

# Pick and Place: Fully Autonomous Robotic Manipulator

By Tyler Boyer

## Description

Within this project, I plan to develop an Autonomous Robotic Manipulator (Robot Arm) with the ability to grab arbitrarily placed Lego minifigures within a set area and place them on the opposite side of the arm's base. This project will involve developing functionality for the various stages of the robotic manipulation process, including perception of the position of minifigures within the original task space, filtration of the resultant point cloud to obtain a specific minifigure position without adverse effects from potential outliers, trajectory planning to change the pose of the robot arm to that of the minifigure and subsequently to the opposite side of the robotic manipulator's base, and PID-based controls to ensure the robot is robust and can continue its task even with outside interference or slight errors in following its desired trajectory. The PID controls should also implement stiffness control to ensure safety around humans and fragile objects.

In the event the initial pick-and-place robot is successfully implemented, I may attempt to replicate the pick-and-place robot in an ESP-connected system of similar devices to construct a full assembly line which will be tasked with manufacturing Lego Star Wars Droid minifigures. This stage of the project is extra, however, and most efforts will be spent on developing an effective autonomous pick-and-place robotic manipulator.

## Resources

Thus far, I have found various resources with regards to constructing motor drivers, implementing PID controls, programming STMs, implementing pick-and-place algorithms, filtering point clouds, trajectory planning, and microcontroller-accessible SLAM algorithms.

<https://www.instructables.com/Build-Your-Own-Motor-Driver/>  
<https://acrome.net/post/pid-controller-design-for-stm32-microcontrollers>  
<https://smartsolutions4home.com/how-to-program-stm32/>  
<https://manipulation.csail.mit.edu/index.html>  
<https://www.roboticsbook.org/intro.html>  
<https://openslam-org.github.io/tinyslam.html>  
<https://www.youtube.com/watch?v=ygrsIqWOH3Y>  
<https://acrome.net/post/applications-that-you-can-do-with-brushed-dc-motors>  
<https://www.electronics-tutorials.ws/io/h-bridge-circuit.html>  
<https://bentlybearings.com/wp-content/uploads/2017/07/Understanding-and-Using-Dynamic-Stiffness.pdf>

## Design

Currently, my design involves using an STM32-F446RE as my main controller board that will run the algorithms for perception, filtration, planning, and controls. It will be connected to 6 motor drivers I make using H-bridge circuits, with each driver driving a dedicated motor. I will use 6 motors in this arm to achieve 6 Degrees of Freedom (DoF) allowing the robot to make complicated

movements to effectively grasp arbitrarily placed objects within its task space. The grasper itself will utilize two “fingers,” or grippers, to grab objects. It will also have an additional motor and driver allowing it to open and close, technically adding an additional DoF. I will use higher power, higher torque motors at the base of the arm, and gradually use lighter motors towards the end effector itself. There will likely be 3 motors placed at the base of the arm, 1 at a joint in the middle of the arm, 2 controlling the position of the gripper itself, and 1 final motor controlling the end effector, which will open and close around desired objects.

Buildwise, The frame of the arm will likely be made of either aluminum or wood sheets, meaning most frame parts will come from a laser cutter or a waterjet at the Invention Studio, and will likely be designed in Inventor or Fusion. For debugging purposes, I will also construct a manual control board before implementing autonomous functionality, and will likely incorporate potentiometer knobs that will control speed and direction of each individual motor. I may also connect this to a serial monitor on my computer for other debugging information that may be useful for creating the necessary algorithms.

With regards to sensors, I will be using encoders to track movement of robotic joints for the purpose of knowing current pose and ensuring pose transformations were done correctly. I may use an STM ToF sensor for perception, though I am still figuring out which one to use specifically. I may also want to use force sensors both for stiffness control and for ensuring the grasper doesn't squish objects it is grabbing.

## Expectations

At a minimum, I expect to have basic pick-and-place functionality implemented. This means being able to grasp and transfer minifigures placed in a known task space at a known, constant distance from each other to the other side of the manipulator's base. However, I do expect to have implemented some form of PID controls and decent stiffness control to ensure the arm is safe around humans if it were scaled up. Placing the minifigures at known positions is a minimum because it assumes either the perception algorithm is faulty, the filtration algorithm is faulty, there are issues with trajectory optimization, or a combination of 2 or 3 of these issues. This minimum viable product, however, should be fully autonomous and not rely on the debugging controller I will likely produce alongside the robot. These are the goals I have set for myself to at least get an A on the project.

## Mid-Project Review

By the mid-project review, I expect to have fully constructed the robot arm and have connected it to a manual debugging controller. This will ensure I have enough time to develop and debug the necessary algorithms to achieve autonomous pick-and-place functionality. I also expect to have implemented basic PID and Stiffness control at this point, as I can test these in conjunction with manual trajectory planning in what would be a Hybrid Control robotic system. While I don't expect the arm to be polished or perfectly functional in all desired DoF at this point, I do want to be able to perform at least basic pick-and-place operations manually at this point, ensuring I have enough time to polish the physical portion and implement the remaining autonomous portion.

## Parts

The microcontroller being used will likely be a STM32-F446RE, though I am also considering some more powerful STM32. I may potentially use an FPGA such as the Digilent Basys 3 Artix-7 board I have (granted this option is expensive), though if a cheaper FPGA is desired perhaps I may use a Tang Nano 20K instead. A complete list of parts I am likely using and others I am considering is provided below:

<https://www.amazon.com/STM32-Nucleo-Development-STM32F446RE-NUCLEO-F446RE/dp/B01I8XLEM8>

<https://www.digikey.com/en/products/detail/dfrobot/SEN0297/10136551>

<https://www.digikey.com/en/products/detail/adafruit-industries-llc/4489/11594498>

<https://www.adafruit.com/product/3190>

<https://www.digikey.com/en/products/detail/pololu/2990/10450471>

<https://www.robotshop.com/products/e-s-motor-dc-brushless-worm-gear-motor-2430bl-robots-24v-90rpm>

<https://www.digikey.com/en/products/detail/dfrobot/FIT0441/6588579>

<https://www.robotshop.com/products/e-s-motor-high-torque-dc-worm-gear-motor-w-encoder-6v-23rpm>

<https://www.st.com/en/imaging-and-photonics-solutions/time-of-flight-sensors.html>

Might consider the following:

[https://estore.st.com/en/nucleo-n657x0-q-cpn.html?icmp=tt42036\\_gl lnkon\\_dec2024](https://estore.st.com/en/nucleo-n657x0-q-cpn.html?icmp=tt42036_gl lnkon_dec2024)

<https://wiki.sipeed.com/hardware/en/tang/tang-nano-20k/nano-20k.html>

<https://www.amazon.com/Digilent-Basys-Artix-7-Trainer-Board/dp/B00NUE1WOG>

<https://www.digikey.com/en/products/detail/dfrobot/SEN0245/8827827>

Something similar to what I want to make:

[https://www.reddit.com/r/robotics/comments/1kvug8o/my\\_diy\\_robotic\\_arm\\_with\\_object\\_detection/](https://www.reddit.com/r/robotics/comments/1kvug8o/my_diy_robotic_arm_with_object_detection/)

My idea doesn't necessary involve sensing the color of the object like this project likely does, though it could be something to fall back on if my current idea gives me too much trouble.