NEW YORK UNIVERSITY

Mobile Mixed-Reality Interaction Using Computer Vision for Robot Control

By: **Hassam Khan Wazir**

N14684314

Submitted in partial fulfilment of the degree of

Master of Science (MS) in Mechatronics and Robotics

New York University

Mechanical and Aerospace Engineering

December 2017

MEGY 9966

Abstract

The aim of this project was to control a swarm of ground robots that do not have onboard sensors and allow them to navigate the environment only using the camera of a mobile device. To accomplish this task, problems related to object detection and tracking, localization and communication needed to be addressed. The object tracking was achieved using several tracking algorithms in OpenCV, and their performance was compared to find a suitable candidate. The localization problem involved finding the location and orientation of the mobile device in space. A Google Tango supported device was used that utilizes point cloud data to calculate its position and orientation accurately. The communication between the robot and the mobile device was established using Bluetooth which allowed the system to send location data to the robot as well as issue various tasks to the robot swarm. As of writing this report, the robot control and object detection is yet to be completed.

Table of Contents

Abstract i
Introduction 1
  Aims and objectives 1
Methodology 2
Hardware 4
  The Robot 4
  Mobile Device 6
Software 7
  Utilising Unity 7
  OpenCV 8
  Boosting tracker 9
  Multiple Instance Learning Tracker 9
  Kernalized Correlation Filter 10
  Track Learn Detect Tracker 10
  Median Flow Tracker 10
Conclusion and Future Work 11
References 12

List of Figures

Figure 1. Implementation of feature tracker that utilizes Optical Flow. 2

Figure 2. Diagram showing the project idea. 3

Figure 3. The differential drive robot. 4

Figure 4. Adafruit 16-channel Servo Controller. 5

Figure 5. The Parallax standard servo. 5

Figure 6. The Lenovo Phab 2 Pro. 5

Figure 7. Different methods of Object tracking tested. 9

Introduction

Although considerable advancements have been made when it comes to solving the estimation and control challenges in multi-agent systems, current solutions rely on sophisticated vision [1], [2] [3] and motion capture systems facilitated by powerful computing hardware and fiducial markers. This approach is useful for tracking and controlling the robots with high precision, however the cost and complexity of the setup makes it unsuitable for deployment in the field.

In order to solve this issue, a different approach leveraging the onboard computing, vision and inertial sensing capabilities of mobile devices has been used to create mobile mixed-reality interfaces that interact with robotic manipulators and multi-agent systems. Although this approach is portable as well as scalable, the use of fiducial markers [3], [4] to track the robots and the environment makes it unsuitable for use in completely uncontrolled environments.

This project aims to address this problem by proposing a mobile mixed-reality interface that will use computer vision techniques and the Inertial Measurement Unit (IMU) data from the mobile device to track the robots and assign tasks to them.

Aims and objectives

The project focuses on creating a system that can be used to control ground robots and allocate different tasks to them. The idea is to create a solution that uses only the vision from a mobile device to control multiple robots, that themselves cannot sense the environment. This idea, along with a user-friendly interface, can provide the user with the tools to allocate complex tasks to the robots with ease. Upon completion of this project, the aim is to successfully demonstrate multiple robots picking up objects and relocating them to the goal location. Furthermore, the robots should be able to do so in a variety of environments and the system should be able to track them accurately.

Methodology

There are three parts to this project. The first part consists of finding a way to track an object in real-time using the video feed from the camera of a mobile device, and create augmented reality markers on the robot for user interaction. The second part is related to localizing the mobile device in 3D space so that the device can know its position relative to the objects in the environment, and therefore, can determine the location of the robot relative from itself. Thirdly, there needed to be a way to send information to the robots to help them navigate the environment.

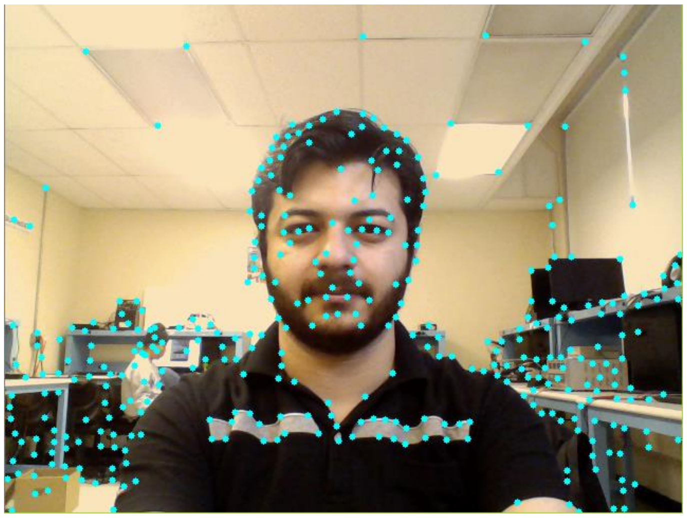

Figure 1. Implementation of feature tracker that utilizes Optical Flow.

A systematic approach was adopted to tackle all three problems, starting with the object tracking. To solve this problem, a computer vision technique called optical flow [5] was used which is later going to be supplemented with object recognition. Optical flow keeps track of the object as it moves through the scene by computing the velocities of individual pixels as they translate between two points in a sequence of ordered images. This, alongside an object recognition algorithm will allow the mobile device to track the location of the robots as well as issue commands to them to move to a particular location. As of writing this report, object tracking has been implemented, however the object tracking part is yet to be completed. A more detailed discussion related to object tracking will be done in the following chapter. Implementation of Optical Flow in Open CV is shown in **Error! Reference source not found.**.

The second problem pertaining to localization was addressed by using the Google Tango API and a supported device. Google Tango devices can project infrared light into the environment and generate a point cloud using the data collected by the depth sensor. This allows for an accurate 3D virtual representation of the environment and any translation or rotation performed by the device can be accurately captured and recorded. This gives users the freedom to traverse a large environment and gives them the ability to interact with the environment virtually. In this project, the user can drop virtual markers on the ground and keep track of their location up to a high degree of accuracy. These markers will be used as waypoints for the robot to navigate to a certain location as well as pick up objects and relocate them.

The third problem of establishing communication between the robots and the Android application was solved by creating a Bluetooth serial communication between the robot and the mobile device. Bluetooth was chosen over Wi-Fi due to high connection reliability, low power consumption, and reduced complexity when it comes to setting up the system. A more suitable solution might be used once the system complexity increases, but for the current stage of the project, the Bluetooth serves as a reliable mode of communication.

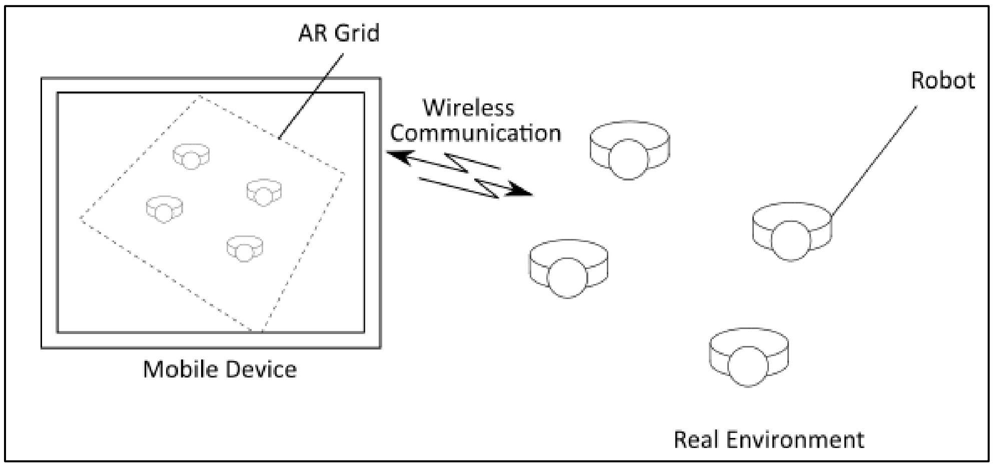

Figure 2. Diagram showing the project idea.

Hardware

The hardware used in this project consists of several ground robots and a mobile device. To start with, a single robot is used as a prototype to test the system. Once data is gathered that shows the system to work smoothly, more robots will be added to the system. The details of the hardware components are as follows.

The Robot

The hardware of the robot was kept simple on purpose since the aim of the project is to control ground robots and make them sense the environment by only using the vision from the mobile device. Moreover, it would prove an important point that robots can be given a certain level of autonomy with a high degree of accuracy without the use of sophisticated onboard sensors.

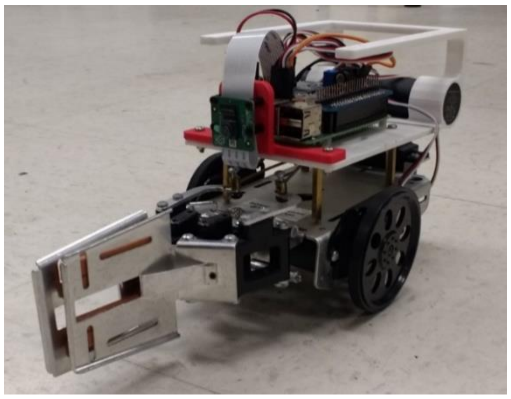

Figure 3. The differential drive robot.

For the project, a differential drive robot was used, as shown in Figure 3. It was primarily controlled using a Raspberry Pi 3 with a 16-channel servo controller manufactured by Adafruit industries. For the actuators, two continuous Parallax standard servos were used on the wheels and one Parallax standard servo was used for the gripper. The body of the robot has been 3D printed for the most part, and a castor wheel was added to the robot base to add stability.

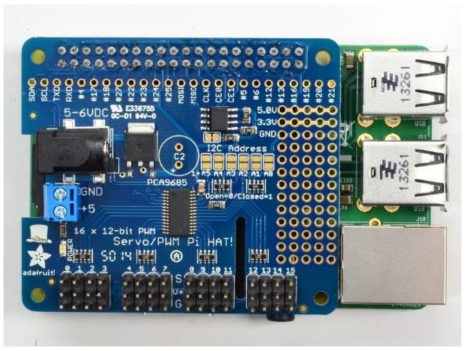

Figure 4. Adafruit 16-channel Servo Controller.

Figure 5. The Parallax standard servo.

The raspberry pi has inbuilt Wi-Fi and Bluetooth capabilities which makes it ideal for wireless communication. The onboard SD card makes it easy to store large amounts of data on the robot, and this ability can be further explored in the future to add extra functionality to the robots.

Figure 6. The Lenovo Phab 2 Pro.

Mobile Device

There were many hardware options available to test the software part of the project. The Lenovo Phab 2 Pro, as can be seen in Figure 6, was chosen for this project because it officially supports Google Tango and has an onboard depth sensor. Apart from the Wi-Fi and Bluetooth capabilities of the device, the large screen size also helps in visualizing the virtual markers and made interacting with them a lot easier.

Software

The mixed-reality mobile interface was developed as a mobile device software application on Android operating system because the Lenovo Phab 2 supports it, which also makes the device ideal for rapid prototyping on any computer capable of running windows or OSX. The application takes IMU data and the video feed from the camera and process the point cloud generated by the device in response to the data collected by the depth sensor, all of which is done in real-time.

The system was developed based on the following assumptions:

- The vertical (z) axis of the robot will always be parallel and opposite to the gravitational force vector.
- The ground plane is completely flat and orthogonal to the gravitational force vector.
- The mobile device will not undergo any sudden or very fast translational and rotational movements when tracking the robots.

Once the system is robust enough to work reliably based on the assumptions stated above, one or more of the assumptions can be revised to add extra functionality to the system and make it respond to a variety of terrain types.

The mobile application was developed using the Unity Engine due to its ease of use and support for multiple platforms including iOS and Android. Google Tango has supported API for Unity which makes it very easy to integrate Tango functionality into a Tango supported mobile device.

Utilising Unity

Although the Unity version 2017.2.1, at the time of writing this report, is the latest version of the Unity game engine, the Unity version 5.3.2f1, which is commonly known as Unity 5, was used for the project because it supports Google Tango API. Unity is a cross-platform proprietary game engine with a built-in Integrated Development Environment (IDE) developed by Unity Technologies. Unity game Engine started as an OS X game development tool in 2005 and is currently supporting an assortment of different platforms that include but is not limited to Windows, iOS, OS X, Linux, Android, BlackBerry 10, web browsers, Xbox, PlayStation, PlayStation Vita, Windows Phone 8 and Wii U.

The game engine is available in two versions namely Unity Free (also known as “Unity Indie”) and Unity Pro. The Unity Pro provides the complete feature set of the Unity 3d engine at a fee which can either be paid at full or in the form of monthly subscription. The Unity Free, on the other hand, is completely free but limited in features and the applications developed with Unity Free show a splash screen with Unity logo on start-up.

OpenCV

OpenCV was used for object tracking because it usually has a lower computational cost as compared to object detection. A tracking algorithm tracks an object that was detected in a previous frame and based on its previous location, direction of movement and velocity, by comparing two or more consecutive frames. Since the tracker can utilise data gathered in the past about an object to predict its location in the current frame, it only has to search in a small area around the predicted location to locate the object. An object detection algorithm, on the other hand, will search the entire frame every time to detect the object. Even though currently, only the tracking algorithm has been implemented, an object detection algorithm will also be implemented to facilitate the tracking algorithm in case the tracking is lost.

To track the robots, object tracking algorithms in OpenCV were used. To find a suitable object tracking algorithm, the performance of different object tracking algorithms was tested on a computer and the tracking ability as well as the framerate were observed for each tracker. The observations are as follows:

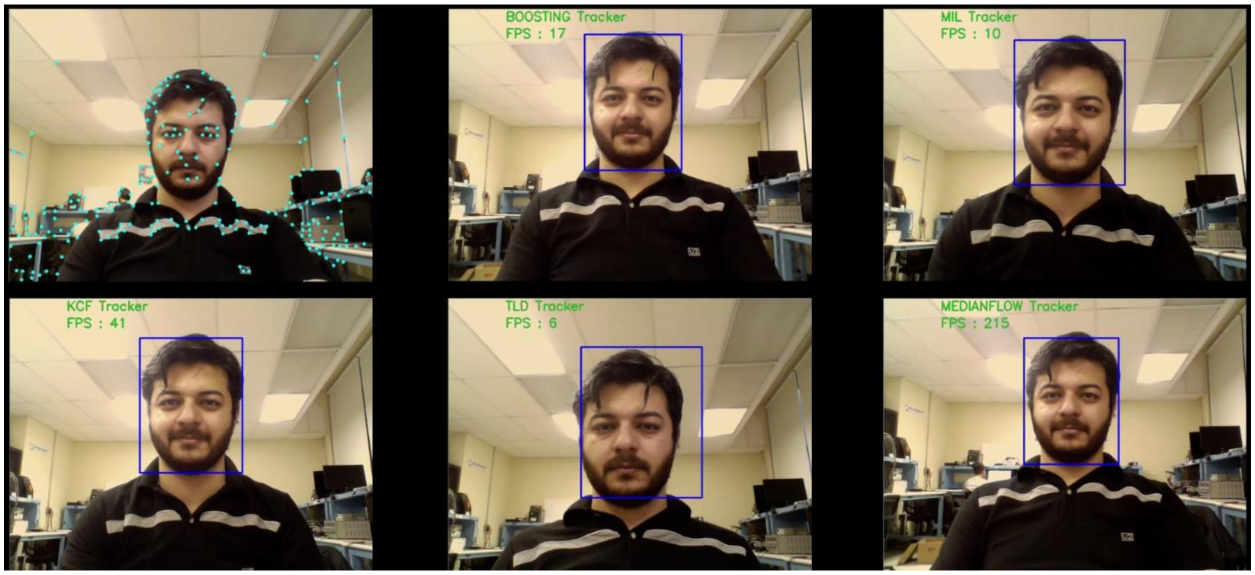

Figure 7. Different methods of Object tracking tested.

Boosting tracker

It is based on an online version of another tracker called AdaBoost, which is the algorithm that the HAAR cascades used for face detection are based on. The Boosting tracker is a classifier that trains itself at runtime with positive and negative examples of the object. The first image with a bounding box around the object supplied by an object detection algorithm or the user at runtime, is taken as a positive example of the object, and several images outside the bounding box are treated as negative examples. As more frames come in, the classifier is run on the pixels near the previous location of the object and the location with the highest score is chosen as the location of the object.

The tracker is not very reliable and upon testing, did not give promising results when it comes to tracking occluded objects and knowing whether tracking has failed or not.

Multiple Instance Learning Tracker

The Multiple Instance Learning (MIL) tracker works by considering the original object location as well as locations in its immediate vicinity as positive examples of the object. This means that not all positive examples might contain the object. However, due to many positive examples, there is a high chance that at least one of them will contain the object.

The tracker did not respond well to changes in the scale of an object and became increasingly inaccurate as the object went farther away from the camera and them came closer again.

Kernalized Correlation Filter

The Kernalized Correlation Filter (KCF) tracker build up on the ideas from the previous trackers and used the idea that the multiple detections by the MIL tracker create an overlapping region. This leads to optimizations that the tracker performs to get reliable tracking.

The tracker performed a lot better than the trackers discussed previously. The bounding box was centred around the object and it only lost tracking once the object was occluded or went out of the frame. KCF is one of the candidates that can be used for this project, mainly due to its accuracy and high performance.

Track Learn Detect Tracker

The Track Learn Detect (TLD) tracker works by tracking the object and the detector checks for the objects and corrects the tracker at every frame. The learning takes this error data and uses it to avoid errors in the future.

TLD gave the best results in terms of tracking accuracy, however the frame rate was very low as compared to other tracking methods. This might be a promising a candidate, but the low performance might make it unsuitable to be used on a mobile device.

Median Flow Tracker

The Median Flow tracker works by tracking the object in both the previous and the next frame to estimate the trajectory of the object which helps the tracker to reliable report tracking failure. The tracker can reliably track objects as long as their trajectories are predictable. The Median Flow tracker gave the best performance results out of all the trackers previously mentioned and along with an object detector, might be the best option to be used in the project.

Conclusion and Future Work

The project so far has considered several object-tracking methods and among those, identified the suitable candidates that can be used on a mobile device. The localization performed through the Google Tango device gave reliable results and the experiments proved that the system is robust. The communication between the robot and the mobile application also proves to be working without any issues. Almost all the building blocks of the project are ready to be put into place except for the object detection and integration of OpenCV inside Unity. Once these two problems are solved, the project can be brought to completion. This, however, does not mean that improvements cannot be made. There are a few ideas and possible improvements that can be made to the system to make it easier to use and work over a number of mobile devices.

Augmented reality solutions like AR Core (Android) or AR Kit (iOS) can be used instead of Google Tango because they do not require specialized equipment to detect surfaces and localize themselves in the environment. AR Core and AR Kit use the inbuilt camera and Inertial Measurement Unit (IMU) of the mobile device to track features in the environment and track its position and orientation. Furthermore, a user friendly Graphical User Interface (GUI) can be developed that will reduce the learning curve of the system and provide an increased overall efficiency when using the system.

References

[1] Y. D.-M. S. G. Lee and M. Egerstedt, "Multirobot control using time-varying density functions," *IEEE Transactions on Robotics,* vol. 31, no. 2, pp. 489-493, 2015.

[2] M. Fiala, "A robot control and augmented reality interface for Multiple Robots," in *Canadian Conference on Computer and Robot Vision*, Canada, 2009.

[3] J. A. Frank, S. P. Krishnamoorthy and V. Kapila, "Toward Mobile Mixed-Reality Interaction With Multi-Robot Systems," *IEEE Robotics and Automation Letters,* vol. 2, no. 4, pp. 1901-1908, 2017.

[4] J. A. Frank and V. Kapila, "Using mobile devices for mixed-reality interactions with educational laboratory test-beds," *Compuers and Education,* vol. 138, no. 6, pp. 2-6, 2016.

[5] B. K.P.Horn and B. G.Schunck, "Determining Optical Flow," *Artificial Intelligence,* vol. 17, no. 1-3, pp. 185-203, 1981.