Robotic Arm Instructions

Here I am presenting a Robotic Arm project using Raspberry Pi 3, Python Code, 3D printed parts, 6 Hobby Servo motors, 5volt 2.5 Amp power supply and an Android App. specifically created to control the arm using 'MIT App Inventor' which is a great cloud based application to create Android apps for free. Project file for the app is also attached so that you can tailor the app to suit your needs and or learn to create new Android apps. Highlight of this project is Zero Jitter Servos, all servos are under perfect control.

This Robotic Arm has 5 DOF (degrees of freedom) controlled by 5 servo motors plus a gripper powered by another servo motor. All parts including the Raspberry pi can be powered with a single 5 volt DC 2.5 amp power supply and does not need any additional boards (Arduino or so). If you have no access to a 3D printer, don't worry, you may purchase a ready-made robotic arm online for a reasonable price (see link below). The Arm runs on code written in Python3 language. Just follow the steps described below and you would be having your very own Robotic Arm at your service I no time!

Downloads:

- Robotic Arm STL Files

  Robotic Arm 3d Model Solidworks files are available here (Courtesy Grabcad.com): https://grabcad.com/library/robotic-arm-215

- Android App -.apk fie
- Android App Project file - .aia file

The files below need to be copied to your RPI /home/pi/robot as described in Step-
Robotic Arm Python Code –robarm.py

- servo01Spos.txt
- servo02Spos.txt
- servo03Spos.txt
- servo04Spos.txt
- servo05Spos.txt
- servo06Spos.txt

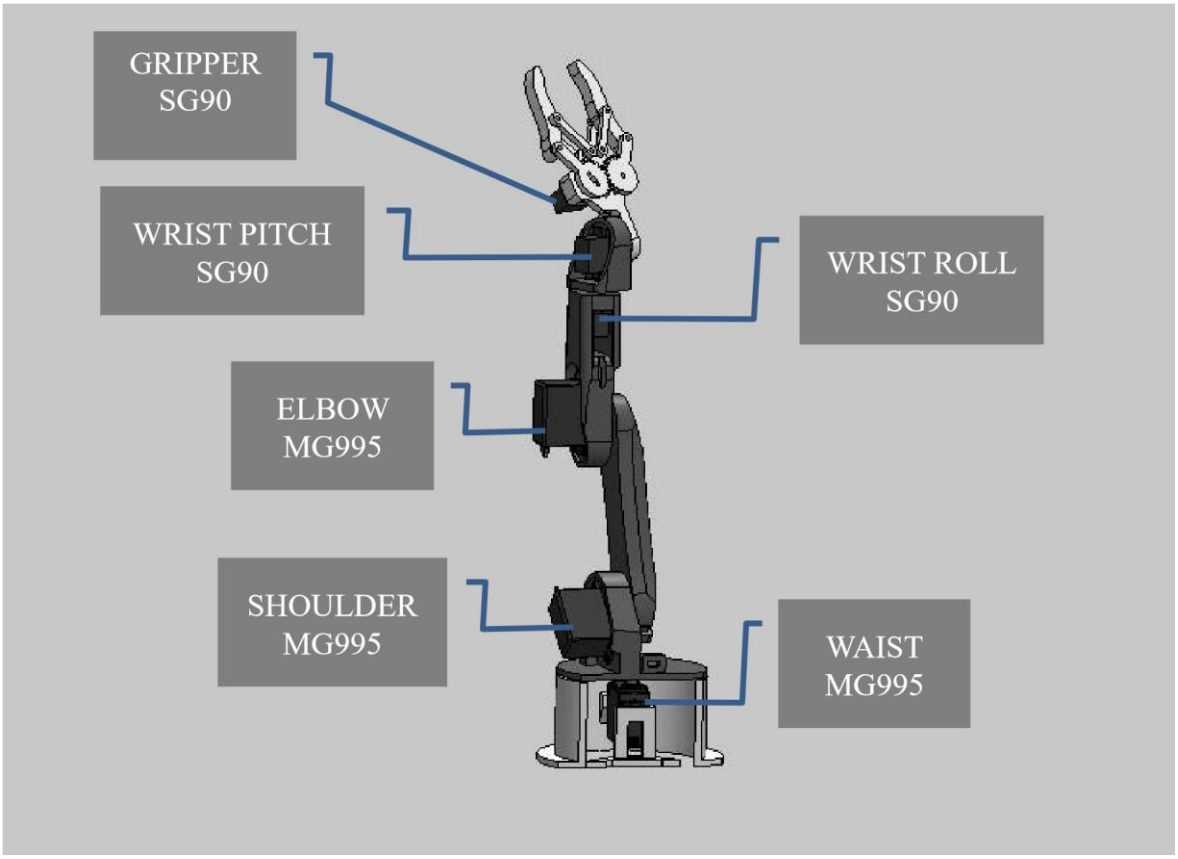
Step: 1 : Bill of Materials

- 3D printed Parts of the Robotic Arm or Ready-made Arm.
- MG 996 Servo Motor – 3 Nos.
- SG 90 Servo Motor – 3 Nos.
- Raspberry Pi3 Board – I Nos.
- 5 Volt 2.5 Amp Power supply – I Nos.
- Bread Board (optional)
- Screws to mount the Arm on base board, screws to fix the servo motors in place.
- Jumper wires.

Step 2 - Printing out 3D parts and Assembling

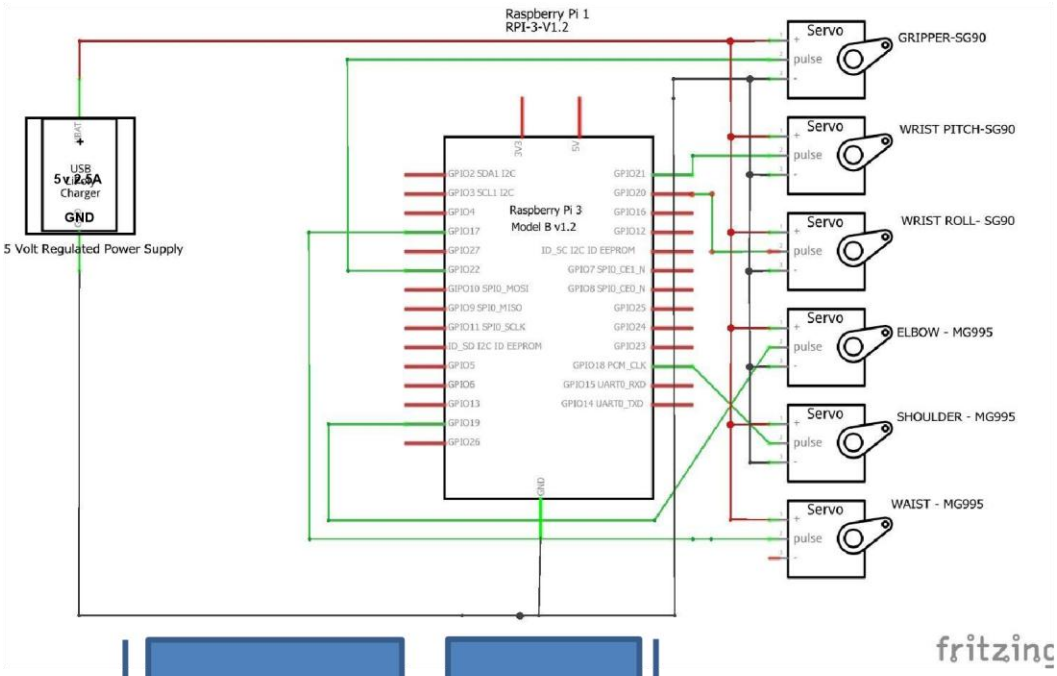
Download the stl files from the download section above and print them out and assemble.

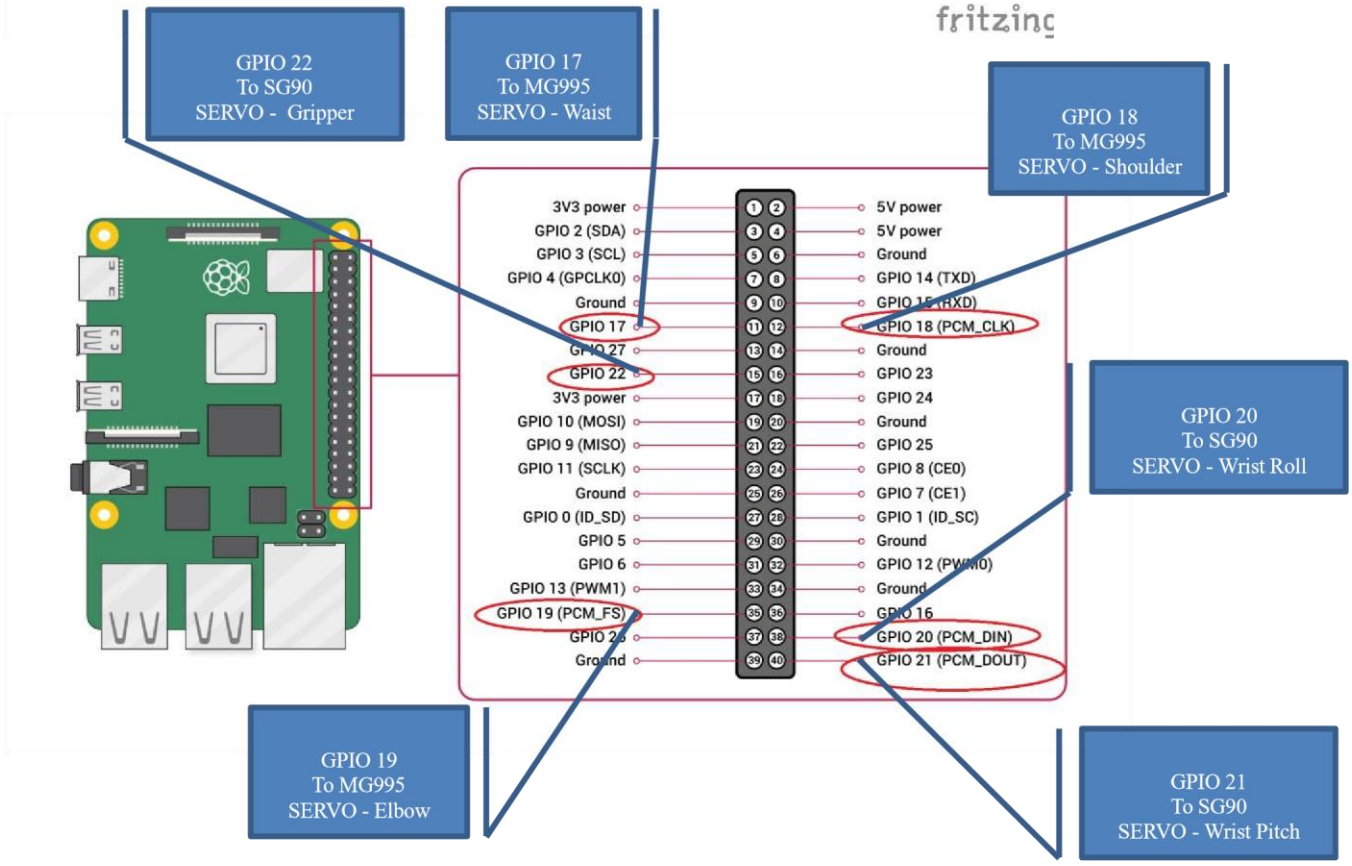
- Please use 20% or higher in-fill, no support structures needed.



Step3: Wiring Diagram

Please connect the Servos, RPI and power supply as shown below:





Step – 4 Installing OS in Raspberry Pi3

Next step is to set our raspberry pi3 to run the Python code. First of all, you need to install the Raspberry Pi OS (Raspbian). If your RPI (raspberry pi) is already loaded with the OS, you can move on to the next step.

Download the OS from https://www.raspberrypi.org/downloads/ and copy it to a SD card and power up your RPI with the SD card inserted. Follow the instructions on screen and complete the installation set up.

Open a terminal window and enter the following commands:

```
sudo apt-get update sudo apt-get
upgrade
```

Step-5: Uploading the Python code:

Make a new directory in your RPI"s home folder called „"robot". (You may use any name, but then just ensure that the same name will be used in the subsequent steps).To create a new directory / folder called „robot", enter the below code in a terminal window and press enter.

```
sudo mkdir robot
```

Download robarm.py, servo01Spos.txt, servo02Spos.txt, servo03Spos.txt, servo04Spos.txt, servo05Spos.txt, servo06Spos.txt and copy to the newly created directory – robot with the following commands.

```
cd Downloads  sudo cp robarm.py
servo*.txt /home/pi/robot
```

Step – 6. Establishing Bluetooth Connection with your Android device.

In order to connect your Android device and RPI via Bluetooth and enable serial communication, we need to set up Bluetooth Serial Port profile (SPP) on RPI. Please follow the steps below to establish SPP connection between RPI and Android phone, and then send and receive data using the serial terminal on both ends. Keep your Android device ready and follow the steps below.

First, we need to add a new service by creating a new rfcomm service. Enter the following command in RPI terminal:

```
sudo nano /etc/systemd/system/rfcomm.service
```

And then copy and paste the following inside

```
[Unit]
 Description=RFCOMM service
 After=bluetooth.service
 Requires=bluetooth.service

[Service]
 ExecStart=/usr/bin/rfcomm watch hci0

[Install]
 WantedBy=multi-user.target
```

Save by pressing crl+x, type „y" and enter. Now we need to enable the service with the following command:

```
sudo systemctl enable rfcomm
```

Reboot:

```
sudo reboot
```

Install SPP in Raspberry Pi:

Open Bluetooth service configuration file:

```
sudo nano/etc/systemd/system/dbus-org.bluez.service
```

Find the line that starts with "ExecStart" and add compatibility flag „-C" at the end of the line.

```
ExecStart=/usr/lib/bluetooth/bluetoothd -C
```

Add the line below immediately after the “ExecStart” line, then save (ctrl + x, type „y‟ and then „Enter‟) and close the file.

```
ExecStartPost=/usr/bin/sdptool add SP
```

Reload the configuration file:

```
sudo systemctl daemon-reload
```

Restart the service:

```
sudo systemctl restart bluetooth.service
```

Pairing: To establish a connection, Raspberry Pi and the phone need to be paired.

First Launch bluetoothctl:

```
bluetoothctl
```

```
discoverable on
```

On the Android phone, scan for Raspberry Pi and pair. Alternatively you can click on he Bluetooth icon on RPI monitor and use „Add device‟ as well. You should be able to see something similar to this:

```
[CHG] Device XX:XX:XX:XX:XX:XX Paired: yes
```

Press Ctrl+D to quit.

Download the „robotic arm controller app‟ and Install it in your Android Phone. Launch the app and press connect. If all goes well you should see the below in RPI terminal window.

```
$ sudo rfcomm watch hci0
Waiting for connection on channel 1
Connection from XX:XX:XX:XX:XX:XX to /dev/rfcomm0
Press CTRL-C for hangup
```

Enabling PIGPIO: We would be using „pigpio" module (library) to control the servos. So we need to enable the same in order to use it.

```
sudo systemctl enable pigpiod sudo systemctl
start pigpiod
```

We are now ready to test the robotic Arm. Ensure all connections are good and make sure that the robotic arm won"t slap you when it springs up to the initial position, keep a safe distance. Navigate to the python code file –home/pi/robot/robarm.py, open it using your favourite editor and run the code. Move the sliders in the App and enjoy.

Step7: Operation:

Open the Python code file (robarm.py), open the robotic arm app in your android phone, press connect (if the phone is already in connected state, refresh by disconnecting and connecting again), and move the sliders once connected and see the arm in action. Press SAVE and move the sliders to record the moves and then you can press the RUN button and see the robot running the moves over and over. The moves will be written to files in your RPI and will not be lost even if the power is switched off. You can PAUSE and resume the re-play with the RUN button. Press STOP button to end re-play and the robot will switch to manual mode again. If the RESET button is pressed, all saved positions will be deleted. Speed of the movements during re-play can be controlled with the „Speed" slider.

Please ensure that your Android device does not disconnect during the course of operation. Disruption will cause the program to stop and you will have to re-run the python code (robarm.py) or reboot the RPI.

If you want to run your RPI in headless mode (without monitor), an easy way is to use VNC (Virtual Network Computing), so that you can control the RPI from your smart phone or PC. Please follow this link. https://www.raspberrypi.org/documentation/remoteaccess/vnc/#:~:text=%20To%20turn%20this%20feature%20on%3A%20%201,connect.%0ANote%3Aexisting%20connections%20must%20be%20restarted%20in...%20More%20 **Step 8 – Final: Execute python code on Boot-up:**

If you want the python code to run on boot up (start-up) we need to add it to „crontab" and make it executable. Follow the steps below in order to achieve this:

Type in:

```
cd robot
```

then:

```
nano launcher.sh
```

this will launch the editor, type in this script:

```
#!/bin/sh
# launcher.sh
# navigate to home directory, then to „robot", then execute the python script, and back
home

cd / cd
home/pi/robot sudo
python3 robarm.py
```

cd / And then make the launcher script executable, which we do with this command:

```
chmod 755 launcher.sh
```

Now test it, by typing in:

```
sh launcher.sh
```

This should run your Python code.

We are using „crontab", which is a background (daemon) process that lets us execute scripts at start-up and at specific times. First we need to make a directory for the log file, which would be helpful in finding errors if the file fails to execute at start-up.

Navigate back to your home directory:

```
cd
```

Create a „logs" directory:

```
mkdir logs
```

Now we need to edit crontab, type in:

```
sudo crontab -e
```

This will bring up a crontab window, add the below line and save (ctrl+x, type „y", press enter):

```
@reboot sh /home/pi/robot/launcher.sh >/home/pi/logs/cronlog 2>&1
```

Reboot RPI,

```
sudo reboot
```

Wait for start-up and see if your script launches automatically (you cannot see the file opening, as it will be running in background). Connect your android device and the arm will respond to your commands from the App. If it doesn't work, check out the log file:

```
cd logs cat
cronlog
```

This will show you any errors that the system might have encountered.

Courtesy / Reference:

- Wikidpedia: https://en.wikipedia.org/wiki/List_of_Bluetooth_protocols?wprov=srpw1_0
- Bluetooth serial communication: https://pirobotblog.wordpress.com/2016/12/22/serialbluetooth-communication/
- MIT App Inventor: https://appinventor.mit.edu/
- Run python script at atart-up: https://www.instructables.com/id/Raspberry-PiLaunchPython-script-on-startup/
- Robotic Arm 3D design files: https://grabcad.com/library/robotic-arm-215
- Robotic Arm readymade: https://www.amazon.in/Robotic-Arm-Printed-DIYkit/dp/B07N41TSC9/ref=sr_1_16?dchild=1&keywords=robot+arm&qid=1600847358&sr= 8-16
- Raspberry pi Org: https://www.raspberrypi.org/