Wearable Robotic Third Arm 2021 Fall Semester Report

Designing an Online Controller for the Wearable Robotic Third Arm for Self-Handovers and Obstacle Avoidance

Raphael, A, F, Fortuna

Cornell University, raf269@cornell.edu

The Wearable Robotic Third Arm previously used an offline controller to go to a predetermined location and deliver an object to the other hand of the human. This can be challenging for the human wearing the arm as it requires more user input and the user must remember where the arm will move to, especially while engaged in another task. This report details the development of an online controller to adapt the robot’s movement based on the target hand motion to deliver the grasped object. This includes avoiding collisions with the human body and other obstacles as well as controlling a underactuated 5-degree-of-freedom robotic arm.

CCS CONCEPTS • Insert your first CCS term here • Insert your second CCS term here • Insert your third CCS term here

**Additional Keywords and Phrases:** Handovers,Wearable Robotics, Motion Planning

ACM Reference Format:

First Author’s Name, Initials, and Last Name, Second Author’s Name, Initials, and Last Name, and Third Author’s Name, Initials, and Last Name. 2018. The Title of the Paper: ACM Conference Proceedings Manuscript Submission Template: This is the subtitle of the paper, this document both explains and embodies the submission format for authors using Word. In Woodstock ’18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY. ACM, New York, NY, USA, 10 pages. NOTE: This block will be automatically generated when manuscripts are processed after acceptance.

1. Introduction

Wearable robotic arms are robots that are attached to a person that can add assistive benefits. For example, providing support when assembling cars from uncomfortable angles, holding a drink while the person is working on something, or opening a door when a person’s hands are full. This report outlines the development of an online controller for a wearable robotic arm, the Third Arm, for use in handing something to oneself, self-handovers. The controller consists of a servo controller to move the arm, sensor integration with ROS and the OpiTrack Motion Capture systems for object locations, a motion planning system to control the Third Arm, and an obstacle avoidance system.

With this system, tasks like cooking can be supported with the Third Arm as one hand can be stirring a pot while another is given an ingredient out of reach normally as in *Figure 1*.

Diagram

Description automatically generated

Figure : User wearing the Third Arm while cooking and using it to pass them eggs.

1. Literature Review

**\_\_\_\_\_** [3] discussed the different types of wearable robotic arms and missing from it were online controllers for obstacle avoidance and handovers. Vatsal [1] discussed the inverse kinematics solution for the Third Arm in his technical report and implemented it in MATLAB. Chen and Song [2] found that they were able to use a potential field controller with their robotic arm to avoid obstacles.

1. Development

The Third Arm was originally developed using ROS Indigo on Ubuntu 14 using the Dynamixel Controllers ROS package. It had no online controller, integrated inverse kinematics controller, or sensor integration barring voice recognition for the experiments run by Vatsal.

Throughout the course of this project, the ROS and Ubuntu systems were upgraded, an updated control system was developed, OptiTrack sensing was added, an online controller was created, and obstacle avoidance systems began development as detailed below.

* 1. Upgrading

We began by updating Ubuntu 14 to Ubuntu 20, ROS Indigo to ROS Noetic, and Python 2.7 to Python 3.1 As the Dynamixel Controllers ROS package was not available for ROS Noetic, Dynamixel Workbench was tested for use but found to lack the velocity control and abstraction needed for the motion planner. PyPot, an open source Dynamixel motor controller, was used instead with great success and came with both register level and higher-level position and velocity control for the servos.

* 1. Control System Development

We began tests at the register level to control the Third Arm with PyPot. Once we found the arm could be controlled, we moved on to the higher-level positional and velocity control and validated the arm also worked with these commands through a series of demos and tests. We then created a servo control class for positional and velocity control and added methods to initialize each controller in the motion planner as well as integration for the PyPot robot configuration. This controller was then tested and validated by running the controller on the Third Arm’s servos.

**PICTURE**

* 1. Sensor Integration

OptiTrack integration was added to allow the Third Arm motion planner to know where the Third Arm base and gripper were, as well as the target position for the gripper to go to. We found that OptiTrack had compatibility problems with ROS Noetic, so we ported the Third Arm code over to ROS Melodic, where it worked well and returned the quaternion and position vector for each object.

**PICTURE**

* 1. Motion Planning Pipeline

The motion planner allows for the Third Arm to have a feedback loop for control and use its sensor integration code to determine where, and how, to move next. We first created a liaison file for all ROS communication that processed the quaternion and positional vectors into transformation matrices to be used by the inverse kinematics solver in the motion planner. This file would also control when the motion planner was called and feed it the required data to actuate the arm.

The motion planner was created next. It instantiated controllers for each Third Arm servo and the PyPot robot configuration. The inverse kinematics solver was ported over to Python and used in the motion planner with the transformation data sent from the ROS liaison file. Its outputs were processed by PIDs we created for each Third Arm servo. With these components, the arm was able to track and move to a target location.

**PICTURE**

* 1. Obstacle Avoidance Introduction

To avoid hitting obstacles on the way to the target, obstacle avoidance was added in the form of a potential field. The distance from each obstacle to the arm is calculated using the forward kinematics from the gripper and arm base positions to create a reference of the arm. The shortest path is then found from the obstacle to the arm as a vector and compared with the joint actuation directions to apply the appropriate adjustments in the correct directions to avoid hitting the obstacle. These adjustments would then go through PIDs before being combined with commands to move to the target.

Conclusions and future work

This upcoming semester, we plan to conduct further tests to finish the obstacle avoidance system, use the servo LEDs to communicate when the arm is out of reach of the target hand, integrate voice control to tell the arm when to perform a handover, and run an experiment where participants will be

Acknowledgements

We would like to thank Alap Kshirsagar for his collaboration and help during this project and research supervisor Dr. Guy Hoffman for his project direction and time management advice.

References

1: <https://arxiv.org/pdf/2011.07286.pdf>

2: <https://ieeexplore.ieee.org/abstract/document/8460185/media#media>

**How to write robotics papers**

Writing tips

What you did, what you found, what you did with it - make sure have sections that answer these questions

Have a good paper 2 weeks before to have a great paper to turn in

Don’t apologize

Point of paper first, not at end of paper

Don’t have large drawn out introductions

Newspaper style: map important thing then expands

Figure text needs to be as big as the text in the paper

Don’t talk about limitations?

Don’t Philosophy outside paper

Have properly formatted citations

Don’t trust google scholars

Don’t leave lines with one word

Don’t have references with questions marks

Don’t work until the deadline, be done before, won’t trust what your results are

Break up sentences

Replace this, that, those, these with what they are

1. Be clear, not fancy: Use everyday language. Be specific, not abstract.  Offer easily imaginable examples. Be sure your words make pictures in people’s heads. Be sure the pictures are the ones you intend.

2. Use most of your column for evidence: Tell stories, give statistics, show the impact of the problem or the solution on the real world. People can form their own conclusions if you give them the evidence. Don’t take much space for grand, abstract conclusions; let the reader form the conclusions.