מועמדים יקרים!

כפי שצוין בכנס המועמדים, עליכם למלא את המטלה (לבחור 1 מתוך ה-4 הקיימות) ולהגישה בצירוף קורות חיים בקישור שיופיע בהמשך, עד לתאריך 03.12 בשעה 23:59.

לאחר ההגשה מועמדים מתאימים יזומנו לראיון אישי.
קישור להגשה: https://forms.gle/Y2Zdghft6CtD527a8

לשאלות נוספות ניתן לפנות אלינו לכתובת המייל: bgracing@post.bgu.ac.il

המשימות מנוסחות בלשון זכר מטעמי נוחות אך מתייחסות כמובן לשני המינים !

בהצלחה!

Home Assignment for Ben Gurion Racing Autonomous Team

Thank you for your interest in joining the Ben Gurion Racing Autonomous Team! Below are the instructions and details for your home assignment. **Please read everything carefully and reach out with any questions. We are here to support you, so don't hesitate to ask—it's better to clarify than leave a task incomplete.**

Contact Information

For any inquiries, please email:

- Gregory Guterman: grisha.guterman@gmail.com

- Roy Amoyal: roy15957@gmail.com

General Instructions for All Assignments

- Open a **public** project on GitHub to host your work.
- Complete the tasks within this project repository and share it with us upon submission.
- Include a README file in your repository. This should provide:
  - Installation and run instructions.
  - A clear explanation of your work and approach, please specify OS you used to complete the task
- Attach your CV and, optionally, a cover letter explaining why you'd like to join the team.

Please select **one of the following assignments to complete**. You're welcome to attempt any task that interests you, or even tackle multiple ones if you'd like. However, the number of tasks completed will not affect the priority.

Assignment 1: Point Cloud Processing

OS Requirements: Mac / Windows / Linux

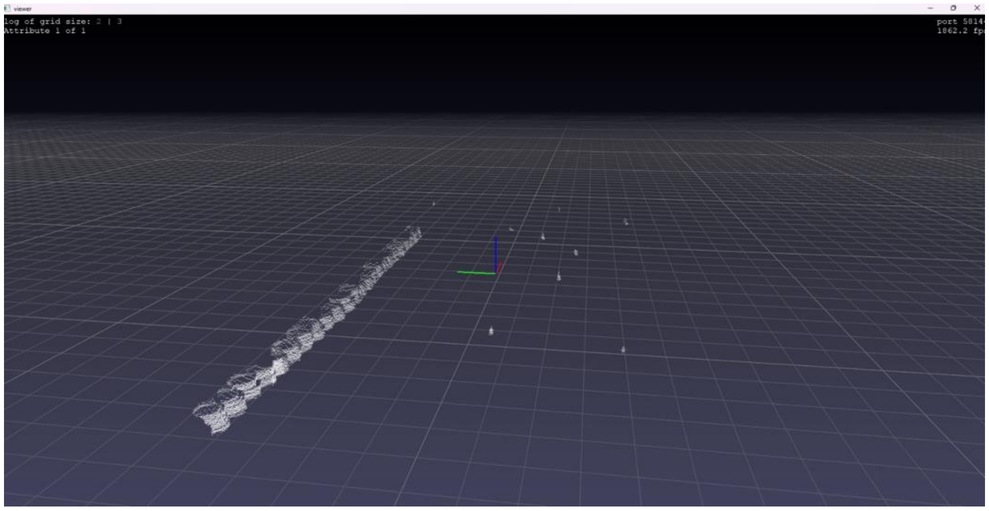
You will be provided with a CSV file containing a 3D point cloud. [link]

Tasks:

- Identify the appropriate False Alarm value to exclude most of the junk points from the point cloud, resulting in a cleaner and more accurate representation.
- Filter out ground points from the point cloud.
- Cluster the remaining points.
- Visualize your process and results using 3D plotting tools.

Bonus:

- Identify and filter clusters representing cones and mark them separately.



Assignment 2: Path Planning

OS Requirements: Mac / Windows / Linux

You will be provided with a CSV file containing points in a 2D plane, where each point represents a cone. [link]

Tasks:

- Design a smooth path for a car to navigate between the cones.
- Optimize the path for minimal curvature while ensuring feasibility for vehicle navigation.
- Visualize your solution using 2D graphics.

Bonus:

- Research the Dynamic Bicycle Model and PID controllers and be prepared to explain these concepts during an interview.

Assignment 3: Cone Detection Using AI

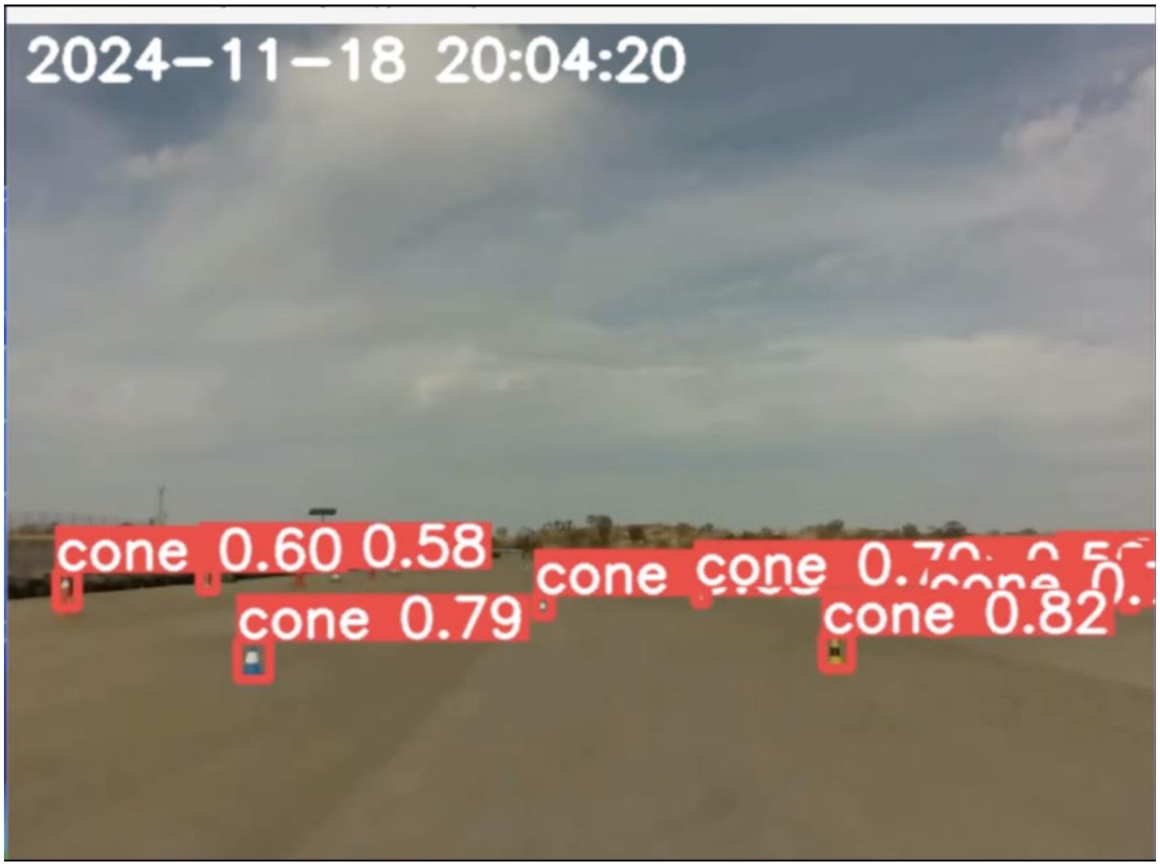
OS Requirements: Mac / Windows / Linux

Tasks:

- Find an open dataset for traffic cones. Suggested keywords: 'traffic cones dataset,' 'Formula Student cones dataset.' A recommended site is Roboflow.
- Use an AI model (e.g., YOLOv5, YOLOv8, OpenCV) to detect cones in the dataset.
- Draw bounding boxes around detected cones.
- Example video you can work on

Bonus:

- Identify the color of the detected cones.



Assignment 4: Simulator-Based Task

OS Requirements: Windows / Linux

If you encounter any installation issues, please contact us for assistance.

Tasks:

- Download and unzip the Formula Student Driverless Simulator. In folder assets of last version (2.2.0) you will find zip files. [link]
- Run the simulator:
  - Windows: Execute the .exe file.
  - Linux: Execute the FSDS.sh script.
  - Press the button "Run simulation" to check if everything works
- Clone the simulator GitHub project:
  - git clone https://github.com/FS-Driverless/Formula-Student-Driverless-Simulator
- Navigate to the Python folder, install requirements.txt, and run "autonomous_example.py", check how it works.
- Adjust the "settings.json" file as specified in the "autonomous_example.py" comments.
- Add a camera to the simulator vehicle by modifying "settings.json" file, consider to use "camera_color_png.py" for a reference.
- Verify the camera addition by running "python/examples/camera_color_png.py".
- Update autonomous_example.py to display a live camera feed (using OpenCV) as the car drives.

  REMEMBER:

  -Use the Airsim Sensors Documentation for guidance.

  -Each time to apply the changes you've maid, you must exit and run simulator again.

  -Do not forget to ask us for help, we are here to teach you.

Submission Requirements:

- Upload only your updated settings.json file and your modified autonomous_example.py.
- Include a README file explaining your setup process and include screenshots or images of your results.

Final notes

This home assignment is an opportunity to showcase your skills and passion for autonomous systems. We’re excited to see your creativity and technical abilities in action. Remember, perseverance and learning from challenges are valued just as much as the results.

Please send us whatever progress you’ve made, but ideally, reach out for assistance if something isn’t working as expected.

Good luck, and we look forward to your submission!