PESU I/O Course Plan

General instructions:

1. The course plan caters to first years, so should start from the **very basics** of the course and go **extensively covering course content in great detail.**
2. The PESU I/O courses are divided into 3 weeks, and by the end of the course, it is expected from the students to come up with a **final project**, related to the course. The product should be **built consecutively as the days progress**. The product they will be coming up, needs to be **specified by the SME at the beginning of the course**. For Example: A dance course can teach your certain steps in each week, but the final product should be a well choreographed sequence comprising of all the steps taught, or an Algorithms course can teach you certain algorithms each week, but the final product can be a software implementing those algorithms.
3. Each day starts with the SME explaining the tasks of the next couple of days. This is followed by the next day at home, where the students, on their own,without the SME, work and discuss with their peers. Next day, we have a SME Meetup, where the students meet and discuss with the SME any doubts, teach any topic if the majority of the class want it to and evaluate students based on their performance in the assignments and SME Meetup. Then followed by the next topic task assignment. The SME meets will happen on 9 days out of the 15.
4. The students will have to submit their project by the end of the Course as a presentation video, which will be assessed by the SME after the course ends.
5. The Course Plan should be divided into 9 days. Each day should contain the following mandatorily: 1. Topics to be taught. 2. Assignments/Quizzes/Other weekly tasks of the day. 3.Daily Project Work to be done, which will build towards the final product. You should specifically outline what your students will accomplish with this course and what will be the final product your students will be making.

Integrated Robotics and Applied Computer Vision – Course Plan

Instructor Name : Sriram Radhakrishna, Samuel Thomas

Branch : CSE, ECE

Semester : 5

Course Duration: 30 Hours (18 hours of in-person mentoring + 12 hours of self learning)

Prerequisite for the course:

1. Knowledge of basic calculus and graphical transformations.
2. Python programming language.
3. Familiarity with object oriented programming concepts.
5. Knowledge of basic physics
4. Software requirements : Ubuntu 20.04 Operating System (dual boot), VSCode w/ Jupyter extension, ROS Noetic

Deliverable from the course:

1. Understand the data structures, mathematics and color theory behind image processing and object recognition.
2. Mastery of image processing & display techniques using OpenCV.
3. Application of Image processing in real life scenarios.
4. Understand the working of a
5. Application of hand tracking & pose estimation in real life.
6.Get a foundational understanding of ROS

7. Be able to use a simulation software like gazebo for testing
8. Implement and deploy motion planning in the form of trajectory tracking and path planning using controllers like the PID Controller

Final Project:

The students can choose one of the following problem statements - Gesture based computer control, complete backend vision pipeline for classification on a mobile app, path planning & pedestrian avoidance software stack for a ground robot, exercise assistant and Mediapipe/Unity3D interfacing, Development of an autonomous drone that moves in a set trajectory and uses a camera for image processing. Students can also choose their own problem statements provided all topics are covered.

Note : Unless mentioned otherwise, all points mentioned under the topics to be taught on each day are conversations with the students. Hands on activities are marked as such.

Day 1 – Mathematical fundamentals applied in image

Processing and Basic Software Engineering Techniques; Software Installation and set up of ROS

1. Topics to be taught:

- Bird's eye view of the field of Computer Vision and an outline of the direction to take.
- Image, video & webcam feed loading in OpenCV [HANDS ON]
- OpenCV documentation review (alongside the following sub-topics)
- Git introduction
- Intro to ROS
- Ensuring software is correctly installed with all dependencies working [HANDS ON]
- Environment setup wrt Catkin Workspaces and Git clones of necessary packages [HANDS ON]

2. Tasks to be completed:

- Team formation
- Get familiar with git clone, add, commit, push, branch, checkout, pull requests
- Creating a catkin workspace

3. Weekly Project Work.

- Set up all the necessary prerequisites : Ubuntu dual boot, ROS, OpenCV.

Day 2 – Introduction to OpenCV and ROS

1. Topics to be taught:

- Kernels and their usage in dilation & erosion.
- Grayscaling, gaussian blur & the reasons to use them. [HANDS ON]
- Matrix rotation transform demonstrated using warp perspective.
- Inserting shapes & text (to emphasize the visual attractiveness of computer vision software) [HANDS ON]
- Matrix stretch transform demonstrated by image slicing & resizing
- HSV color space & cylindrical representation demonstrated by color detection & masking.
- Laplacian operator in HSV space demonstrated by Canny edge detection.
- Contour detection
- Image stacking & trackbars
- ROS topics overview, publisher and subscriber

2. Tasks to be completed:

- Shape differentiation program
- HSV limit finder program with trackbars & stacked images.
- Deploying a ROS node, and publishing and subscribing to a topic.

3. Weekly Project Work.

- Turn prediction software for a video feed from a line following robot.

Day 3 – Introduction to Neural Networks, Gazebo and Project Setup

1. Topics to be taught:

- What is a neuron?
- What is a neural network?
- A conversation on summation in NNs
- A conversation on gradient descent
- A conversation on activation function
- A conversation on forward propagation & backpropagation
- ROS services [HANDS ON]
- Creating custom messages [HANDS ON]
- Introduction to TurtleSim

2. Tasks to be completed:

- Quiz on the conceptual understanding of the above
- Drawing a basic shape in Turtlesim using teleop node

3. Weekly Project Work.

- MNIST classification using ANNs (ref. : https://www.kaggle.com/code/rupeshs/mnist-baseline-without-cnn/notebook)
- Creating a node to publish a shape to TurtleSim that will draw it

Day 4 – Hector Quadrotor:

1. Topics to be taught:

- Gazebo Setup
- Using RViz to create visualizations in Gazebo [HANDS ON]
- TurtleSim control with nodes
- A brief conversation about convolutions in neural networks (as a break from the robotic systems)
- Using teleop and nodes for motion control
- Implementing laser scan and cmd_vel topics for pose estimation and obstacle detection

2. Tasks to be completed:

- Experiment with the honey bee pollen dataset for binary classification (ref. : https://www.kaggle.com/datasets/ivanfel/honey-bee-pollen)
- Installing dependencies and gaining motion control of the quadrotor drone
- RViz graph creation

3. Weekly Project Work.

Pick any classification problem on kaggle, train, test and visualize the losses and accuracies.
Writing a ROS Node to move the quadrotor in a specified manner

Day 5 – Transfer learning and playing around with AWS EC2, Understanding PID Controllers:

1. Topics to be taught:

- What is transfer learning in the context of neural networks?
- A case study of MobileNetV2
- Booting up an AWS EC2 instance [HANDS ON]
- SSH controls
- Building pipelines
- Error Control, Rise time, Peak overshoot
- P, PI, PD and PID controllers
- PID tuning using turtlesim

2. Tasks to be completed:

- Boot up an EC2 instance.
- Deploy a PID Controller
- TurtleSim PID control

3. Weekly Project Work.

- No project work this session. Take a break.

Day 6 – Hand Tracking & Pose Estimation, Deployment of Motion Control:

1. Topics to be taught:

- Segway into mediapipe as the deep learning section of computer vision
- Introduction to mediapipe with neural networks & convolution at a high level.
- Emphasis on mediapipe as a collection of ML solutions and not models (hence the light processing load), mention of bounding box classifiers.
- Mediapipe documentation review
- Landmarks
- Hand tracking module [HANDS ON]
- Pose estimation module [HANDS ON]
- Using PID to control motion
- Deploying a ROS node to control motion in a pre-set manner while integrating PID controllers [HANDS ON]
- Laser Scan integration for obstacle detection
- Hector quadrotor understanding

2. Tasks to be completed:

- Program to count the number of fingers on screen.
- Understanding motion of hector quadrotor using teleop

3. Weekly Project Work.

Literature survey for the final project.

Day 7 – Computer Vision in Robotics, Trajectory Tracking and Path Planning

1. Topics to be taught:

- Running a CV algorithm on a ROS node
- Implementing PID in hector quadrotor

2. Tasks to be completed:

- Spawn a turtlebot.
- Spawn a hector quadrotor
- Demonstrate coke can detection on Gazebo using the quadrotor
- PID tuning in hector quadrotor
- Integration of the motion and vision stacks

3. Weekly Project Work

- Demonstrate coke can detection on Gazebo using the quadrotor

Day 8 – Project work

Tasks to be completed:

- Have an execution plan for the project.
- Have a codebase ready for the same.
- Have a specific set of targets to achieve to define the project's completion.
- Complete 65% of the project with respect to the total number of lines of code.

Day 9 – Project Presentation:

Tasks to be completed:

- Students present their final projects
- All projects must be evaluated by the SME