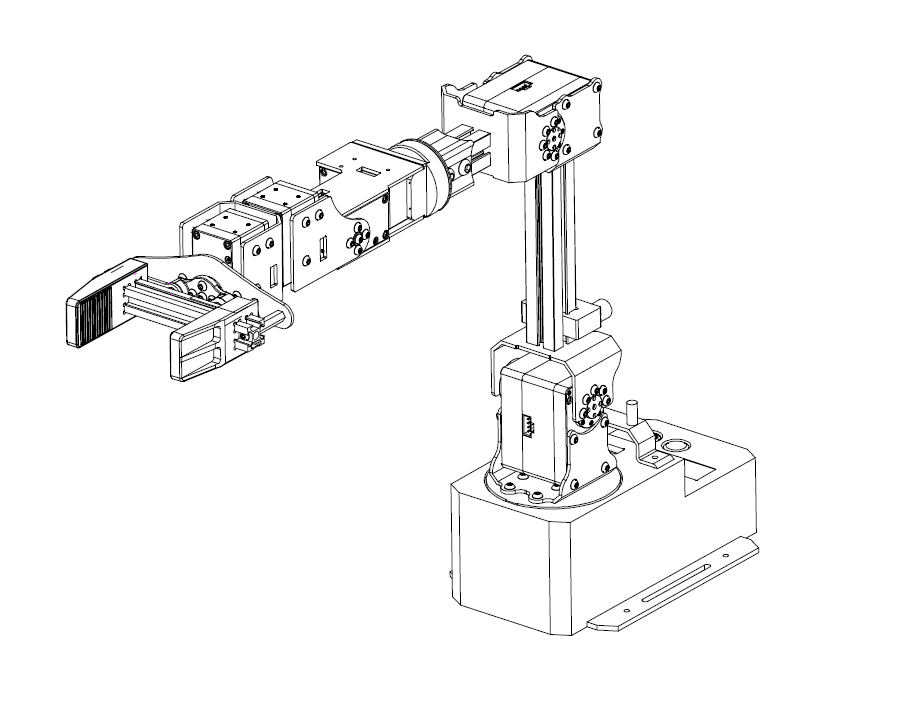
**K1 Robotic Arm**

**User Manual**

V0.4.1





**Catagory**

[1. Safety 4](#_Toc104450808)

[1.1 Precautions 4](#_Toc104450809)

[1.2 General Cautions 4](#_Toc104450810)

[1.3 Emergency stop 5](#_Toc104450811)

[2. Product Introduction 6](#_Toc104450812)

[2.1 General 6](#_Toc104450813)

[2.2 Specifications 6](#_Toc104450814)

[2.3 External Structure 7](#_Toc104450815)

[3. Quick Start 8](#_Toc104450816)

[3.1 Cable Connection 8](#_Toc104450817)

[3.2 Operating Environment 8](#_Toc104450818)

[3.2.1 Environmental Requirements 8](#_Toc104450819)

[3.2.2 Button Introduction 8](#_Toc104450820)

[# 1 Power Button 8](#_Toc104450821)

[# 2 DOWN Button (Free/Lock) 9](#_Toc104450822)

[# 3 UP Button (Enable/Disable) 9](#_Toc104450823)

[# 4 Menu Button 9](#_Toc104450824)

[# 5 Display Area 9](#_Toc104450825)

[4. Operation Instruction 11](#_Toc104450826)

[4.1 SDK Version (non-ROS version) 11](#_Toc104450827)

[4.1.1 Source Code Download and Dependent Libraries Installation 11](#_Toc104450828)

[4.2 SDK Version (ROS version) 12](#_Toc104450829)

[4.2.1 Source Code Download and Dependent Libraries Installation 12](#_Toc104450830)

[5. Demos Running 14](#_Toc104450831)

[5.1 Swap Blocks 14](#_Toc104450832)

[5.2 Action Repetition 14](#_Toc104450833)

[5.3 Visual Capture (Visual Perception Suite is required to purchase) 14](#_Toc104450834)

[5.3.1 Installation 15](#_Toc104450835)

[5.3.2 Calibration 15](#_Toc104450836)

# 1. Safety

This chapter contains important safety information, which must be read carefully before using this robotic arm for the first time. The installers and operators should read the manual carefully and in strict accordance with it.

# 1.1 Precautions

* Please make sure there is enough space near the robot arm when using.
* When the robot arm is working, it is strictly prohibited to enter its range of activity.
* Please use the official standard power adapter.

# 1.2 General Cautions

* If the product is faulty, please contact the after-sales service in time.
* If the product is disposed of, please comply with the relevant laws to properly dispose of industrial waste and protect the environment.
* The product packing box contains small spare parts. Please do not let children play with them to prevent them from being swallowed by mistake.
* This product should only be used by professionals or under the guidance of a relevant instructor. Turn off the equipment in time when the operation is finished
* Do not put your hand into the safety range of the product when it is being operated. Be careful of bruising and pinching.
* Please be careful when handling and installing the arm, and follow the instructions on the box to place the robot gently and correctly in the direction of the arrow, otherwise the machine will be easily damaged.
* Please read this manual carefully before operating the robotic arm.

# 1.3 Emergency stop

* Press the power button on the base, and the arm will be powered off and it will enter the release state
* Please do not press the power button during the normal operation, otherwise the motion trajectory before stopping will be different from that in the normal situation, which may cause unexpected situations such as collision.

# Product Introduction

# 2.1 General

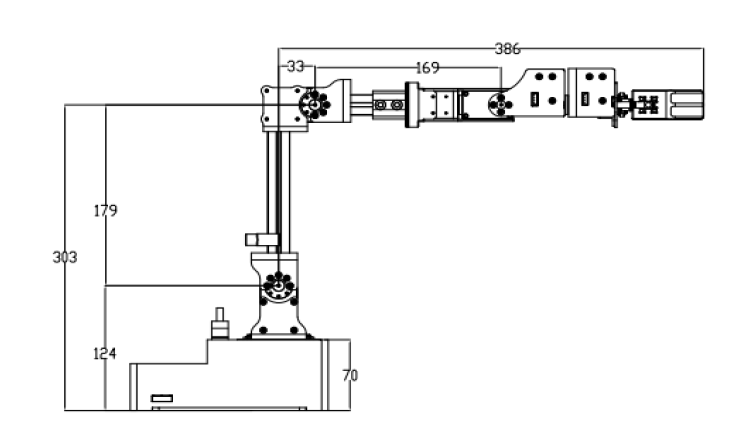
K1 six-axis robotic arm is a robotic arm device for ROS MoveIt ！teaching. It consists of iwo 85kg and five 45kg servos. It realizes 6 degrees of freedom + 1 end control.

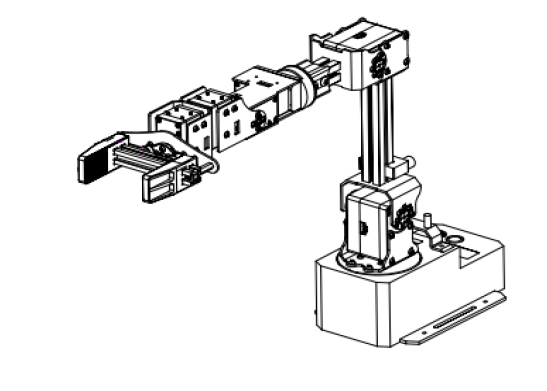
# 2.2 Specifications

|  |  |
| --- | --- |
| **K1 Robotic Arm Specifications** | |
| **Contents** | **Parameters** |
| Base Dimensions | 180mm（L)\*120mm(W)\*70mm(H) |
| Horizontal Extension Distance | 532mm |
| Repeatability Precision | ±0.5mm |
| Installation Method | Screw fixation |
| Servo specifications | 5 \* 45kg + 2 \* 85kg |
| Degree of Freedom | 6+1 |
| End Clamp | Parallel gripper (replaceable) |
| Operating System | Ubuntu + ROS / Linux + SDK |
| Weight | 2.5KG |
| Payload | 1KG |
| Power Supply | 24V/5A |
| Communication Interface | USB type-C |

# 2.3 External Structure

The K1 robotic arm consists of a base, an arm and an end parallel gripper. Appearance as shown in the picture:





## 3. Quick Start

This chapter briefly introduces how to control the robotic arm through the MoveIt! (an open source robotic manipulation platform that allows you to develop complex manipulation applications using ROS) under the Ubuntu system.

# 3.1 Cable Connection

1. Connect the USB cable to the computer (or other Ubuntu-based host) with a USB cable
2. Connect the adapter and turn on the power switch (the switch light is on).

# 3.2 Operating Environment

Users can download the latest ROS codes of arm from official GitHub website (see Chapter 4 in details), Compile and run.

# 3.2.1 Environmental Requirements

System: Ubuntu 16.04 or above

ROS version: Kinetic or above

# 3.2.2 Button Introduction

# # 1 Power Button

When the switch is pressed, light is on.

# # 2 DOWN Button (Free/Lock)

In the Menu Mode, it is the Down button of the menu.

In the Non-Menu Mode, press this button, the servo can be released or locked. At that time, the status of ***torque*** in the display is ***“free”*** or ***“lock”***.

# # 3 UP Button (Enable/Disable)

In the Menu Mode, it is the Up button for the menu.

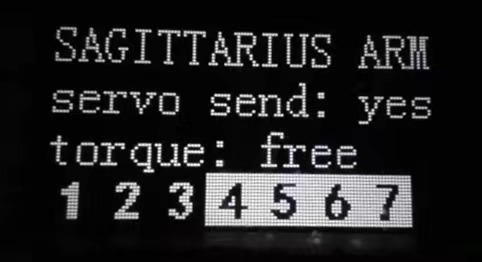
In the Non-Menu Mode, press this button, the control of servo can be allowed or forbidden. At that time, the status of ***servo send*** in this display is ***“yes”*** or ***“no”***.

# # 4 Menu Button

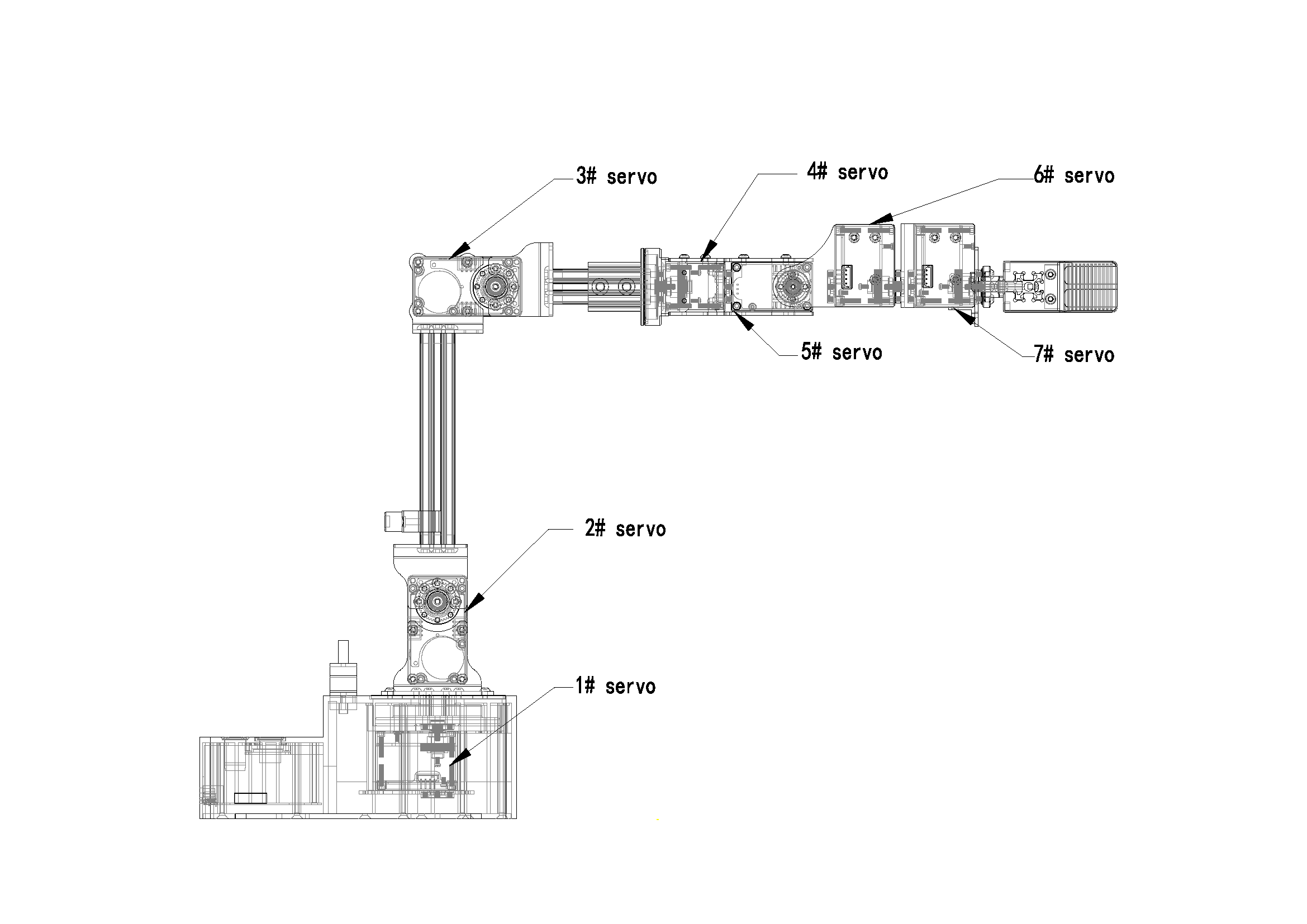
Press the Menu Button to enter the Menu Selection Mode. There are two options in the Menu: “RunDemo” and “Setting”. Press the #3 Up Button or #2 Down Button to select the option, then press the Menu Button again to enter the selected option.

# # 5 Display Area

There is some information displayed on Display Area as the following figure:



* The second line is the status of Servo Send.
* The third line is the status of torque.
* The last line is the communication status of each servo. The number represent the servos at different positions. White numbers with black background indicate that the communication of the servos is normal, while black numbers with white background indicate abnormal.



## 4. Operation Instruction

# 4.1 SDK Version (non-ROS version)

# 4.1.1 Source Code Download and Dependent Libraries Installation

(1) The address of the SDK source code for the K1 robotic arm is: <https://github.com/NXROBO/sagittarius_sdk.git.>

# cd ~

# git clone https://github.com/NXROBO/sagittarius\_sdk.git

(2) Eigen Library Installation

# sudo apt-get install libeigen3-dev

If you can't find Eigen/Dense when compiling, the reason is that Eigen3 has an extra layer of folders compared to the previous version.

The solution is a softlink to the previous directory. Find the installation directory of Eigen3 first.

# whereis eigen3

Assume that the installation directory of Eigen3 is“/usr/include/eigen3”

# cd /usr/include/

# sudo ln -s eigen3/Eigen Eigen

(3) Install the dynamic library of arm:

If your hardware platform is x86:

# cd sagittarius\_sdk

# sudo cp ./lib/x86\_64/libsagittarius\_sdk.so /usr/lib/

If your hardware platform is arm64:

# cd sagittarius\_sdk

# sudo cp ./lib/arm64/libsagittarius\_sdk.so /usr/lib/

(4) Compile

# g++ -I ./ -o sagittarius\_example sagittarius\_example.cpp -L ./ -lsagittarius\_sdk -lpthread -lboost\_system -lboost\_thread

(5) Run

# ./ sagittarius\_example

# 4.2 SDK Version (ROS version)

# 4.2.1 Source Code Download and Dependent Libraries Installation

* Install the Robot Operating System (ROS) first (if it has been installed, please skip this step). Follow the installation method on this page:

<http://wiki.ros.org/ROS/Installation>

* Install the K1 robotic arm SDK. The download address is: <https://github.com/NXROBO/sagittarius_ws.git.>
* Install the Dependent Libraries as follows:

(1) Open the terminal:

# cd ~/

# mkdir -p sagittarius\_ws/src

# git clone https://github.com/NXROBO/sagittarius\_ws.git

# cd ~/sagittarius\_ws/src/sagittarius\_arm\_ros/

# ./install.sh

(2) Enter the password follow the prompts and install the Dependent Libraries.

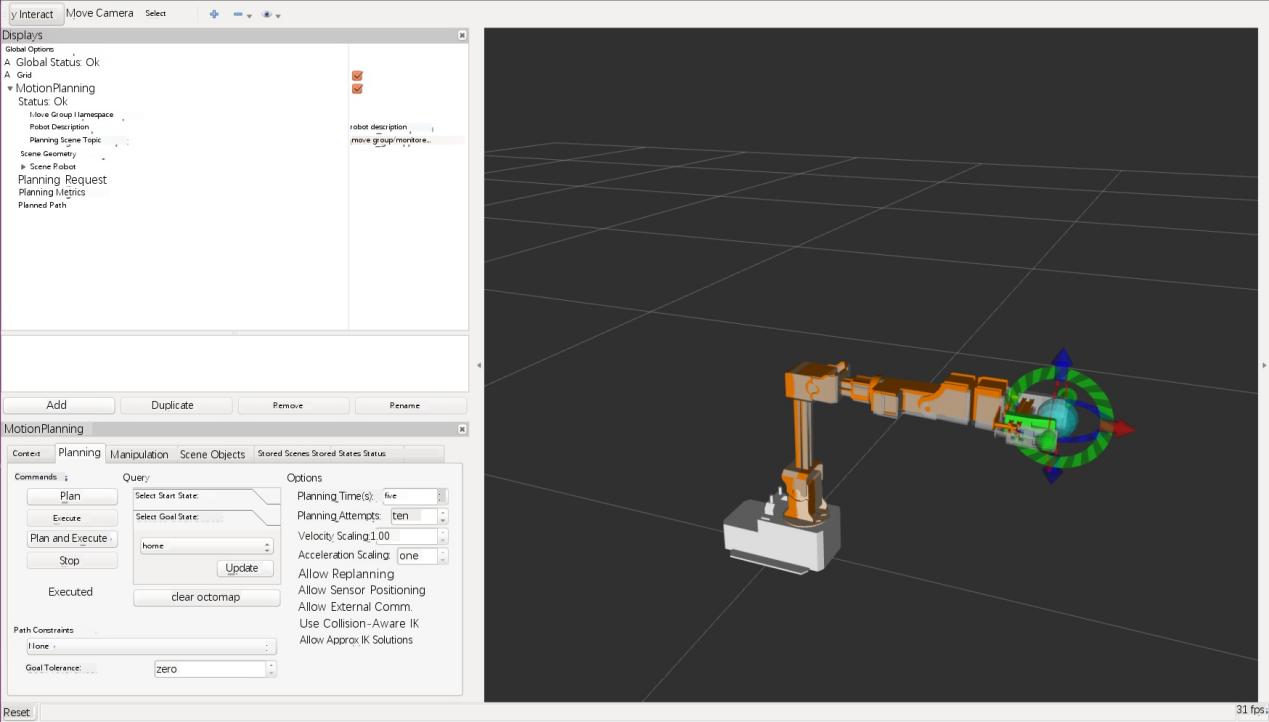
* Compile and run

# cd ~/sagittarius\_ws

# catkin\_make

# source devel/setup.bash

# roslaunch sagittarius\_moveit demo\_true.launch



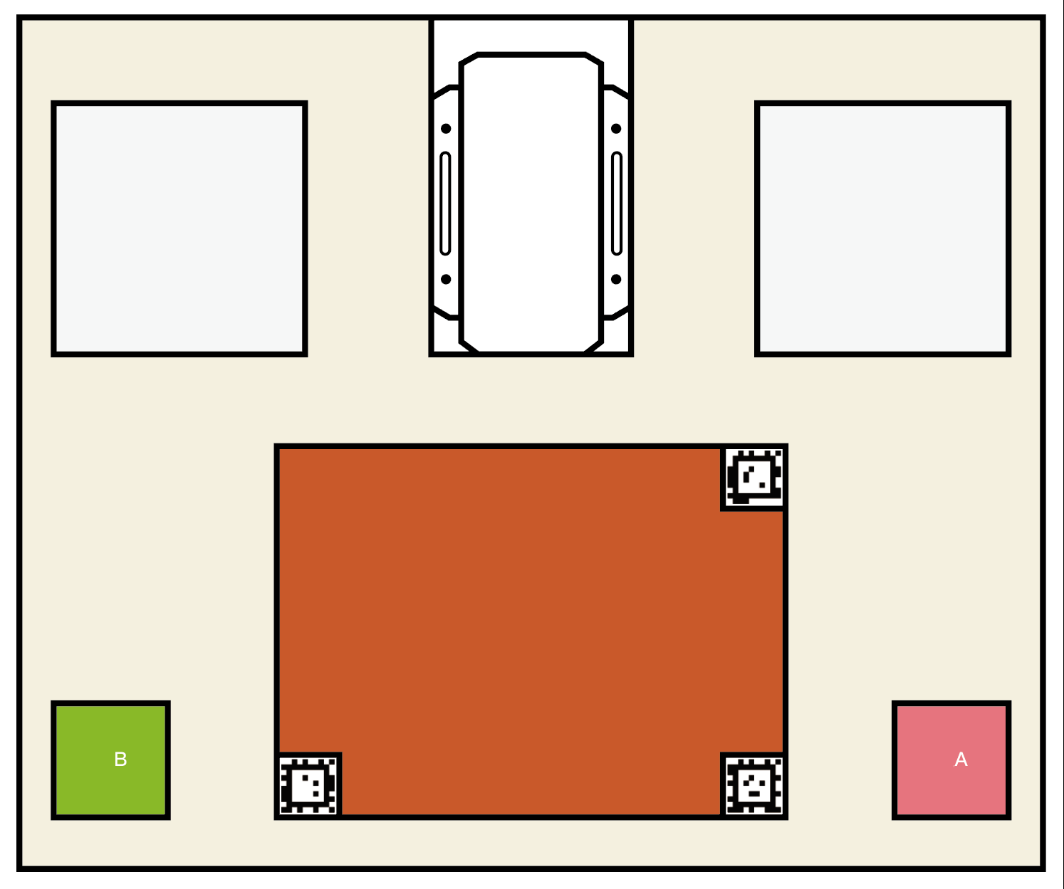
Please refer to the operating instructions in the code for specific operation.

## 5. Sample experiments

# 5.1 Swap Blocks

This demo shows how to exchange the blocks in Area A and Area B on a map. The identification area will be used as a transit area on the way.

The specific process is as follows:



* Images above are just examples
* Place two blocks in Area A and Area B respectively.
* Press the Menu Button to enter the ***Menu Mode*** and select ***“RunDemo-> SwitchAB”***.
* The robotic arm automatically starts to exchange the blocks, then returns to the "sleep" position when finished. Press the Menu Button during the process, the program will be terminated, and the robotic arm will be stop at the current position.

# 5.2 Action Repetition

Action reproduction function allows user recording and reproducing the process, time up to 80 seconds. The specific process is as follows:

* Press the Menu Button to enter the ***Menu Mode*** and select ***“RunDemo-> ActionREC”***.
* Select “***Record”*** to start recording, and the user manually operate the robotic arm. Notice! Once the recording starts, the robotic arm will enter the ***Release State***, Please hold it manually before recording to avoid accidents.
* Select “***Record***” again to stop recording and the robotic arm will be fixed at the current position.
* Select ***“Play”*** to start repeat the action. The robot arm moves to the starting point of the action and then starts to reproduce the action, after the reproduction is completed the robot arm will be fixed at the last position

# 5.3 Vision capture (Visual Perception Suite is required to purchase)

# 5.3.1 Installation

* Please install the Dependent Library first.

Copy the path ***“~/sagittarius\_ws/src/sagittarius\_arm\_ros/3rd\_app/apriltag.zip”*** to another location, such as “***~/Downloads”***, and decompress the ***“apriltag.zip”*** to get the folder ***“apriltag”*** .

# cd ~/Downloads/apriltag

# cmake .

# sudo make install

* Delete or rename a file to make the package “***apriltag\_ros”*** a compilation option. The path of the file as follows: ***~/sagittarius\_ws/src/sagittarius\_arm\_ros/3rd\_app/apriltag\_ ros/CATKIN\_IGNORE.***

Execute the “***catkin make”*** to compile.

* Connect the Intel D435 camera to the computer, then calibrate the camera and adjust the value of the filter before vision capture.

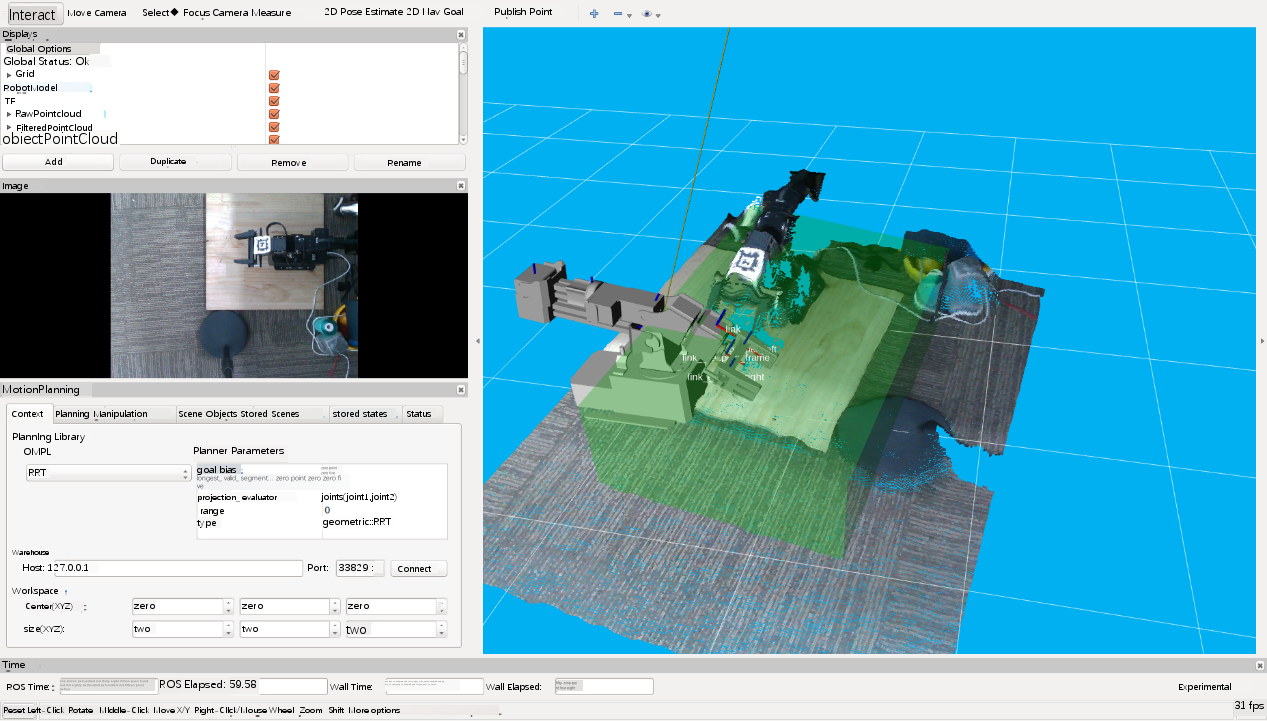
# 5.3.2 Calibration

* Open the terminal:

# cd ~/sagittarius\_ws

# source devel/setup.bash

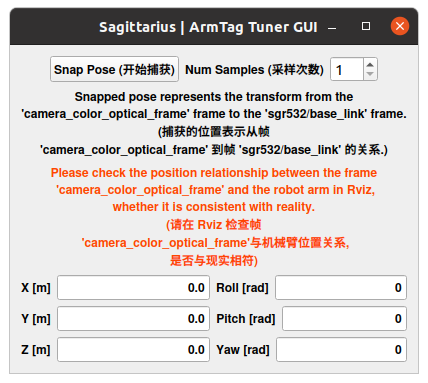
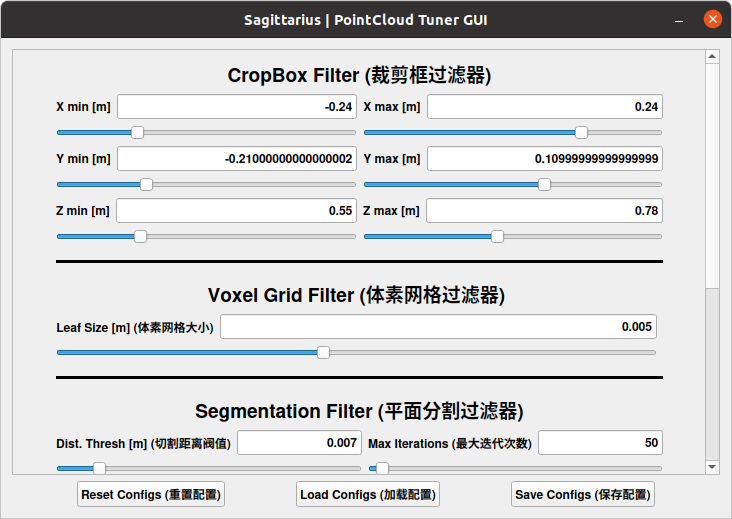
# roslaunch sagittarius\_perception calibrate.launch



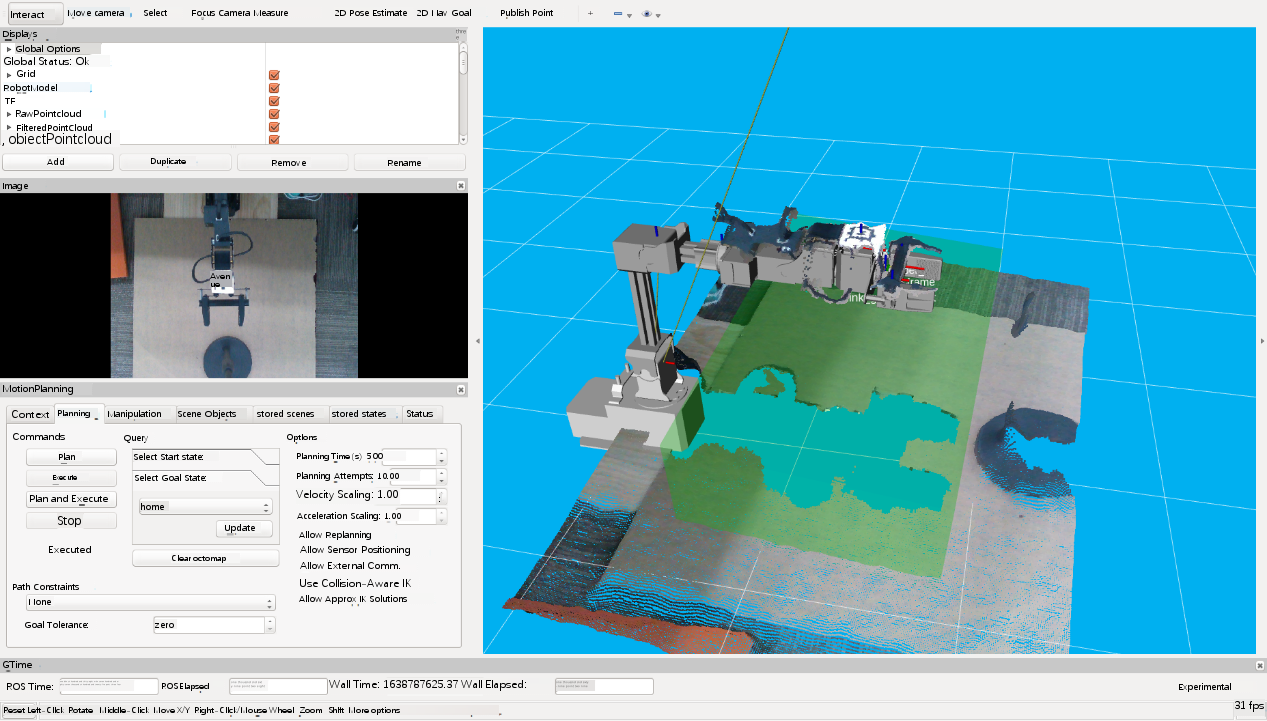
* Control the robotic arm move to the "home" position. Stick the Apriltag to the center of the surface of No.7 servo, and place the camera above the robotic arm to keep Apriltag in camera range. The specific effect is shown in the following figure:



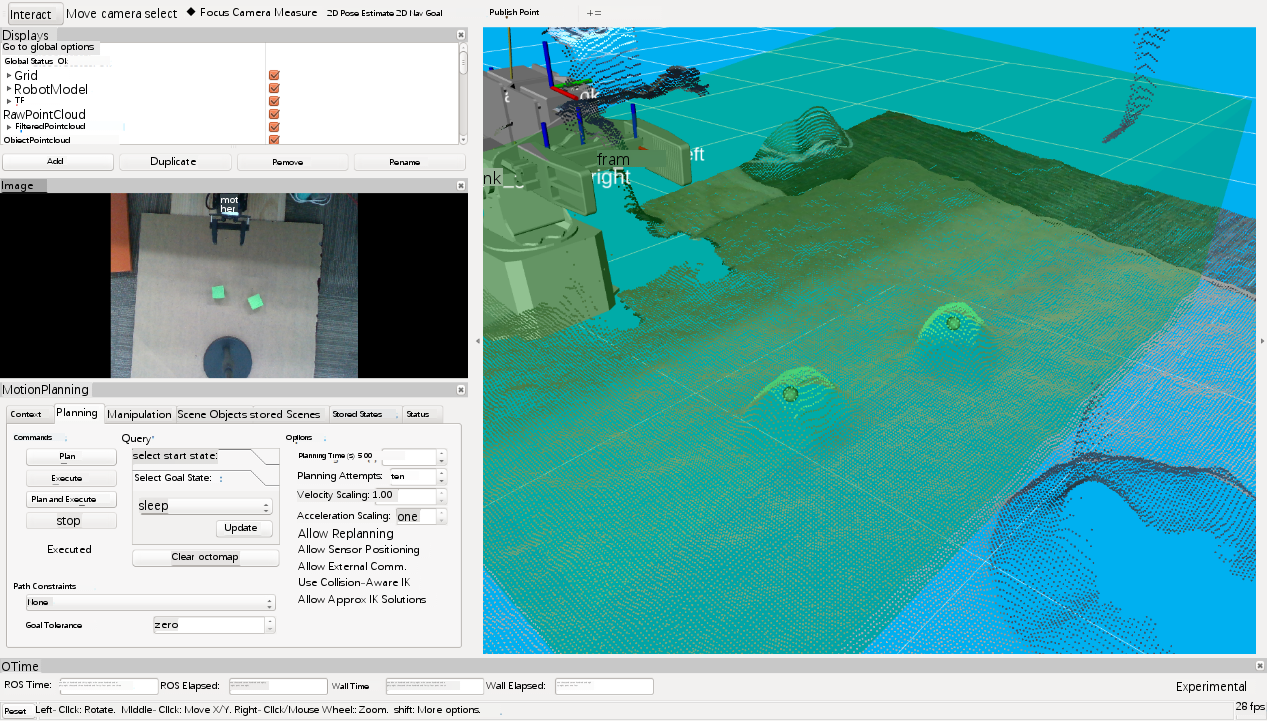
* The program ***RViz*** will start automatically. At the same time, the calibration program ***“ArmTag Tuner Gui”*** and the filter program ***“PointCloud Tuner GUI”*** , as figures below, will start together.



* Click ***"Start capture"*** inthe program ***“ArmTag Tuner GUI”***, and the program will automatically calibrate and save the calibration data itself. If succeed, you will see the location of Apriltag in RViz is consistent with that in the real world. The effects are as follows:



* Control the robotic arm to the "Sleep" position. Adjust the filtering conditions with the program “PointCloud Tuner GUI”. The green area in RViz represents the filtering area, in which the filter will identify all the data that meet the requirements. The effect is as follows:



* After the adjustment completed, you need to save the parameters of the filter manually. The save path is ***config/filter\_parms. yaml,*** system will use the filter by default.
* Close the program when the calibration is completed.

**5.3.3 Grab**

* Open the terminal

# roslaunch sagittarius\_perception do.launch

* Control the robotic arm to move to the "sleep" position and open a new terminal

# source devel/setup.bash

# rosrun sagittarius\_perception pick\_place\_sa.py

* The robotic arm starts to grasp, and ***RViz*** displays the grasping process in real time. When the grasping is completed, the robotic arm returns to the "sleep" position.