After First years' end sem

We would be resuming our teaching sessions from **1st July 2023**

The following sessions will be taken at 9 pm

1. 27th june - introduction meet explaining the workflow
2. 1st July - IOT - Shashank
3. 4th July - Kinematics - Shaswat
4. 6th July - AI/ML - Piyush Mishra
5. 9th July - control systems - Aniket
6. 12th July - Ros Installation - Shailesh

A. After every session, everyone has to complete a **Basic Task** within 3 Days until next session starts (Submissions would be done before the commencement of the next session)
B. After the tasks and sessions are over, everyone has to pick up a **Main Task.** You can choose *any 1* of the tasks, which are listed under the main task section and submit it by **30 july 2023**
C. You all are requested to cooperate with us and convey your engagement for the holiday month beforehand to your friends and family
D. Do keep the sessions interactive and reach out to your seniors in case of any doubt

Basic Tasks (compulsory to complete by next upcoming session) -

1. 27th June - CAD - Make a 4 wheeled robot chassis. Use your creativity!
2. 1st July - PCB Design Task - Design a PCB for ne555 Monostable circuit.[Reference https://www.circuitbasics.com/555-timer-basics-monostable-mode/]
3. 4th July - MATLAB - Complete Matlab Onramp course (First years are encouraged to download Mathworks MATLAB software, not compulsory though)
4. 6th July - AI ML - Basics of numpy and pandas, Supervised Machine Learning (by Andrew NG, on coursera) : Week 1
5. 9th July - OpenCV - Write a python script to find angles using OpenCV.
6. 12th July - ROS installation - Install Ros in your laptop

Main Tasks (Submit on 30th July, choose any 1 task)

1. Computer Aided Design Task -

- Design a *Robowars Bot* in one of the categories-

A. Large - dimension 750mmx750mmx750mm
B. Small - dimension 400mmx400mmx400mm

- Use proper electronics components in the bot
- Bot can be wired or wireless

- Use GrabCAD for borrowing electronics (Don't borrow the entire bot we will get to know. You can take inspiration from existing bots)
- Following methods are **NOT** allowed-
    1. Flying (using airfoil, helium balloons, ornithopters, etc.) is not allowed.
    2. The robots should not secure themselves on the ring surface by using suction cups, diaphragms, sticky treads, glue or other such devices.
    3. Flame throwing or any kinds of inflammable substance ejection
    4. Liquid projectiles, Foam, liquefied gases
    5. Acid/corrosive substance based weapons
    6. Weapons causing invisible damage (Electrical weapons, RF jamming weapons and others).
    7. Weapons causing opponents' weapons (spinners) to entangle in them (Chains, Ropes or loose Fabrics).
- Bot will be judges on -
  a. Manufacturability
  b. Mobility
  c. Robot Control Requirements
  d. Battery and Power
  e. Pneumatics and Hydraulics if used
- Tip - Don't use wood/plastic. Use sheet metal with/without metal cast or machined body. Can use steel rods as truss frames as the skeleton of the bot
- Best design "might" be selected for manufacturing, for one of the heavy bots team
- Submit a zip folder with all the components and the assembly file if using Solidworks
- Or submit a step file if using other CAE softwares
- ★ PS - A design Report will be appreciated. NOT compulsory to make
  Sample design Reports -
  https://drive.google.com/drive/folders/1Fxod_rZRNqe9GhTEwaoHLMWOVAivYGMV?usp=sharing
  It's okay if you don't make it..CAD Design is enough

2. OpenCV Task -

You have to complete **both** task A and B

Task A-

Draw the line pattern as shown in "Reference Grid.png" image on the "Task.png" image.(Both the images are in the drive link given)

Task B-

Your task is to place the given 4 marker images ["Ha.jpg", "HaHa.jpg", "XD.jpg", "LAMO.jpg"] on the boxes of "CVTask.png" image as given below :

Box Color	Marker ID
Green	1
Orange	2
Black	3
Pink-Peach	4

The marker should exactly be placed on the squares only, even the orientation of the squares should be kept in mind.

Images of both the tasks - https://drive.google.com/drive/folders/1a1penjeyqyhaZ4bCBaKr9ne20IRM3IRD?usp=drive_link

Submission in form of zip folder containing both the final output images and code

3. MLTask -

Apply logistic regression on the given dataset:

There are two files namely:-
train_data.csv :- Typically a data set of 792x16 . The **survived** column is your target variable (The output you want to predict).

test_data.csv :- A data of 100x16 , for testing your model , The arrangement of **test_data** exactly matches the **train_data.**

Train data:
https://drive.google.com/file/d/1XRy-xzMfdu4aalqZzOqJqYqbrsugPF6u/view?usp=drive_link
Test data:
https://drive.google.com/file/d/1ZPzDjyQmGhPa2T_mgao1symo5d8-pMUO/view?usp=drive_link

Represent the cost v/s no. of iterations graphically using matplotlib/seaborn.

NLP Task :- Perform sentiment analysis on comments of “Avira Password Manager” APP. Collect the comments from Google Play Store via scraper. Analyse and Visualise the data properly and display the output using matplotlib or seaborn. Add the results with the corresponding comments and save the file as “result.csv”.

Submit the link to your github repository containing the code(s)(.py/.ipynb)and the datasets used.

4. Electronics task -

- **Make a device to control home appliances remotely using ESP32/8266, RF module, NRF module(Any)**
- **Specification :**
    - PWM for speed control
    - Control speed of AC appliances from microcontroller via appropriate circuit
    - 4 switches on Transmitter side and 4 relays on Receiver side
    - To display PWM a LED can be used on receiver side
    - Timer for sleep

- **Submission** should be on a perforated copper dot Pcb or on a Breadboard but a fully designed PCB should be submitted in PDF/PNG/SVG format.

Transmitter	Receiver
● 4 Push Buttons ● Battery 5V (3 AAA battery) ● NRF module/ RF module / ESP32 or 8266 ● Potentiometer for PWM ● Potentiometer for Timer (Ex : if Knob is 10% rotated then switch off Main lights after 1 min, if 20% then 20 min and so on)	● Receiver ● 4 Relays ● Transistors for Relays(Don't forget Diodes) ● LED for PWM demonstration ● Batteries, better if it can be powered by a generic mobile charger 250 mA

5. Robot Path Planning Task -

Aim: Robot Path Planning Algorithm with Simulation Environment

Description:
You are tasked with developing a path planning algorithm for a robot in a simulated environment. The goal is to enable the robot to navigate from a starting point to a target location while avoiding obstacles. In addition to the algorithm implementation, you are required to create a simulation environment to test and visualise the robot's path planning behaviour.

Requirements:

1. Path Planning Algorithm:
   - Choose a suitable path planning algorithm (e.g., A*, Dijkstra's algorithm) to find an optimal or efficient path from the robot's starting point to the target location.
   - Implement the algorithm using a programming language of your choice, considering the robot's current position, the target location, and obstacle information.

2. Simulation Environment:
   - Select a simulation framework or tool (e.g., ROS) to create a virtual environment for the robot.

- Design and build the simulated environment, including obstacles, boundaries, and any relevant features or landmarks.
- Integrate a robot model into the simulation environment, ensuring that it accurately reflects the physical characteristics, sensors, and actuators of the real robot.

3. Sensor Simulation:
   - Simulate the robot's sensors within the simulation environment to provide data for obstacle detection and environment perception.
   - Ensure that the sensor readings and measurements align with the capabilities of the real robot.

4. Integration and Visualization:
   - Connect the implemented path planning algorithm to the simulation environment.
   - Provide inputs to the algorithm based on the virtual robot's state and environment information, and retrieve the planned path as an output.
   - Set up visualisation tools to observe the robot's behaviour, visualise the planned path, and monitor the interaction with obstacles during simulation.

5. Performance Evaluation:
   - Evaluate the performance of your path planning algorithm within the simulation environment.
   - Measure and analyse metrics such as path length, computation time, and success rate of reaching the target.
   - Document and present the results of the evaluation, comparing different algorithm variations or testing multiple algorithms.

Deliverables(What we need):

1. Documented implementation of the path planning algorithm, including a clear explanation of the chosen algorithm, data structures used, and integration with the simulation environment.
2. Simulation environment setup, including the virtual environment design, integration of the robot model, and sensor simulation.
3. Visualisation of the robot's behaviour and planned paths within the simulation environment.
4. Performance evaluation report, including analysis of the path planning algorithm's effectiveness based on the defined metrics.

Submission should be in the form of a PPT/Doc file along with the GitHub repository link.

6. Robotic Arm Task -

Aim: Inverse kinematics solver for a Robotic arm

Background: A robotic arm with multiple degrees of freedom is required to perform precise manipulation tasks in various industries such as manufacturing, assembly, and automation.

In order to control the arm's end effector position and orientation accurately, an inverse kinematics solver needs to be developed.

Objective: create a simulink model of a 3dof robotic arm for detecting a path to object and tracing it to reach the desired object (if you think that's a very big task, at the end we could give you the detected path, try to trace it to reach object from that path. But try on your own).

Requirements:

1. Robotic Arm Model: create a very basic model of a 3dof robotic arm and import it to simscape using **simscape multibody link** that is present in solidworks.
Model can be very basic with two revolving joints. And a base with two links.
(It's the most important step do it carefully)


something like this. Or you can grab something from grabcad.

2. Forward Kinematics: Implement forward kinematics equations to calculate the position and orientation of the end effector based on the joint angles of the robotic arm. Add a block for forward kinematics. You can find various formulas for forward kinematics of the 3 dof robotic arm, and use them to model a simulink block.

3. Inverse Kinematics Solver: Develop an algorithm to solve the inverse kinematics problem. Given a desired end effector position and orientation, the solver should compute the corresponding joint angles required to achieve the desired configuration.add a inverse kinematics block. (jacobian method equations preferred ,easy to model)

4. You can add some PID blocks for better accuracy **LIMITATIONS**: only two PID blocks are allowed

Deliverables(What we need):

1. MATLAB code implementing the inverse kinematics solver and forward kinematics solver for the robotic arm.
3. Trajectory plots. (If you use given trajectory(at the end) your final trajectory should be accurate to it)
2. Documentation describing the design, implementation, and usage of the inverse kinematics solver.

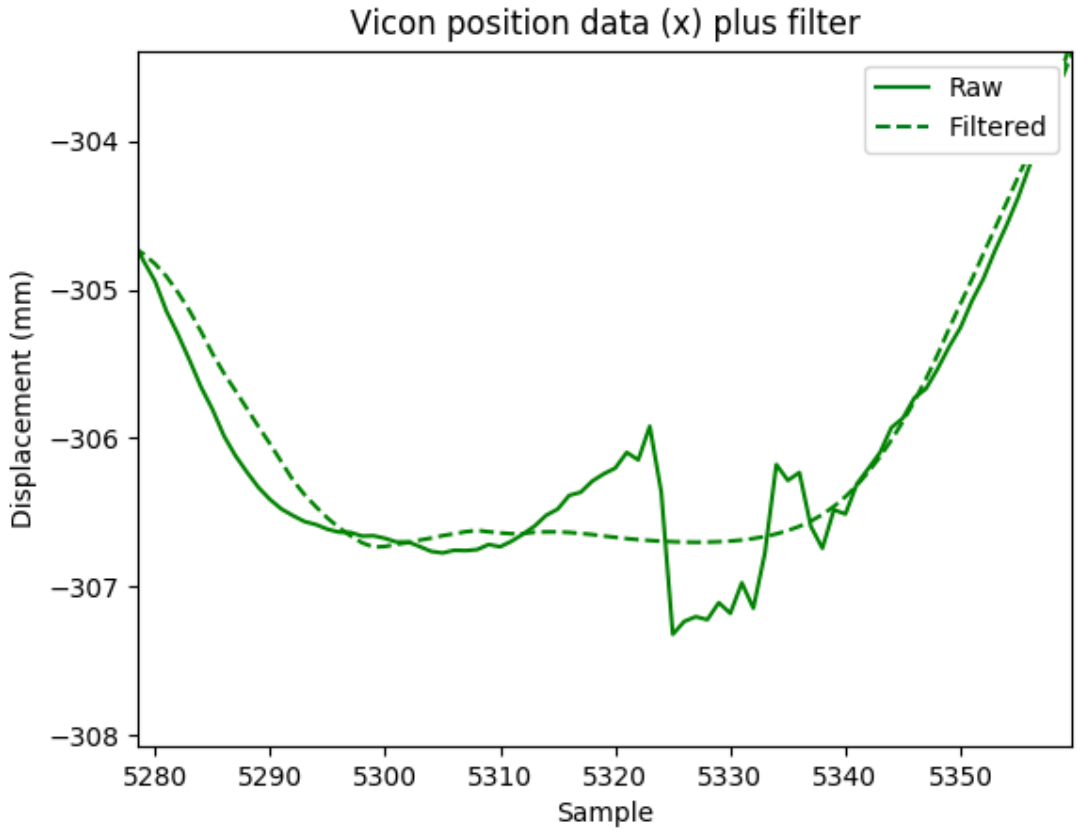
Note: The project can be extended to include additional features such as trajectory planning, obstacle avoidance, or integration with a physical robotic arm for real-time control.

For the ones who require an already detected path use/import the given simulink block.your model should accurately trace the given path.
Link: simulink_block

Submission should be in the form of presentation with mentioned attachments all together in github repository

7. Sensor Fusion Task -

Aim: Understanding Sensor Fusion
Pre-Requisites : Purchase a MPU6050(6DOF) or MPU9250(9/10DOF).
Background: While getting real life sensor's data, individual sensors produce noisy data and hence can't be used for our objective. Here comes sensor fusion, which uses multiple sensors(same or different kind) to reduce noise and make our data useful. Gyroscope and accelerometer data is often fused to produce smooth and accurate Roll,Pitch and Yaw values.



Example

Objective: Use a MPU6050(6DOF) or MPU9250(9/10DOF), extract individual data from each sensor(namely gyroscope, accelerometer, magnetometer(optional)) then write sensor fusion algorithms to get smooth and accurate Roll,Pitch and Yaw values.

Steps:
1. Get a MPU6050(6DOF) or MPU9250(9/10DOF).

2. Hook up with any microcontroller board(Any Arduino board, Raspberry Pi, Teensy, Ordroid, Nvidia Jetson nano, etc.).
IMPORTANT: Calibrate your sensor.

3. Extract individual sensor data through I2C communication.

4. Use sensor fusion algorithms or rather 'filters' to get smooth and accurate Roll,Pitch and Yaw values.

Some common filters: *Kalman Filter(Recommended to learn), Complementary filter, Madgwick Filter.*

NOTE: Use of any external libraries is not allowed. Libraries that contain the full algorithms, that should be written by you using mathematical expressions. Use of basic libraries is surely allowed.

Deliverables(What we need):
- Written code.
- Video of demonstration of getting smooth and accurate data on a graph(Serial plotter for arduino).

8. Smart Home Automation System task -

Aim: Develop a comprehensive smart home automation system that utilises ROS for seamless integration of various components and enables users to remotely control and monitor home devices.

Description:
The goal of this project is to design and implement a smart home automation system that leverages the capabilities of ROS (Robot Operating System) to create a centralised platform for controlling and monitoring different home devices and appliances. The system will incorporate IoT, kinematics, AI/ML, and control systems to provide efficient and intelligent automation functionalities.

Workflow:

1. Introduction:
   - Conduct an introductory meeting to discuss the project goals, objectives, and workflow.
   - Assign roles and responsibilities to team members and establish communication channels.

2. IoT Integration:

- Implement IoT devices and sensors to monitor and control various aspects of the smart home, such as temperature, lighting, security, and energy consumption.
- Develop communication protocols and interfaces to enable seamless integration of IoT devices with the smart home automation system.

3. Kinematics and Motion Planning:
- Design and implement kinematic models for robotic arms or mobile robots that can perform specific tasks within the smart home.
- Develop motion planning algorithms to enable the robots to navigate through the environment and interact with objects.

4. AI/ML for Smart Home Intelligence:
- Utilise AI/ML techniques to enhance the intelligence of the smart home automation system.
- Implement algorithms for activity recognition, anomaly detection, energy optimization, and personalised automation based on user preferences.
- Train and fine-tune ML models using appropriate datasets to achieve accurate and efficient results.

5. Control Systems:
- Design and implement control systems to regulate the operation of various devices and appliances in the smart home.
- Develop feedback control algorithms to maintain desired setpoints and respond to changing environmental conditions.
- Incorporate control mechanisms for efficient energy management and optimization.

6. ROS Integration:
- Install and set up ROS on the designated hardware platform or simulation environment.
- Implement ROS nodes and communication interfaces to facilitate seamless integration of the different components of the smart home automation system.
- Establish communication between the IoT devices, robotic systems, AI/ML modules, and control systems through ROS topics and services.

7. User Interface and Interaction:
- Develop a user-friendly interface that allows users to remotely control and monitor the smart home devices and appliances.
- Enable real-time visualisation of sensor data, robot motions, and system status.
- Implement interactive features such as voice commands or mobile applications for intuitive control.

8. Testing and Evaluation:
- Conduct rigorous testing of the complete smart home automation system to ensure its functionality, reliability, and performance.
- Evaluate the system's response to different scenarios and user interactions.
- Measure and analyse key performance metrics such as response time, energy efficiency, and user satisfaction.

Deliverables:

1. Detailed documentation of the system architecture, including the integration of IoT, kinematics, AI/ML, control systems, and ROS components.
2. Source code and implementation details for each module, along with clear instructions for replicating the system.
3. A well-designed user interface for remote control and monitoring of the smart home devices.
4. Testing results and performance evaluation report, including metrics and analysis.
5. Presentation slides summarising the project workflow, contributions of team members, and showcasing the system's features and capabilities.

Submission Format:
1. PPT/Doc file: Provide a presentation or document summarising the project, workflow, and outcomes.
2. GitHub Repository: Share the link to the repository containing the project source code, documentation, and any additional resources.

IMPORTANT NOTE -

- ➔ After task submission, a survey form will be released analysing your interests and you will be allotted your project accordingly. This will also assign each and every individual a particular role and a goal to achieve
- ➔ In the first offline meet, the projects and people will be allotted accordingly. Room will be open for negotiations and change of project allotment.
- ➔ Discussions on the yearly workflow and further scrutiny regarding the members activity shall be done in the same meet
- ➔ After the meet ends, cleaning both RoboISM rooms will be given as an initial task, after which the team leads will take over the project building.

Let's hope we exert immense effort and unlock our full potential. Till then everyone, Think Beyond Infinity !