# Table of Contents

**Abstract** ........ 2
1. **Introduction** ........ 3
   1.1. **Motivation** ........ 3
   1.2. **Challenge** ........ 3
2. **ROS Installation** ........ 3
   2.1. **Simulator** ........ 3
3. **Methodology** ........ 4
   3.1. **Creation of manipulator** ........ 4
   3.2. **Object detection** ........ 5
   3.3. **Placement of Object** ........ 5
4. **Analysis** ........ 6
5. **Result** ........ 7
6. **Conclusion** ........ 7
7. **Reference** ........ 8

# Abstract

This paper will include the thesis for the development of a SCARA robot. The robot is to be developed an onboard vision-based system for the image processing used to recognize the position of an object and to pick and place application which would function in manual mode by taking inputs from a user for the placement of the object. If time permits it would be further developed to operate in automatic mode in which robot undertake repetitive tasks without the interference of human.

## 1. Introduction

The SCARA acronym stands for **Selective Compliant Assembly Robo**t or **Selective Compliant Articulate Robot**. Each of the links constitute to a single degree of freedom while an articulate robot should consist a four degrees of freedom which equates to four links from base to end-effector. These four links are usually termed as "shoulder", "elbow", "wrist" and "end-effector" so that its functionality mimics that of a human hand. We attach the camera at wrist joint to detect the existence of the object.

A simulator is an essential tool to have if you want to start playing with almost anything in reality it will save you a lot of time and money. If you plan to have a distributed team to work on project, it is almost impossible to have all team member with the same hardware and environment to work, and so it's a good tool to provide a platform of all team members to learn from the project under the same environment.

**Robot Operating System** (**ROS)** is a collection of software frameworks for robot software development, although ROS is not an operating system. ROS is a great platform that provide its services for designer of a heterogeneous computer cluster such as hardware abstraction, low-level device control, implementation of commonly used functionality. One of the most popular applications for robotics simulators is for 3D modeling and rendering of a robot and its environment is **Gazebo**. Using a computer vision to knowing the distance between the robotic arm and the object, by using this method, data from the camera will be reconstructed to form the coordinates of object presence. The image coordinates are connected to the Robot Operating System (ROS), so the manipulator can navigate automatically detect object.

### 1.1. Motivation

The robot is a variation of articulated robot-arms that are widely used in industries for applications ranging from welding, painting, assembly and pick and place. The main reason for the popularity of industrial robots is the efficiency in doing repetitive tasks. Robots of this nature works repetitive and efficiently without any break that gives the industries a tremendous advantage our human labor work.

### 1.2. Challenge

In detection, autonomous picking requires robust recognition of the object and accurate measurement of the object movement with respect to the ground target. Difficult phase in this project is to create the SCARA robotic arm in gazebo simulator. Under current conditions the manipulator cannot be pick any object because it is automatic so it can pass the object without picking it up.

## 2. ROS Installation

I installed Ubuntu 20.04 as ROS Neotic is recommended, but it was not hard to follow the steps here. ROS was installed. I get a simulator based on Gazebo to enjoy its excellent integration with ROS.

### 2.1. Simulator

**Gazebo** is a 3D high-quality robot simulator that is well integrated into the Robot Operating System (ROS) as the interface for the robot and may be used for your project. With Gazebo, you are able to create a 3D scenario with a physical engine for illumination, gravity, inertia etc. on your computer with robots, obstacles and many other objects.

# 3. Methodology

## 3.1. Creation of manipulator

The first setup is to create the URDF file of the robotic arm for the visualization of the SCARA robot on **Gazebo Simulator.** After that we create the real time world environment in simulator my placing the conveyer and robotic arm on it. Add the controller on the manipulator and put a camera on the end-effector to detect the moving object.

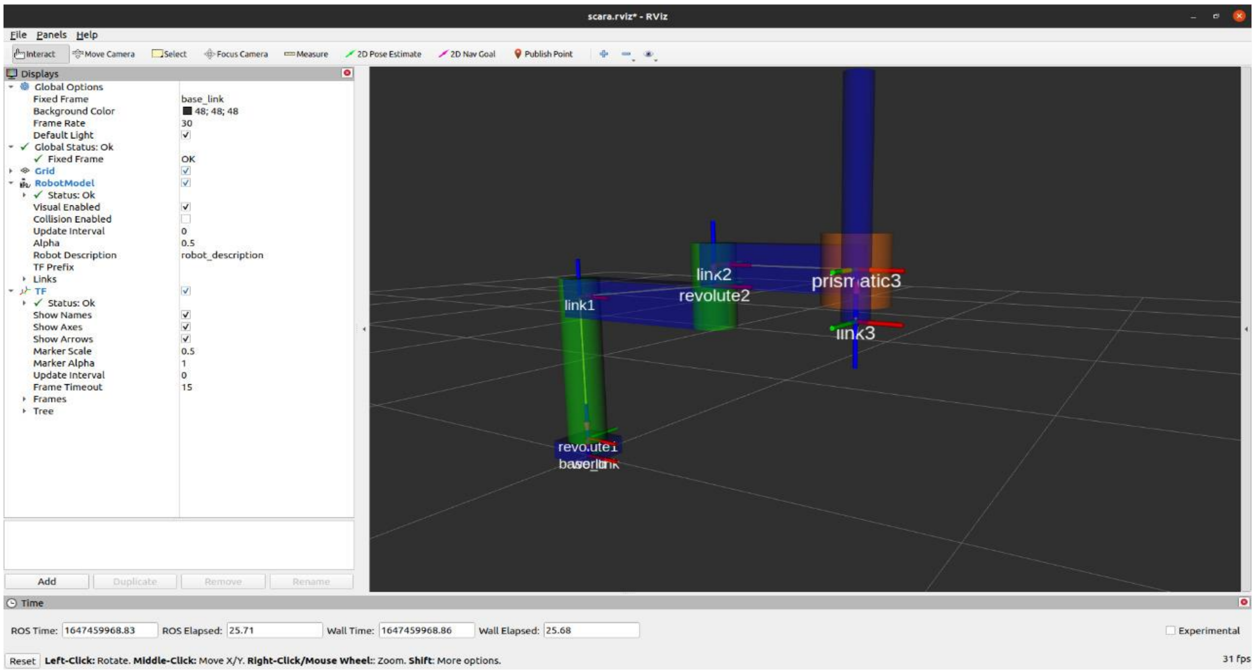



*Figure1: SCARA robotic arm in Rviz*

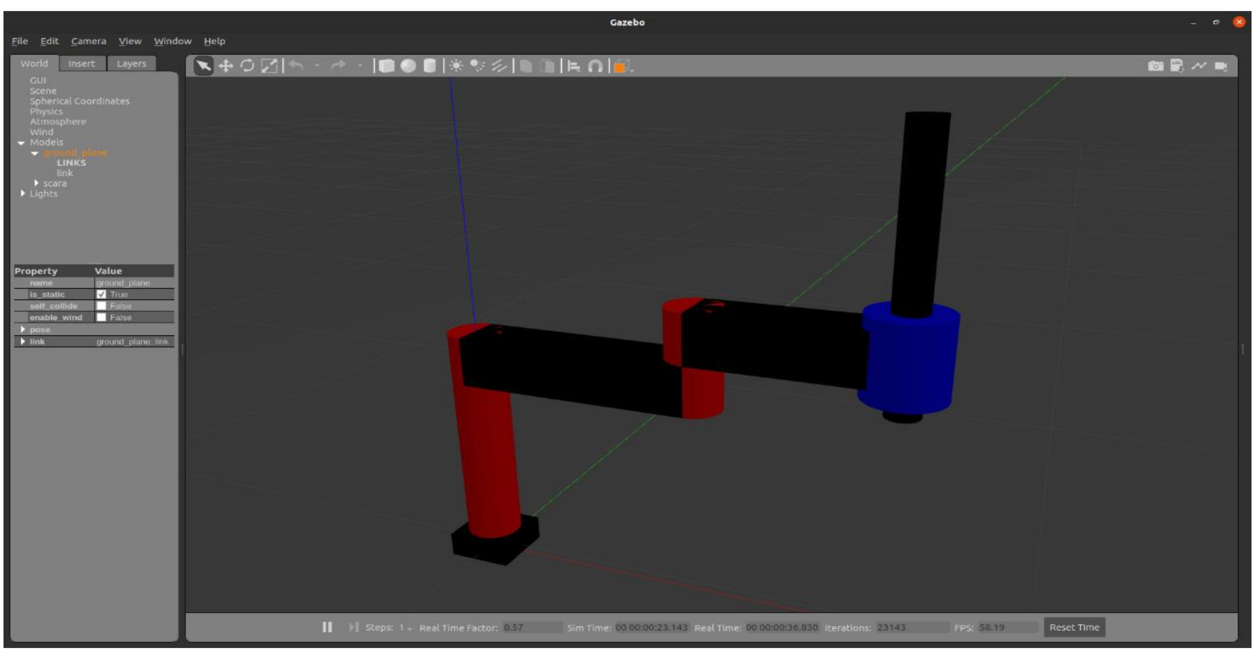



*Figure2: SCARA robotic arm in Gazebo Simulator*

### 3.2. Object detection

The second step to detect and decode an object, the image is converted to gray-scale and segmentation used to find the contiguous regions in the thresholder image with same color. Pick the object by detecting the exact location of object by using the gripper attached on the robotic arm.

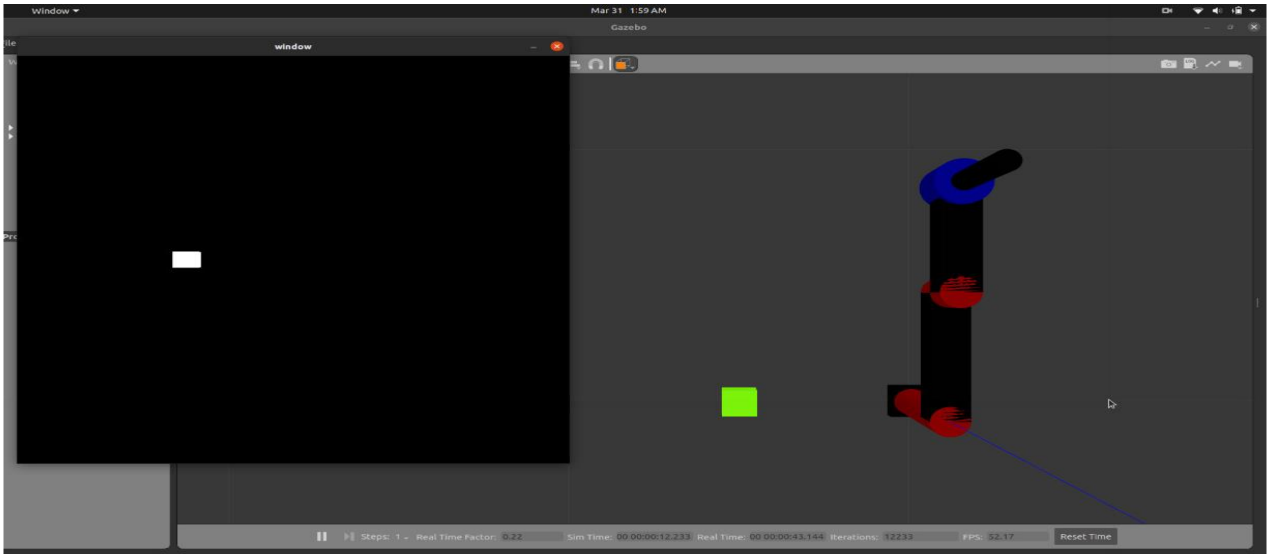


*Figure 3: Detect the box by camera*

### 3.3. Placement of Object

We give the coordinates of the object placement by using user manual interface. There are two ways to place an object on defined place by using either forward kinematics or inverse kinematics.

**Forward kinematics** refers to the utilization of the kinematic equations of a robot to compute the position of the end effector from user specified values. **Inverse kinematics** is a method that helps define the motion of a robot to reach a desired location by computing the joint angles from the input user.

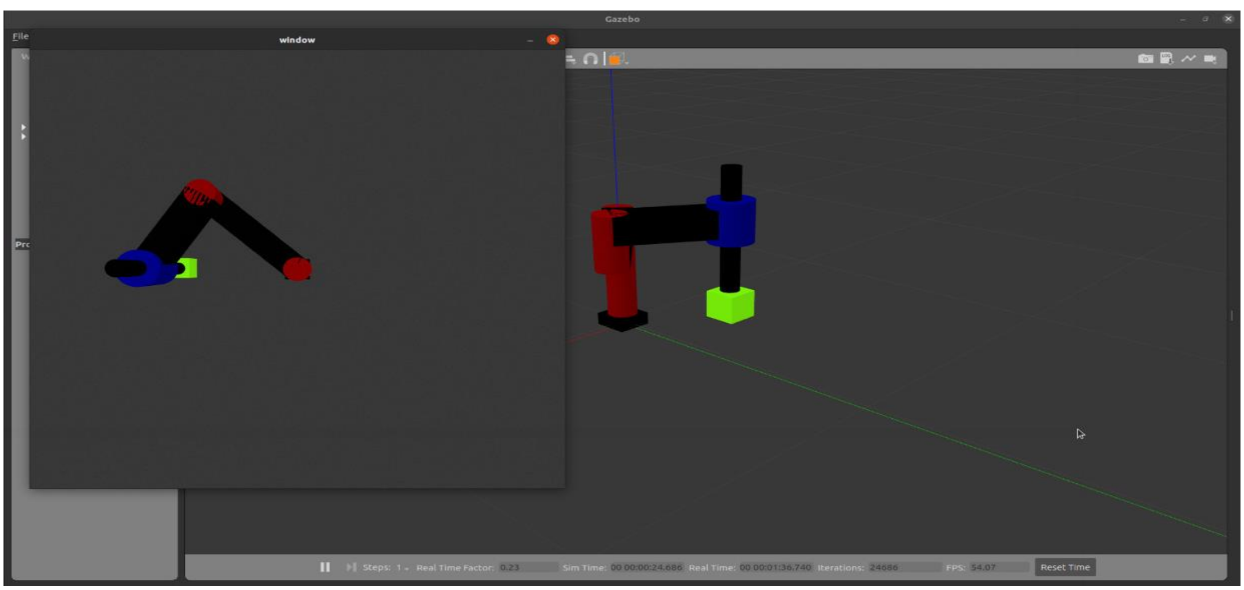
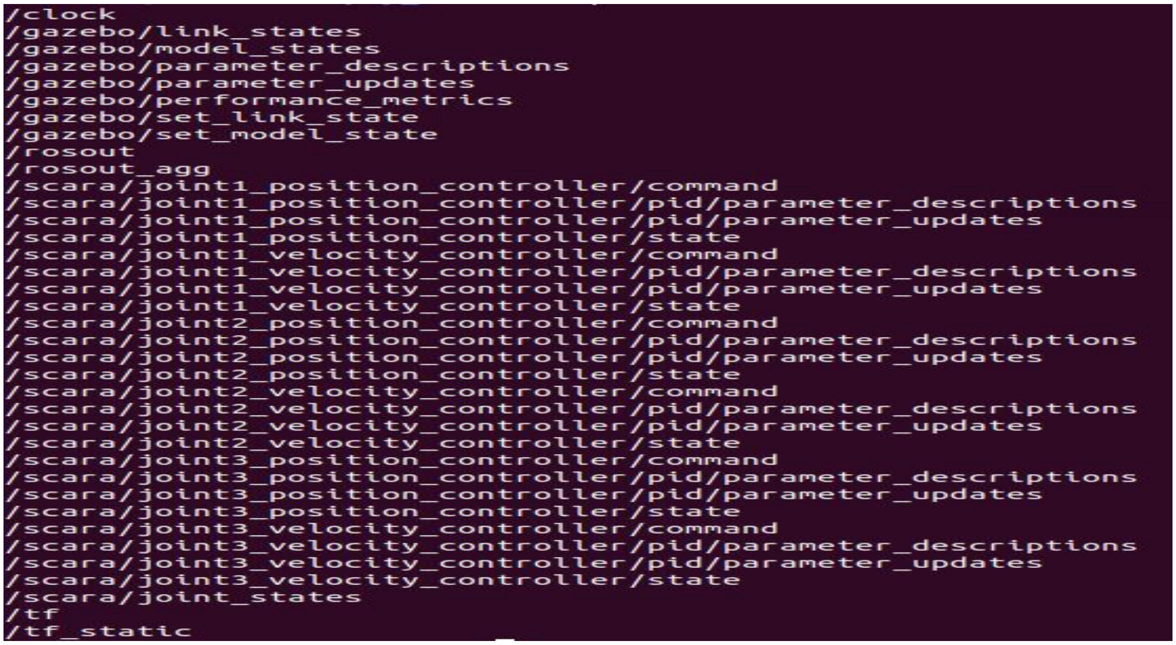


*Figure 4: SCARA arm picking up the box*

## 4. Analysis

To get all the information about the SCARA robot, we debug the ROS topics, ROS messages, publisher and subscriber and publisher rate information by using rostopic list command line. This information help us to find the information about the current state of the robotic arm links and also get camera topic info. Here is the list of all the topic robotic arm contained



```
/clock
/gazebo/link_states
/gazebo/model_states
/gazebo/parameter_descriptions
/gazebo/parameter_updates
/gazebo/performance_metrics
/gazebo/set_link_state
/gazebo/set_model_state
/rosout
/rosout_agg
/scara/joint1_position_controller/command
/scara/joint1_position_controller/pid/parameter_descriptions
/scara/joint1_position_controller/pid/parameter_updates
/scara/joint1_position_controller/state
/scara/joint1_velocity_controller/command
/scara/joint1_velocity_controller/pid/parameter_descriptions
/scara/joint1_velocity_controller/pid/parameter_updates
/scara/joint1_velocity_controller/state
/scara/joint2_position_controller/command
/scara/joint2_position_controller/pid/parameter_descriptions
/scara/joint2_position_controller/pid/parameter_updates
/scara/joint2_position_controller/state
/scara/joint2_velocity_controller/command
/scara/joint2_velocity_controller/pid/parameter_descriptions
/scara/joint2_velocity_controller/pid/parameter_updates
/scara/joint2_velocity_controller/state
/scara/joint3_position_controller/command
/scara/joint3_position_controller/pid/parameter_descriptions
/scara/joint3_position_controller/pid/parameter_updates
/scara/joint3_position_controller/state
/scara/joint3_velocity_controller/command
/scara/joint3_velocity_controller/pid/parameter_descriptions
/scara/joint3_velocity_controller/pid/parameter_updates
/scara/joint3_velocity_controller/state
/scara/joint_states
/tf
/tf_static
```

*Figure 5: rostopic list*

We all get the node information we use rqt_graph, this tool is very useful to see what's going on in your ROS. It provides a GUI plugin for visualizing the ROS message that publishes on different ROS nodes. Its components are made generic with other packages.

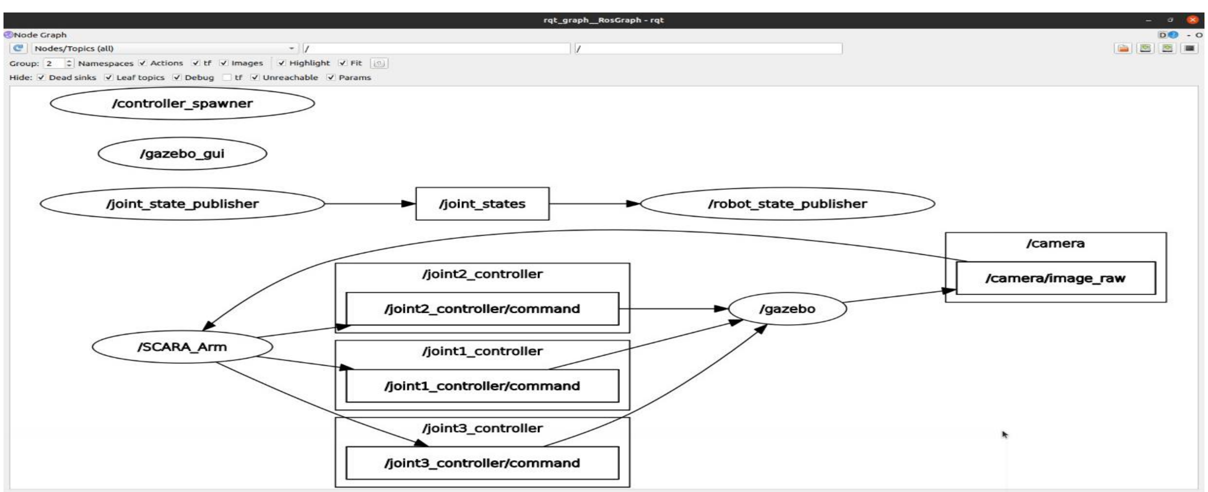



*Figure 6: rqt_graph*

## 5. Result

After the SCARA-manipulator was successfully tuned, it was tested thoroughly for position-control. This was achieved either by sending two theta values θ1 and θ2 and a distance value d3 for the three links or giving the coordinates of the placement point through the terminal user input window. The code was also modified so that once the links reach their final position the current position of each of the three links was printed on the screen.

## 6. Conclusion

The conclusion, a robotic arm is simulated with real time simulation world in gazebo simulator, an onboard vision system that consists of an image processor to visually and autonomously provide position and attitude estimates of the object on conveyer belt. Image processing in detecting object with 85% success rate at 50-200 cm distance, where there is a relation between distance and image processing capability in recognizing object.

In future work, we will focus to giving the coordinates without the interruption of user manual, robotic arm autonomously place the object in desired position. Since complex tasks will require more than one manipulator and more than one ground robot, the coordination and motion-planning is important to prevent collision between manipulators and to allocate the tasks.

## 7. Reference

[1] Kinematic Modeling and Simulation of a SCARA Robot by Using Solid Dynamics and Verification by MATLAB/Simulink, Mahdi Salman Alshamasin, Department of Mechatronics Faculty of Engineering Technology, Albalqa' Applied University, Jordan, Florin Ionescu Mechatronics Institute Hochschule – Konstanz, Htwg, Germany, Riad Taha Al-Kasasbeh Department of Power Engineering Faculty of Engineering Technology, Albalqa' Applied University, Jordan, 2009

[2] SCARA Robot: Modeled, Simulated, and Virtual-Reality Verified, Yousif I. Al Mashhadany (MIEEE, MIIE), Electrical Engineering Department, College of Engineering, University of Anbar, Baghdad, Iraq,2012

[3] Trajectory Planning Design Equations and Control of a 4 - axes Stationary Robotic Arm T.C. Manjunath, Student Member IEEE, SPIE, IOP, Life Member ISSS, ISOI, SSI, ISTE, 2007 [4] Mathematical modelling, simulation and experimental verification of a Scara robot M. Taylan Das, L. Canan Du¨ lger, Department of Mechanical Engineering, University of Gaziantep, 27310 Gaziantep, Turkey,2004