BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2023
http://excel.fit.vutbr.cz

Robotic arm with RC components

Petr Bobcik

Abstract

The goal of this work is focused on creating own robotic arm with sensors and camera attached to it, so the robotic arm will be able to detect the surroundings. The robotic arm is well known and used in many proffesions. The problem is that the common robotics arm (not industrial robotic arms) are quite dump. They do not use any sensors for detecting collision, sensors for detecting the gripper picked up any item or camera to detect the surroundings. I decided to add camera to the gripper of robotic arm to detect obstacles. The user can use the robotic arm GUI, that I made, to control robotic arm motor by motor separately or use camera vision to detect obstacles according its color or AI item recognition. For that I use the openCV python library. For stalled detection I decided to use accelerometers and encoders the encoders is not suitable due to it is time consuming and it caused many problems. The reason is that every few ticks the previous angle must be compared with new value and if the value was not read continuously, it wrongly detected the stall detection. In case of accelerometer only what is needed is to read the value and check if it is greater then trashold. During testing proces I realized that this stall detection is not so precise. Better solution will be check the stall detection on HW level, which means on stepper motor driver side. The resulting robotic arm is good for begginers and for advanced robotic arm users too. The begginer can use it for testing and discovering the robotic arm solution and advanced user can use basic stall detection and implemented camera as well.

*xbobci02@vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

The reason why I decided to build such a robotic arm was just a curiosity if I can make my own robotic arm that will be clever in some way. The reason why I was curious was that solutions I found were in some way "dump" (with a few exceptions), with no sensors, camera, etc. So the robotic arm was just a simple device that did not care if the result of desired operation was successful. I wanted to make something more clever and my own.

2. Contribution

The problems I tried to solve were to add "eyes" and "senses" to the robotic arm. These sensors should be used for position detection of each joint, detecting objects around the robotic arm or detecting if the gripper grabbed any object. Another problem to solve is the default position of the robotic arm. The robot should automatically set itself to the default position instead of manual setup by user.

3. Existing solutions

The existing solutions that have close connection to my robotic arms are for example, Nyrui One [1], AR4 [2] or BCN3D MOVEO [3]. All of them inspired me in some way (each robotic arm inspired my in different way) and I tried to pull advetages of all these robotic arms together.

For example the AR4 robotic arm is very precise and with high repeatability (0.2mm). Another adventages are for example the attachable camera or stall detection in case of Nyrio One. Another adventage is that both Nyrio robotic arms and BCN3D MOVEO are 3D printed, which means that when any component brokes, it can be easily can be reproduced in home condition (by using 3D printers for example). The AR4 robotic arm is metalic, so it is not too easy to reprodue it at home. The BCN3D MOVEO does not offer necessary informations, but its adventage is that many users use it, so it was improved many times.

4. Results

I reused existing robotic arm called MOVEO from BCN3D, which is commonly used by many users. Original BCN3D MOVEO documentation does not offer to much information. I decided to use it, due to it was tested meny times and several problems were fixed. The original BCN3D MOVEO robotic arm was designed for 3D printer, so I decided to print it as well from PLA material.

I designed own electronic and printed circuit board, such as **main control board** and **sensor board**. These boards are visualize in the middle of the baner. The **main control board** connects and centralize all sensors and periperal to one place. The sensor board is the board for each stepper motor that contains accelerometer and encoder. The accelerometer is used for stall detection and the encoders for position capture when all steppers are turned off. Due to this feature the robotic arm can be set to some position, capture that position and then the robot can reach tem automatically.

The graphical user interface, which is visualize on the bottom two figures of the banner, allows two controlling modes. The first is to control one stepper motor by one or all motors together via inverse kinematics (for the inverse kinematics i used ikpy python library) together with camera view. The camera offers to detect moving objects, objects by color or by item recognition with AI. I am planning to add the capturing robotic arm position for further rerun, but this is still in progress.

One of the goals I mentioned in **Problem definition** chapter was gripper logic. The gripper should know if it is holding object or not. For that I am using the current sensor, that measures the current through servo motor. When the gripper is closing and the current is higher then specified threshold, the gripper stops closing, because the object was grabbed.

5. Conclusion

The goal of this work was to create own robotic arm with these features: detection of stepper motor position, detecting grabbed object in gripper, detecting of the stepper motor stall and simple graphical user interface with camera. The goals were almost fulfilled. For example the graphical user interface and camera are well made, but the connecting with inverse kinematics is in progress. The problems I found is the long wires for I2C communication, which leads to timeouts or lost data.

Another problematic part is stall detection. The problem is that it is not too precise as I wanted to. Sometimes it detects stall even if the stall did not happen. Previously I wanted to implement it with encoders, but it was time consuming operations so it was not problematic to implement. Another part that I am planning to add is capturing robotic arm positions for further replication, where the encoders will be used.

As the future improvements the stall detection can be improved by choosing sophisticated stepper motor drivers that check the stall by measuring the current when coils are disconnected.

References

[1] Niryo One [online]. Wambrechies, France: Niryo [cit. 15. January 2022]. Available at: https://niryo.com/fr/product/niryo-one/.

[2] AR4 Robot Manual [offline]. Annin Robotics, 2022 [cit. 17. January 2022]. Available at: https://drive.google.com/file/d/1T7tWd5-ZMWTbNEdSN-Tvi7uAK0rFrwcS/view.

[3] BCN3D MOVEO: A fully Open Source 3D printed robot arm [online]. Barcelona, Spain: BCN3D, july 2016 [cit. 17. January 2022]. Available at: https://www.bcn3d.com/bcn3d-moveo-the-future-of-learning-robotic-arm/.