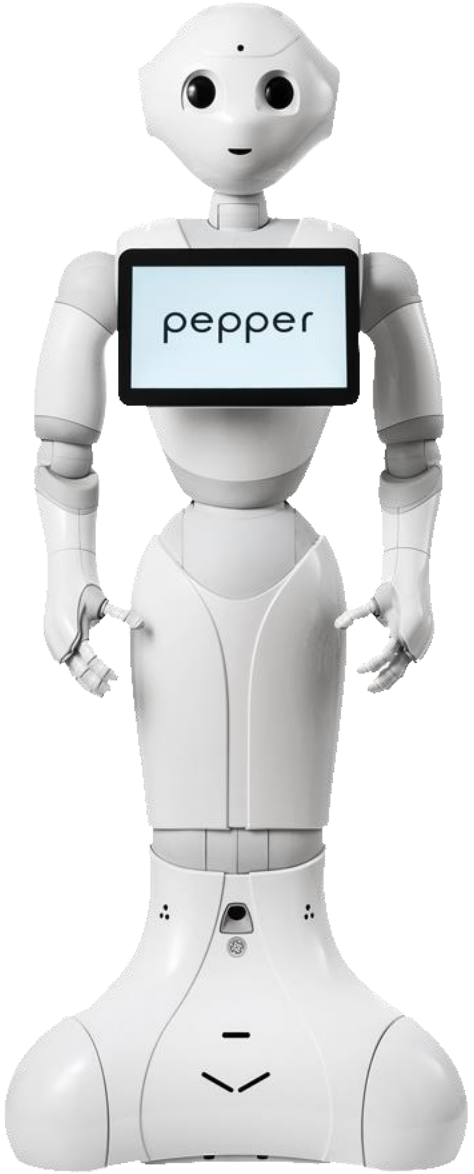
My Projects

Aditya Sivaraj

Telepresence

Bachelor’s Thesis Project



Broad Objective

- Sweden has a problem. They are not able to give adequate care to their senior citizens in nursing homes.
- The whole department at this university were working on deploying robots to lighten the load of doctors and nurses at these facilities.
- Their robot of choice at that point was Pepper Robot, by Softbank Robotics.

My Objective

- To develop a telepresence setup and implement it in the Pepper Robot.
- This setup must be done without the use of their proprietary software, language or IDE. Because it must be available to all and should be versatile.
- Hence, I was supposed to it in all open source code, including, Python, ROS Libraries, Qt, Gstreamer, etc.

Summary of My Work

- Created Video and Audio pipelines for both the respective data to be transmitted and received by the server and client, using the Gstreamer media library.
- Setup controls for the client to be able to move the robot including the ability to set the angle and speed of movement, using ROS libraries.
- Went around the manufacturer set firewalls of being able to access the robot's root only via their proprietary software, using a modular program file in the proprietary language.
- Made a basic Graphical User Interface (GUI) to put all my background work together for the client.

Cart Pole – Model Predictive Control

Course Project

Cart Pole Problem

- A cart pole setup is a standard cart with 4 wheels with an inverted pendulum on top.
- The goal of the problem is to let the pendulum go in an upright position and move the cart so that the pendulum never falls down.

Model Predictive Control

- It is suboptimal control algorithm that combines features of many other primitive control algorithms and is considered a significant improvement over the standard or popular Linear Quadratic Model.
- The short hand version of the algorithm is this: at each and every discrete point defined in space, the model calculates the best possible trajectory or rather solves an optimization problem with respect to that point as if that is the starting point.

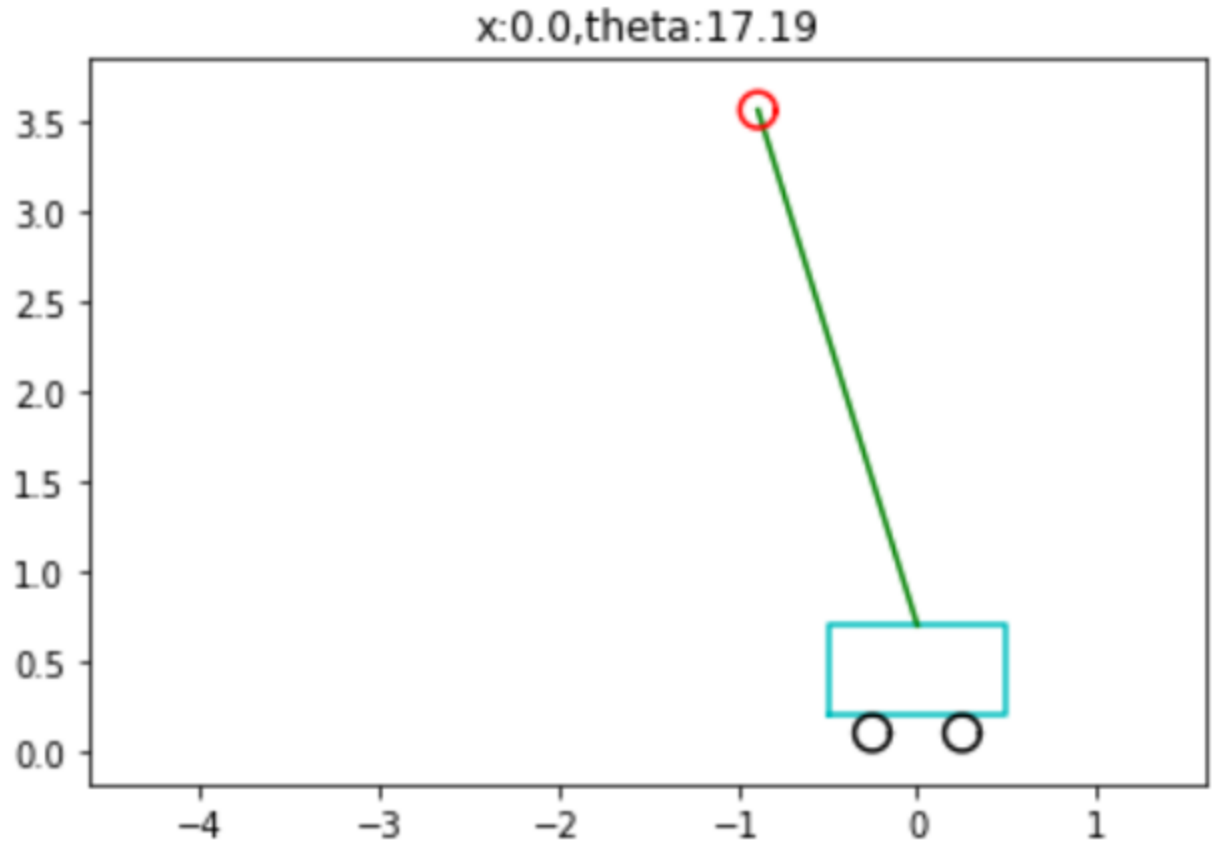
Project Objective

- The goal of the problem is to let the pendulum go at a random state and the cart should be controlled until a certain time so that the pendulum does not fall down.
- Should use Model Predictive Control method to perform the control calculations. Should write code for the same.
- Should print a rudimentary picture of the same for every time differential and final make a video out of all the frames.

Summary of My Work

- Set the initial global values and set the initial state to a random state subject to the constraints of the problem, including length of pendulum pole, masses, acceleration, number of states, number of inputs, horizon length and the time differential.
- Let the cart go from aforementioned position and calculated the state and horizon at each time differential.
- Was not able to put everything together as a video, but printed out every frame instead.

```
calculated time:0.2008724212646484 [sec]
```



A Frame from the Output

Video Gesture Recognition

Course Project

Project Objective

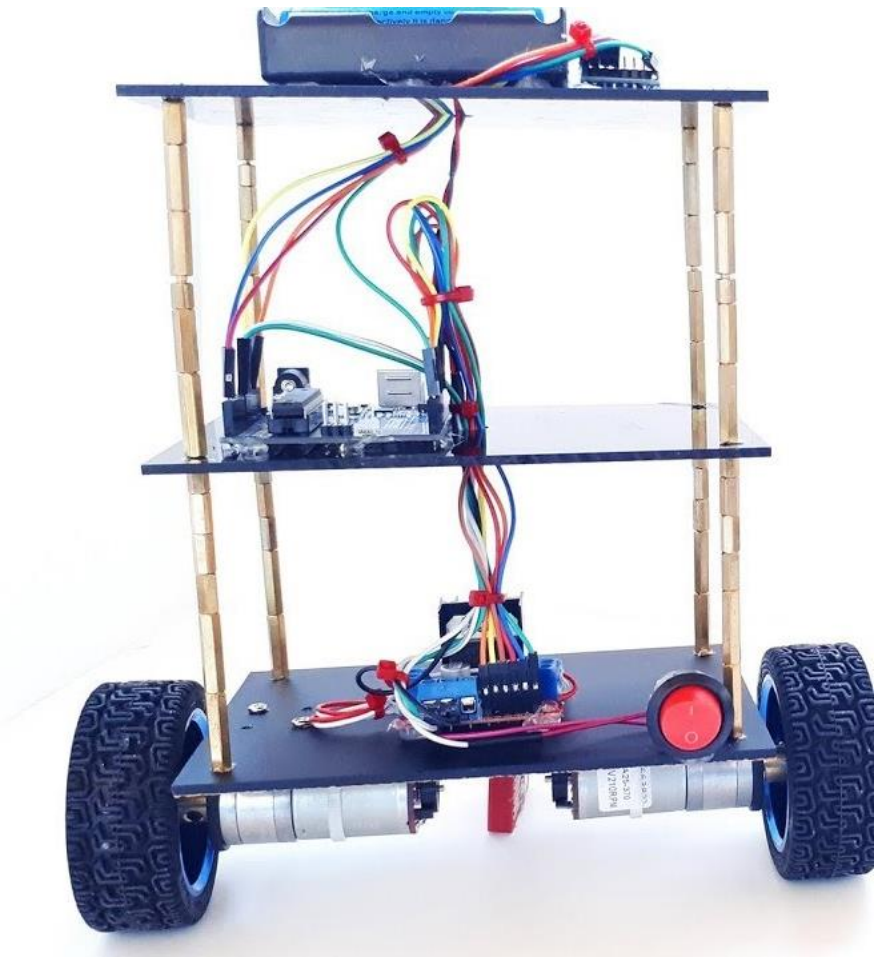
- To create a module which recognizes gestures made by the user via a live video feed and performs actions based on the gesture. In this case, the actions perform media control.
- Should use a Raspberry Pi 3B to take input of the live video feed and process it to recognize the gestures and perform media control actions based on it.

Summary of My Work

- Wrote code for media control and how to manage the gestures.
- Tried writing code and training the ML model for gesture recognition using OpenCV libraries to get a series of images at all possible different angles for different gestures. The trained model did not work as well as expected due to a lack of enough sample size in different environments. Also, the computing power on the Raspberry Pi was not enough run the model trained by me.
- Used an open source trained model for the required gestures and integrated it into my module.

Self Balancing Robot

Summer Internship Project



Picture not from my project, but, is similar to this.

Project Objective

- The goal of this project is to make a cuboid structure about a foot tall with wheels on either side stand without falling.
- The computer on the structure, and the sensors on it should foresee when it is about to fall and in which direction. After which, the bot will move in that direction to avoid the fall.

Sensor Used

- We used MPU6050 sensor (6-axis IMU), which is actually a combination of two sensors, namely, an accelerometer and a gyroscope. It also houses a Digital Motion Processor.
- Accelerometer senses the angle of tilt in 3 axes.
- The gyroscope in here is a 3-axis one which senses the orientation and the angular velocity.
- The sensor gives a 6 value output with a 3 axis value from each sensor.

Summary of Our Work

- Built the setup with plastic boards, steel rods and toy car wheels.
- Used Arduino to do the processing. Wrote code in C for the whole setup.
- Interfaced MPU6050 sensor and the motors with Arduino.
- Calibrated the sensor to the values that suited our project. This took the longest time, as the sensor was highly sensitive, and the six values did not coordinate well.

Other Projects Done (Non-Exhaustive)

- Path planning of a SCARA Manipulator using MATLAB.
- Performed SLAM on a virtual quadrotor using MATLAB.

Work Projects

Autonomous Mobile Robot (Tespa Robotics)

- Created stack and built the circuit of an AMR from scratch
- Worked extensively on ROS1 and ROS2.
- Worked with LiDARs, Ultrasonic Sensors, Rotary Encoders, ToF Sensors, etc., and created the stack surrounding them.
- Created power circuits for the AMR.
- Worked with various SLAM, localization, mapping, cost mapping and path planning algorithms.

Autonomous Vehicle (Minus Zero Robotics)

- Worked extensively on Sensor Fusion, Path Planning, Global Planner, Collision Avoidance and Vehicle Control.
- Worked with various GNSS modules, industrial grade 9-axis and 6-axis IMUs, Camera Modules, Hall Effect wheel encoders, etc.
- Additionally worked with websockets to create a data visualization stack.
- Worked extensively on C++ sensor fusion algorithms, path planning and control algorithms. And extensively in Python for in house experimental algorithms.
- Worked extensively with git version control.
- Worked with ROS2, eCAL, Protobuf, etc.