θ₃)⋅cos(θ₁) + cos

₂)⋅cos(θ₃) + sin(θ₁)⋅sin(θ₃)⋅cos(θ₂))⋅cos(θ₄)  (-sin(θ₁)⋅sin(θ₂)⋅sin(θ₃) + sin

) + sin(θ₃)⋅cos(θ₂))⋅sin(θ₄)                                   -(sin(θ₂)⋅sin(θ


(θ₁)⋅cos(θ₂)⋅cos(θ₃))⋅cos(θ₄) - (sin(θ₂)⋅cos(θ₁)⋅cos(θ₃) + sin(θ₃)⋅cos(θ₁)⋅cos

(θ₁)⋅cos(θ₂)⋅cos(θ₃))⋅cos(θ₄) - (sin(θ₁)⋅sin(θ₂)⋅cos(θ₃) + sin(θ₁)⋅sin(θ₃)⋅cos

₃) - cos(θ₂)⋅cos(θ₃))⋅sin(θ₄) + (sin(θ₂)⋅cos(θ₃) + sin(θ₃)⋅cos(θ₂))⋅cos(θ₄)

                             0

(θ₂))⋅sin(θ₄)  sin(θ₁)   -328.93⋅(-sin(θ₂)⋅sin(θ₃)⋅cos(θ₁) + cos(θ₁)⋅cos(θ₂)⋅c

(θ₂))⋅sin(θ₄)  -cos(θ₁)  -328.93⋅(-sin(θ₁)⋅sin(θ₂)⋅sin(θ₃) + sin(θ₁)⋅cos(θ₂)⋅c

                  0                                  -328.93⋅(sin(θ₂)⋅sin(

                  0

os(θ₃))⋅sin(θ₄) - 328.93⋅(sin(θ₂)⋅cos(θ₁)⋅cos(θ₃) + sin(θ₃)⋅cos(θ₁)⋅cos(θ₂))⋅c

os(θ₃))⋅sin(θ₄) - 328.93⋅(sin(θ₁)⋅sin(θ₂)⋅cos(θ₃) + sin(θ₁)⋅sin(θ₃)⋅cos(θ₂))⋅c

θ₃) - cos(θ₂)⋅cos(θ₃))⋅cos(θ₄) - 328.93⋅(sin(θ₂)⋅cos(θ₃) + sin(θ₃)⋅cos(θ₂))⋅si

                                                                   1

os(θ₄) - 70.6⋅sin(θ₁) - 190.5⋅sin(θ₂)⋅cos(θ₁)⋅cos(θ₃) - 190.5⋅sin(θ₂)⋅cos(θ₁)

os(θ₄) - 190.5⋅sin(θ₁)⋅sin(θ₂)⋅cos(θ₃) - 190.5⋅sin(θ₁)⋅sin(θ₂) - 190.5⋅sin(θ₁)

n(θ₄) - 190.5⋅sin(θ₂)⋅sin(θ₃) + 190.5⋅cos(θ₂)⋅cos(θ₃) + 190.5⋅cos(θ₂) + 38.1


- 190.5⋅sin(θ₃)⋅cos(θ₁)⋅cos(θ₂)⎤

⋅sin(θ₃)⋅cos(θ₂) + 70.6⋅cos(θ₁)⎥
                               ⎥
                               ⎥
                               ⎥
                               ⎦
```

Fig 5.1: Final Transformation Matrix

5.2. Inverse Kinematics

Setup Of Jacobian Matrix Using the Method 1 Involving Z_i And O_i

- To calculate the Jacobian matrix J for a robot arm using the first method of lecture 8, these steps for each joint (i) will follow.
- T_i^0, the final transformation matrix is found once all individual matrices are found from link 1 to 4.
- Calculate O_i, the position of the origin of the i_{th} frame in the base frame, which is the fourth column of T_i^0, excluding the last row.
- Calculate Z_i, the orientation of the z-axis in the base frame, which is the third column of final T_i^0.
- Calculate J_i, the i_{th} column of the Jacobian matrix for a revolute joint, using the form for revolute joints.

$$J_i = \begin{bmatrix} Z_{i-1} \times (O_n - O_{i-1}) \\ Z_{i-1} \end{bmatrix}$$

- where n is the number of the end-effector frame, X denotes the cross product, and O_n is the position of the end-effector on the wall.
- Now, we can arrange the components found in the T_i^0 and find the Jacobian matrix of every i link and finally compute the final Jacobian matrix as
  $J = [J_1, J_2, J_3, J_4]$
- Finally, we will be getting a Jacobian matrix which is printed in the terminal.

```
Jacobian matrix:
328.93·(-sin(θ_1)·sin(θ_2)·sin(θ_3) + sin(θ_1)·cos(θ_2)·cos(θ_3))·sin(θ_4) + 328.93·
-328.93·(-sin(θ_2)·sin(θ_3)·cos(θ_1) + cos(θ_1)·cos(θ_2)·cos(θ_3))·sin(θ_4) - 328.93




(sin(θ_1)·sin(θ_2)·cos(θ_3) + sin(θ_1)·sin(θ_3)·cos(θ_2))·cos(θ_4) + 190.5·sin(θ_1)·si
·(sin(θ_2)·cos(θ_1)·cos(θ_3) + sin(θ_3)·cos(θ_1)·cos(θ_2))·cos(θ_4) - 70.6·sin(θ_1) -
                                        0
                                        0
                                        0
                                        1
n(θ_2)·cos(θ_3) + 190.5·sin(θ_1)·sin(θ_2) + 190.5·sin(θ_1)·sin(θ_3)·cos(θ_2) - 70.6·c
190.5·sin(θ_2)·cos(θ_1)·cos(θ_3) - 190.5·sin(θ_2)·cos(θ_1) - 190.5·sin(θ_3)·cos(θ_1)·




os(θ_1)
cos(θ_2)
```

```
          (-328.93·(-sin(θ_1)·sin(θ_2)·sin(θ_3) + sin(θ_1)·cos(θ_2)·cos(θ_3))·sin(θ_4)





  - 328.93·(sin(θ_1)·sin(θ_2)·cos(θ_3) + sin(θ_1)·sin(θ_3)·cos(θ_2))·cos(θ_4) - 190.5·



                -(-328.93·(sin(θ_2)·sin(θ_3) - cos(θ_2)·cos(θ_3))·cos(θ_4) - 328.93·
                -(-328.93·(sin(θ_2)·sin(θ_3) - cos(θ_2)·cos(θ_3))·cos(θ_4) - 328.93·
 sin(θ_1)·sin(θ_2)·cos(θ_3) - 190.5·sin(θ_1)·sin(θ_2) - 190.5·sin(θ_1)·sin(θ_3)·cos(θ_2
```

```
(sin(θ_2)·cos(θ_3) + sin(θ_3)·cos(θ_2))·sin(θ_4) - 190.5·sin(θ_2)·sin(θ_3) + 190.5·co
(sin(θ_2)·cos(θ_3) + sin(θ_3)·cos(θ_2))·sin(θ_4) - 190.5·sin(θ_2)·sin(θ_3) + 190.5·co
) + 70.6·cos(θ_1))·sin(θ_1) + (-328.93·(-sin(θ_2)·sin(θ_3)·cos(θ_1) + cos(θ_1)·cos(θ
                       sin(θ_1)
                      -cos(θ_1)
                          0
s(θ_2)·cos(θ_3) + 190.5·cos(θ_2))·cos(θ_1)
s(θ_2)·cos(θ_3) + 190.5·cos(θ_2))·sin(θ_1)
_2)·cos(θ_3))·sin(θ_4) - 328.93·(sin(θ_2)·cos(θ_1)·cos(θ_3) + sin(θ_3)·cos(θ_1)·cos(θ_2





))·cos(θ_4) - 70.6·sin(θ_1) - 190.5·sin(θ_2)·cos(θ_1)·cos(θ_3) - 190.5·sin(θ_2)·cos(
```

```
 θ_1) - 190.5·sin(θ_3)·cos(θ_1)·cos(θ_2))·cos(θ_1)  (-328.93·(-sin(θ_1)·sin(θ_2)·sin(θ




 _3) + sin(θ_1)·cos(θ_2)·cos(θ_3))·sin(θ_4) - 328.93·(sin(θ_1)·sin(θ_2)·cos(θ_3) + sin(
```

```
                    -(-328.93⋅(sin(θ₂)⋅sin(θ₃) - cos(θ₂)⋅cos(θ₃))⋅cos(θ₄) - 3
                    -(-328.93⋅(sin(θ₂)⋅sin(θ₃) - cos(θ₂)⋅cos(θ₃))⋅cos(θ₄) - 3
θ₁)⋅sin(θ₃)⋅cos(θ₂))⋅cos(θ₄) - 190.5⋅sin(θ₁)⋅sin(θ₂)⋅cos(θ₃) - 190.5⋅sin(θ₁)⋅s



28.93⋅(sin(θ₂)⋅cos(θ₃) + sin(θ₃)⋅cos(θ₂))⋅sin(θ₄) - 190.5⋅sin(θ₂)⋅sin(θ₃) + 19
28.93⋅(sin(θ₂)⋅cos(θ₃) + sin(θ₃)⋅cos(θ₂))⋅sin(θ₄) - 190.5⋅sin(θ₂)⋅sin(θ₃) + 19
in(θ₃)⋅cos(θ₂))⋅sin(θ₁) + (-328.93⋅(-sin(θ₂)⋅sin(θ₃)⋅cos(θ₁) + cos(θ₁)⋅cos(θ₂)
                    sin(θ₁)
                   -cos(θ₁)
                       0
0.5⋅cos(θ₂)⋅cos(θ₃))⋅cos(θ₁)
0.5⋅cos(θ₂)⋅cos(θ₃))⋅sin(θ₁)
⋅cos(θ₃))⋅sin(θ₄) - 328.93⋅(sin(θ₂)⋅cos(θ₁)⋅cos(θ₃) + sin(θ₃)⋅cos(θ₁)⋅cos(θ₂))
```


```
                    -(-328.93⋅(sin(θ₂)⋅sin(θ₃) - cos(θ₂)⋅cos(θ₃))⋅cos(θ₄) - 3
                    -(-328.93⋅(sin(θ₂)⋅sin(θ₃) - cos(θ₂)⋅cos(θ₃))⋅cos(θ₄) - 3
θ₁)⋅sin(θ₃)⋅cos(θ₂))⋅cos(θ₄) - 190.5⋅sin(θ₁)⋅sin(θ₂)⋅cos(θ₃) - 190.5⋅sin(θ₁)⋅s



28.93⋅(sin(θ₂)⋅cos(θ₃) + sin(θ₃)⋅cos(θ₂))⋅sin(θ₄) - 190.5⋅sin(θ₂)⋅sin(θ₃) + 19
28.93⋅(sin(θ₂)⋅cos(θ₃) + sin(θ₃)⋅cos(θ₂))⋅sin(θ₄) - 190.5⋅sin(θ₂)⋅sin(θ₃) + 19
in(θ₃)⋅cos(θ₂))⋅sin(θ₁) + (-328.93⋅(-sin(θ₂)⋅sin(θ₃)⋅cos(θ₁) + cos(θ₁)⋅cos(θ₂)
                    sin(θ₁)
                   -cos(θ₁)
                       0
0.5⋅cos(θ₂)⋅cos(θ₃))⋅cos(θ₁)
0.5⋅cos(θ₂)⋅cos(θ₃))⋅sin(θ₁)
⋅cos(θ₃))⋅sin(θ₄) - 328.93⋅(sin(θ₂)⋅cos(θ₁)⋅cos(θ₃) + sin(θ₃)⋅cos(θ₁)⋅cos(θ₂))
```


```
93⋅(sin(θ₂)⋅cos(θ₃) + sin(θ₃)⋅cos(θ₂))⋅sin(θ₄))⋅cos(θ₁)
93⋅(sin(θ₂)⋅cos(θ₃) + sin(θ₃)⋅cos(θ₂))⋅sin(θ₄))⋅sin(θ₁)
-328.93⋅(-sin(θ₂)⋅sin(θ₃)⋅cos(θ₁) + cos(θ₁)⋅cos(θ₂)⋅cos(θ₃))⋅sin(θ₄) - 328.93⋅
)
)



(sin(θ₂)⋅cos(θ₁)⋅cos(θ₃) + sin(θ₃)⋅cos(θ₁)⋅cos(θ₂))⋅cos(θ₄))⋅cos(θ₁)│
```

Fig: 5.2: Jacobian Matrix

CHAPTER – 6: KINEMATICS VALIDATION

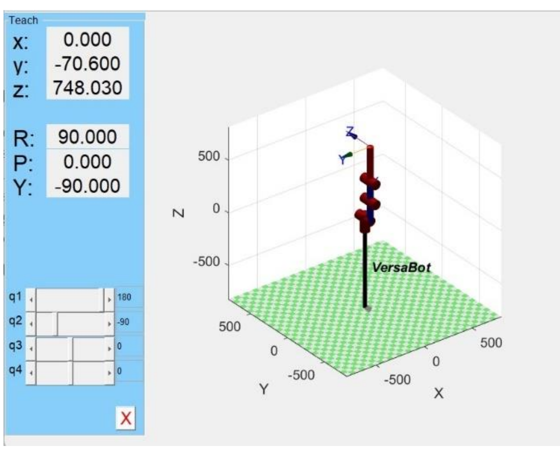
6.1. Forward Kinematics Validation

Forward Kinematics is validated both theoretically and using MATLAB software included with Peter Corke toolbox [5].

- To validate the forward kinematics, we use three different orientations using different θ_i values.
- Altering the D-H table, joint angle values in MATLAB will show the model using modified parameters.
- The transformation matrices are also calculated using the algorithm.

Case 1:

Transformations	θ_i	α_i	a_i	d_i
0 -> 1	180°	90°	0	38.1 mm
1 -> 2	-90°	0	-190.5 mm	-70.60 mm
2 -> 3	0	0	-190.5 mm	0
3 -> 4	0	0	-328.93 mm	0



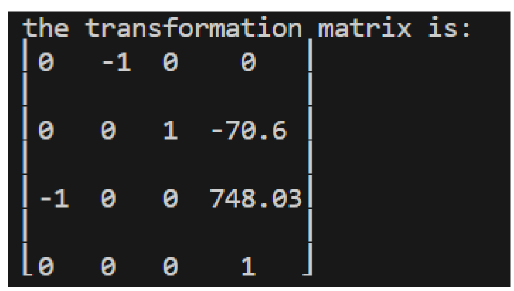


Fig 6.1: Forward Kinematics Validation – Case 1

Case 2:

Transformations	θ_i	α_i	a_i	d_i
0 -> 1	0	90°	0	38.1 mm
1 -> 2	-90°	0	-190.5 mm	-70.60 mm
2 -> 3	0	0	-190.5 mm	0
3 -> 4	90°	0	-328.93 mm	0

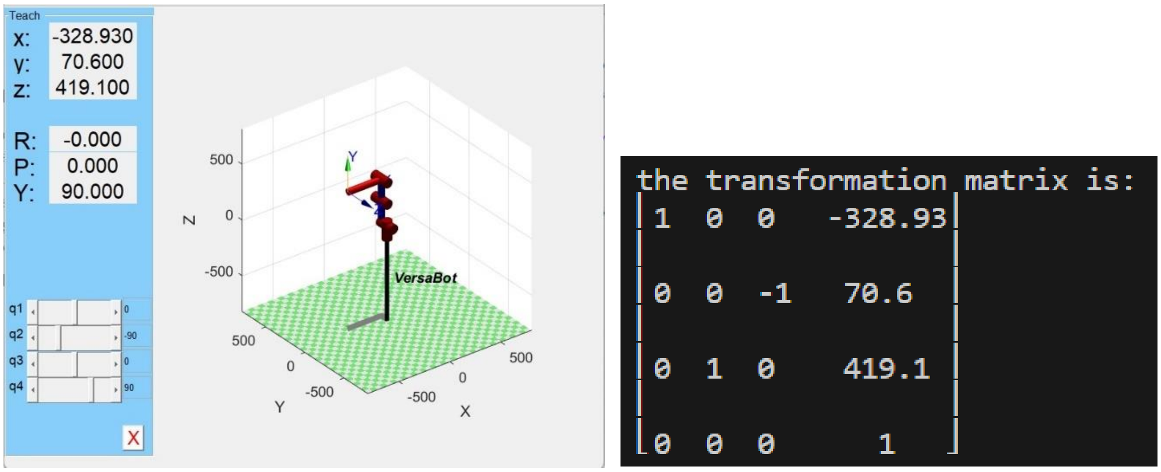


Fig 6.2: Forward Kinematics Validation – Case 2

Case 3:

Transformations	θ_i	α_i	a_i	d_i
0 -> 1	**0**	**90°**	**0**	**38.1 mm**
1 -> 2	**-90°**	**0**	**-190.5 mm**	**-70.60 mm**
2 -> 3	**90°**	**0**	**-190.5 mm**	**0**
3 -> 4	**0**	**0**	**-328.93 mm**	**0**

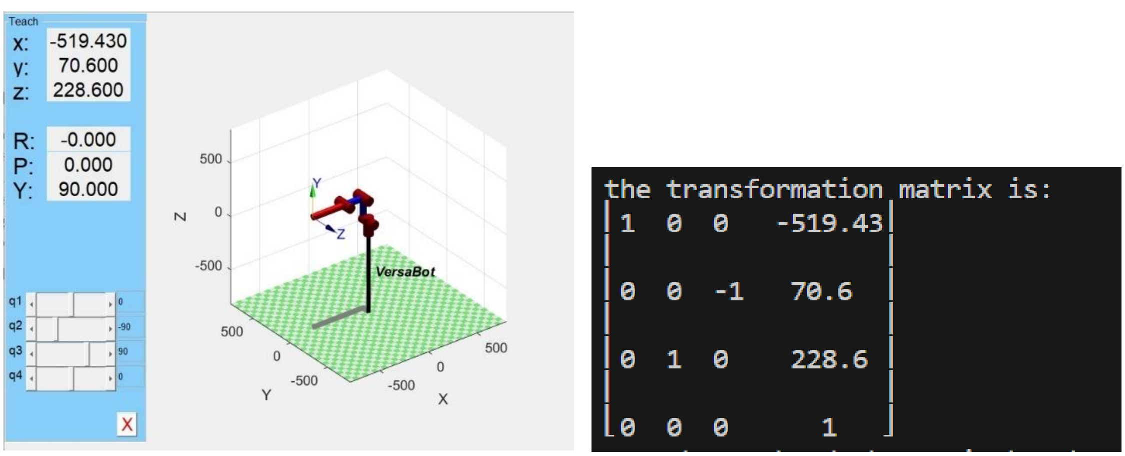


Fig 6.3: Forward Kinematics Validation – Case 3

6.2. Inverse Kinematics Validation

Inverse Kinematics is validated using the algorithm used in Homework 5. The steps followed are mentioned below.

- The end-effector trajectory is a circle with a 40-mm radius by the end effector. This trajectory is validated using the Jacobian Matrix that is calculated using the final transformation matrix.
- The Jacobian Inverse is found and equations for the circle trajectory are defined.
- Inverse velocity kinematics are found and finally the numerical integration is iterated to get the desired trajectory.
- The inverse kinematics is validated using the plotted trajectory.

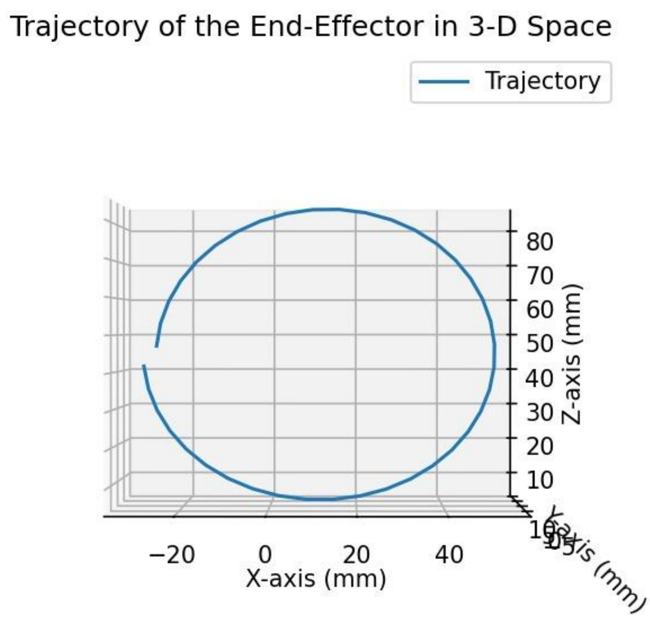


Fig 6.4: Trajectory of the End-effector in 3-D Space

CHAPTER – 7: WORKSPACE STUDY

The workspace of the manipulator arm above the mobile base is identified, and shown below and the below figure shows that the manipulator can rotate in 360^{o} space.

The arm with revolute joints can reach inside the space without any disturbances. As seen in the image the Arm base can rotate up to 0^{o} to 360^{o}. The end effector can reach up to 300 mm in all the X, Y and Z axis.

The workspace is of manipulator below is plotted while the mobile base in stationary position.

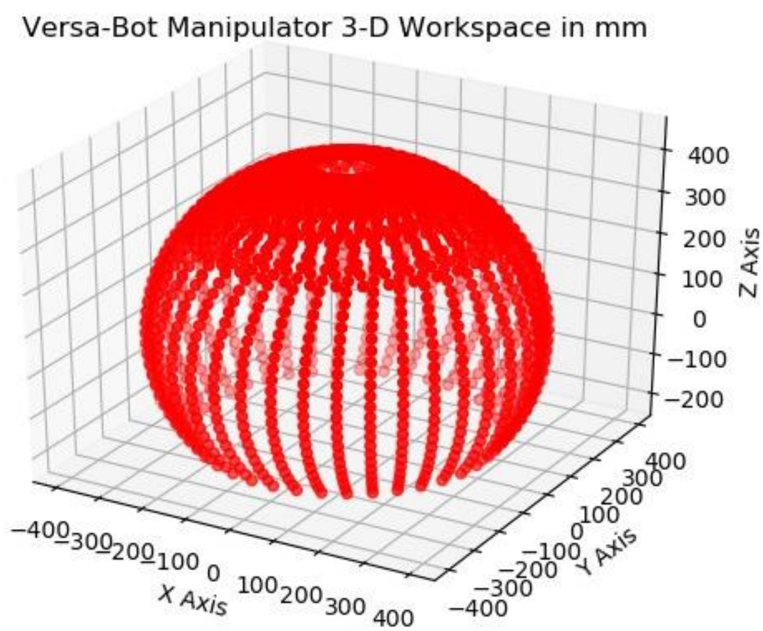


Fig 7.1: Workspace of the manipulator of the robot

CHAPTER – 8: ASSUMPTIONS

There are some assumptions made for this prototype with the comparison of real world model, those are,

- The model is a small-scale version of the real-world model.
- All the links and joints are not modelled to match the appearance of the actual robot.
- The open loop controller used here is one way to control the robot out of many ways.
- The vacuum gripper attracts the object, but not as attracted in the real world.
- The robot is now only equipped with LIDAR sensor, but the original version can hold many sensors.
- The D-H parameters are taken for the manipulator assuming the mobile base is stationary while the operation of manipulator arm.
- The third link is overall is considered to be the end effector which is fixed with the vacuum gripper.
- The weight is distributed evenly in the links.
- The weight of the object is assumed to be very light.

CHAPTER – 9: SIMULATIONS

9.1. Gazebo & RViz Visualization

The URDF is exported from the SolidWorks software, it is then moved in Ubuntu to create a ROS 2 package with launch files for the empty world and The robot is spawned in the gazebo environment and RViz is used to visualize the output of LIDAR sensor on the robot detecting the obstacles in front of the robot.

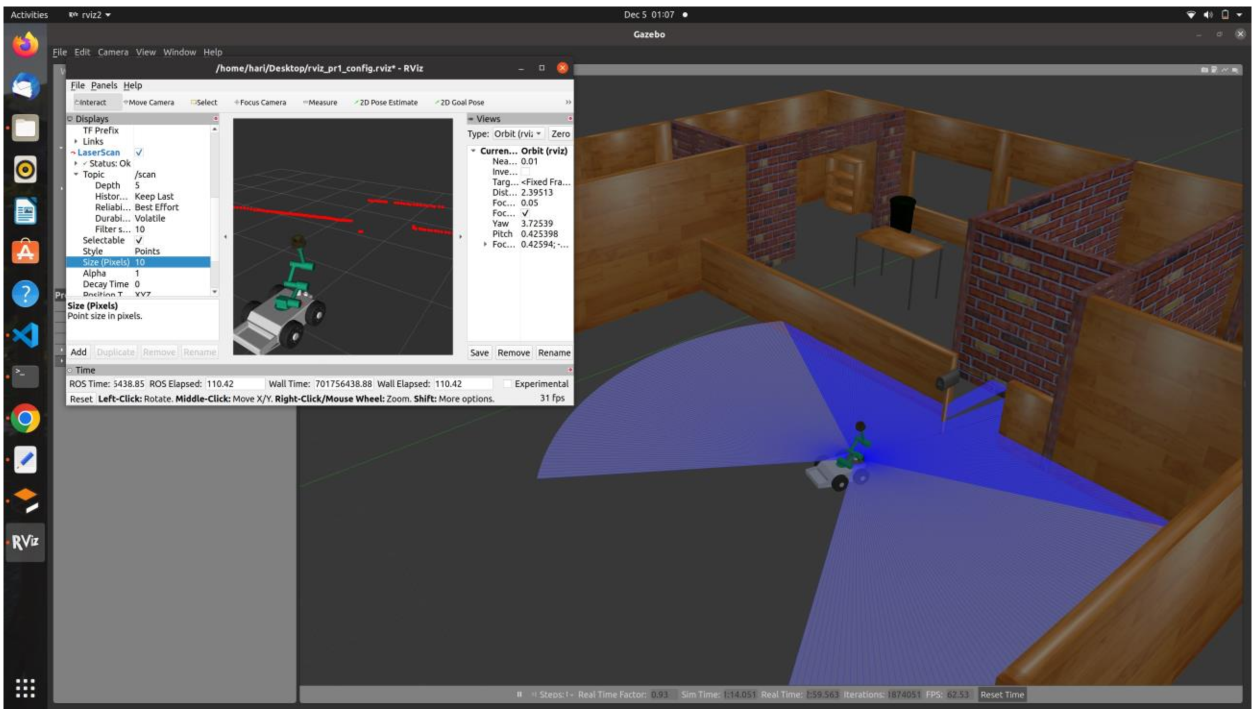


Fig 9.1: Gazebo & RViz Visualization of the robot

Tele-operation of the robot:

- The algorithm is framed to control the joint angles of the arm by and the mobile base is controlled by giving the linear velocities for the wheels when clicked the predefined button on the keyboard which will be displayed on the terminal once Tele-operation is launched in terminal alongside the Gazebo.
- The video of the Tele-operation of the robot:

[Tele-operation of the robot video]

9.2. Control Method- Open-loop Controller

The control method used to control the robot is an open loop controller. The open-loop controller is used in the gazebo environment to pick a can with the vacuum gripper, which is the end-effector of the mobile manipulator. The control flow can be explained by,

- For the first five seconds, the robot is programmed to move the robot at a speed of 3m/s, after that, it will accelerate to 6 m/s for the next three seconds. After this linear movement, the robot arm articulates in a controlled way to carry out a predefined joint angle trajectory. The trajectory illustrates a set of motions by following the path of an arc. The robotic arm is stopped by a time function after exactly two seconds to guarantee smooth and controlled operation. The vacuum gripper engages the mechanism to grasp a coke can as soon as the arm stops moving.

- The robot then goes into reverse motion, traveling 3 m/s for the first 5 seconds and then reaching a speed of 6 m/s for the last 3 seconds of this phase. The vacuum gripper holds onto the Coke can while it moves back in reverse. The gripping action is stopped when the gripper disengages after it completes reversing. This series of regulated motions, which combine articulated manipulation and linear motion, demonstrates robotic operation intended to complete a pick and place task accurately and efficiently.

- **The video of the simulation of the robot using open-loop controller:**

[Open-Loop Control Video]

Click the link above to view the video.

CHAPTER – 10: PROBLEMS FACED

There are certain problems faced during the modelling, package creation and spawning of the robot in gazebo and some problems while controlling the robot using ROS 2.

- During the modelling the scaling of every part after assembly of those parts when some dimensions are not matched the assembled model was tough.

- Once the modelling is done, while exporting to URDF version of the model due to incorrect axis assignment to the joints and links, the robot was not spawning in the gazebo environment.
- Sometimes, after running the ROS 2 packages of project was not working, due to forgetting using 'colcon build' the project after making a change in the root package.

- Since the system was in reset condition, lately realized that the XACRO package was not installed which was throwing some errors while running.

- The vacuum gripper plugin was not working initially and got the help from the teaching assistants to make it work, after reducing the weight of the object is reduced it started working.

CHAPTER – 11: LESSONS LEARNED

This project helped us to try out the topics we learned in the class and learn new things out of the course while the process of modelling, validating the kinematics and controlling of the robot.

- While modelling the robot, we learnt how the industrially built manipulator looks like and how it can be modelled in SolidWorks like those arms.

- While exporting the model into URDF we learnt how to assign the axis to every joint to reduce errors while using it with ROS 2 packages.

- During framing the D-H table, we used the procedure used in previous homework to a different robot made us to get more clearer in the concept.

- We learnt how to frame algorithms to validate the forward and inverse kinematics of any robot which could be the key learning of this project.

- Also learnt about framing an algorithm to control the robot using tele-operation like project one but also altered the algorithm to control the joint angles of the robot.

CHAPTER – 12: CONCLUSION

This project is completed with the successfully modeling and controlling the Versa-BOT, a mobile manipulator in the gazebo environment in picking up an object from one place and put it in another place with the help of algorithm framed using the frame assignment to the robot, D-H table, validating the kinematics. Every single process and step involved in this whole project explained in detail and showed validations and simulations of kinematics which was done using Peter Corke toolbox and python algorithm script involving in computing the final transformation matrix, Jacobian matrix, inverse Jacobian, inverse velocity kinematics.

Furthermore, the assumptions made and used on the robot were listed in the report. Every simulation done in the gazebo environment using tele-operation and Open-loop controller is attached to the report as a link. This project helped us to build a strong foundation and understanding of what we learnt during the course. The project which built using the SolidWorks, ROS2, Gazebo and RViz helped us to gain exposure in these tools to work on future projects.

12.1. Future Work

Introducing a groundbreaking retail concept: Integrating Versa-BOT V1.0 with VR technology to provide a unique, immersive in-store shopping experience from home. This innovative system offers unparalleled convenience and accessibility, revolutionizing the way customers interact with our products and services.

The mobile manipulator recreates every hand and mobile operations of humans in future to do not only the shopping but other complicated and hazardous tasks in the machine industry level which could be nearly lethal to humans and to make the humans work in their comfort [6].

REFERENCES

[1] Robot dynamics and control - Book by Mark W. Spong

[2] Robotnik – Mobile manipulators - Mobile Manipulators Robots - Robot Manipulator | Robotnik®

[3] Forward kinematics by Ros robotic learning - rosroboticslearning

[4] Introducing: Accessible Mobile Manipulation for Warehouse and Factory Automation Productive robotics

[5] Peter Corke- toolbox for kinematics validation - Robot arm kinematics

[6] VRR- Virtual Reality Red New Virtual Reality Application: Supermarket VR