Final Project Instructions

Kaveh Fathian, Email: fathian@ariarobotics.com

Handout: Thu, 2024-04-03
Due: Tue, 2025-04-29

Important instructions:

- The final project should be done as a **team**, not individually.
- The teams must consist of 3-4 students, and they are responsible for entering their group members on Canvas.
- The team will select their team leader, who will assign and coordinate tasks among members, grade them based on their participation and effort, and submit the team project report.
- The project report should be submitted only by the **team lead** (on behalf of the whole team), on Canvas.
- The Canvas submission must include
    - A **single PDF** report, uploaded directly to Canvas. Do **NOT** put the PDF file inside your compressed file.
    - A single compressed .zip or .tar file, containing videos, code, etc, as specified by each problem. Do **NOT** add code, data, or anything else that the problem is not asking for. Otherwise, the file size will be large, resulting in upload failure.
- Due to the random nature of RANSAC, the results produced by each run of a SLAM system will be slightly different. Therefore, if two teams submit **identical results**, it is clear that the results are copied. This is an act of **plagiarism** and all team members will receive **zero** for their final project AND bonus points.
- You are **not allowed to use generative AI** for the parts that ask you to describe how the SLAM system works, and must explain the system based on your understanding. Any suspicion of using generative AI will result in a grade of **zero**. This will be based on the instructor's discretion and is not disputable.
- Because SLAM systems (and their implementations) are complex, it is crucial that you carefully read and follow the instructions provided by their authors. Debugging any issues you face during implementation will be difficult as they can be caused by a variety of reasons. While you are encouraged to post your problems/questions on Ed Disccusion and get help from other students, **the instructor/TA is not responsible for answering debugging questions**. To ensure you succeed, please start working on the project early, use online resources (e.g., Google search problems, use Gen AI, look at GitHub issues) to find answers to your questions, and use Docker/DevContainer to isolate any interfering libraries installed on your system. Note that we have implemented all SLAM systems mentioned here, and we are certain that they work if implemented correctly.

Grading: The team leader will grade all members based on their participation and effort from 0 to 1. If a team member does not help (e.g., with attending meetings, developing code, writing the report, etc.), their grade must be **zero**. The maximum grade assigned to each member is **1**. **The team lead must add the following table at the top of the submitted PDF report:**

Member name	Grade
Firstname Lastname	1
Firstname Lastname	0.8
Firstname Lastname	0
Firstname Lastname	1

Final Project Instructions

Kaveh Fathian, Email: fathian@ariarobotics.com

Handout: Thu, 2024-04-03
Due: Tue, 2025-04-29

Once the final project is submitted, the instructor will evaluate the report/videos/code and give it a grade in the range of **0-10**. This grade is affected by several factors, such as how clean, organized, and comprehensive the report is, if videos are captured correctly, and if the code (for the bonus problem) works out of the box. The final project grade for each member is computed by ***multiplying*** the instructor's grade by the team leader's grade.

Problem 1: Monocular SLAM (+5 pts)
Choose **one** system from the following list of state-of-the-art mono SLAM systems and implement it.

- ORB-SLAM3:
  - Code: https://github.com/UZ-SLAMLab/ORB_SLAM3
  - Paper: https://arxiv.org/pdf/2007.11898
- PLP-SLAM:
  - Code: https://github.com/PeterFWS/Structure-PLP-SLAM
  - Paper: https://arxiv.org/pdf/2207.06058
- DROID SLAM (needs Nvidia GPU)::
  - Code: https://github.com/princeton-vl/DROID-SLAM
  - Paper: https://arxiv.org/pdf/2108.10869
- DPV-SLAM (needs Nvidia GPU):
  - Code: https://github.com/princeton-vl/DPVO
  - Paper: https://arxiv.org/pdf/2208.04726 (and https://arxiv.org/pdf/2408.01654)
-

Run the system on the monocular dataset provided in the project folder, called "**Euroc_MH_01_easy**", and report the following.

- **Video**: Screen capture a video of the system running on the dataset from beginning to end. The video must show the input image sequence, the camera trajectory estimated by the SLAM system, and the 3D-constructed map.
- **PDF report**:
  - Specify which SLAM system you implemented, and how you got it working (e.g., in Docker, etc.). Explain any challenges you faced during the implementation and how you overcame them.
  - Read the paper associated with the SLAM system. In a couple of paragraphs, briefly describe how the system works (e.g., what are the components of the SLAM pipeline, and what do each do). Feel free to use the figures in the paper. You are NOT allowed to use generative AI for this part.
  - Report the ATE and RTE of the estimated trajectory by evaluating against the provided ground truth using EVO (https://github.com/MichaelGrupp/evo).

Problem 2: LiDAR SLAM (+5 pts)
This problem is similar to Problem 1 but is for LiDAR. Choose one system from the following list of state-of-the-art LiDAR SLAM systems and implement it. The implementation **must run in ROS**.

- ALOAM:
  - Code: https://github.com/HKUST-Aerial-Robotics/A-LOAM
  - Paper: https://www.ri.cmu.edu/pub_files/2014/7/Ji_LidarMapping_RSS2014_v8.pdf

Final Project Instructions

Kaveh Fathian, Email: fathian@ariarobotics.com

Handout: Thu, 2024-04-03
Due: Tue, 2025-04-29

- KISS-ICP:
  - Code: https://github.com/PRBonn/kiss-icp
  - Paper: https://arxiv.org/pdf/2209.15397
- DLO:
  - Code: https://github.com/vectr-ucla/direct_lidar_odometry
  - Paper: https://arxiv.org/pdf/2110.00605

Run the system on the LiDAR dataset provided in the project folder, called "**sequence_00**". You can convert the ROS1 bag to ROS2 by searching online to find one of the existing tools (e.g., this link). Report the following.

- **Video**: Screen capture a video of the system running on the dataset from beginning to end. The video must contain the trajectory estimated by the SLAM system, and the 3D LiDAR map.
- **PDF report**:
  - Specify which SLAM system you implemented, and how you got it working (in ROS, Docker, etc.). Explain any challenges you faced during the implementation and how you overcame them.
  - Read the paper associated with the SLAM system. In a couple of paragraphs, briefly describe how the system works (e.g., what are the components of the SLAM pipeline, and what do each do). Feel free to use the figures in the paper. You are NOT allowed to use generative AI for this part.
  - Report the ATE and RTE of the estimated trajectory by evaluating against the provided ground truth using EVO (https://github.com/MichaelGrupp/evo).

Bonus problem: Mono-SLAM Competition (+7 bonus pts)

For this problem, you will need to improve and run the basic visual odometry pipeline provided at https://github.com/ariarobotics/robotic-mapping/tree/main/code/cpp/visual-odometry.
The monocular datasets that you should test your system on are provided in the project folder, and are called "**Synthetic_base**" and "**Synthetic_competition**". You are free to make any changes and add any improvements you want. This includes bringing functionalities from other systems, using existing libraries, etc. Please carefully read and follow the instructions below:

- Your code must run in the DevContainer provided in the repo. You will submit your code and we will run and test it in the DevContainer. The code should run out of the box, and we will not attempt to fix it if there is an error during compilation. You must test your code (e.g., on another computer) & make sure that it compiles without an issue in the DevContainer. In the past, some students who have spent a significant amount of time on the project have gotten a grade of zero simply because their code did not compile. Therefore, it is your responsibility to make sure that the submitted code works so that your effort is not wasted.
- You are allowed to modify the "Dockerfile" in the ".devcontainer" folder only after line #120. Do NOT modify any code/commands before line #120. Doing so will result in a grade of zero.
- You should test the performance of your system on the "**Synthetic_base**" dataset provided in the project folder, which contains the ground truth trajectory to help you evaluate your system. The dataset "**Synthetic_competition** will be used to benchmark your system against other teams. The ground truth for this dataset has not been released to you.

Final Project Instructions

Kaveh Fathian, Email: fathian@ariarobotics.com

Handout: Thu, 2024-04-03
Due: Tue, 2025-04-29

- The benchmark will be based on ATE, and we will use EVO (https://github.com/MichaelGrupp/evo) to compute the ATE.
- We will **not** evaluate runtime, so your code does not have to run in real time, and you can improve the performance of your systems without being concerned about runtime.
- The teams are ranked based on the ATE, team members with the top 25% lowest ATE will receive 7 points, the next 25% will receive 5 points, the next 25% will get 3 points, and the last 25% will receive 1 point.