Deep-RL-Based safety landing using RGB camera on rough terrains

Perception and Learning for Robotics
Project Proposal
Supervised by: Michael Pantic, Tim Kazik and Olov Andersson
March 14, 2021

GROUP MEMBERS

Nicola Loi Valentin Ibars


I. DESCRIPTION OF THE PROJECT

The objective of our project is to apply Deep-RL method to the problem of finding a safe landing spot on rough terrains for a drone using a RGB camera. Our work will not start from scratch, but it will be a direct continuation of an already existing project. To achieve our goal we will start by developing a realistic simulation, since the current one is an oversimplification that does not account neither the three-dimensionality of terrain nor its true texture. In this new simulation setting, the next step will be to remodel the current action space, the state space, and the rewards: other alternatives must be investigated to enhance and adapt the old reinforcement learning to the more complex environment. The images from the camera must be processed to extract useful features for terrain identification. Reliable extraction of task-specific features plays an important role in providing the reinforcement learning algorithm with effective data. A substantial portion of the project will be spent on extensive testing of the new features to optimize them. Another important aspect, which is missing from the current version of the project, is to perform a proper evaluation of the functionalities and applications of the algorithm, studying its limits.

II. WORK PACKAGES AND TIMELINE

The work provided during this project will be a python based Deep-RL using OpenAI PPO running on a GPU with Nvidia driver (Ubuntu 18.04). To test our model we will use simulation OpenAI Gym [1] with UnrealCV [3] in order to have a realistic 3D representation of our environment with Unreal Engine [2]. Thus to challenge our model, we will use a 3D representation of a glacier and apply different illumination configurations using the UnrealCV plugin. Our work will be divided into 3 parts: construct a realistic simulation, optimize the actions/states/rewards model and Training/Test/Behavior identification. We currently foresee the following milestones:

- Realistic Simulation and safe spot discretization on 3D model[2/04]
- Remodel the current action space, the state space, and the rewards [9/04]
- Midterm Progress Report [25/04]
- Project video [31/05]
- Training/Test/Behavior identification [4/06]
- Final project presentations [07/06]
- Final project reports
- CoRL-2021 [July]

Possible challenges: Through this project we will have to deal with new platforms and tools but we will tackle these problems with all documentation provided and support we have from ASL lab. In addition we will have to deal with the problem of how to define a safe spot from 3d model, and to identify proper image features to distinguish safe spots. Another challenge related to the feature extraction is the robustness against illumination changes, to make the drone capable of flying at all hours.

III. Outcomes and Demonstration

The expected outcome of the project is to obtain an efficient and effective model capable to land drones autonomously with a low rate of failures. The algorithm should be able to identify the level of safety of the terrain beneath, recognizing from the data received from the camera a safe spot from a dangerous one with a high degree of accuracy. Many test will be run to analyze the performance with different parameters, such as the discretization of the terrain or the size of the state space. The final demonstration would be to show through a graph the performances and the limits of a RGB camera depending on different illumination configurations, such as a change in the intensity or of the angle of incidence. An online or offline video showing the simulation of a drone landing will be presented.

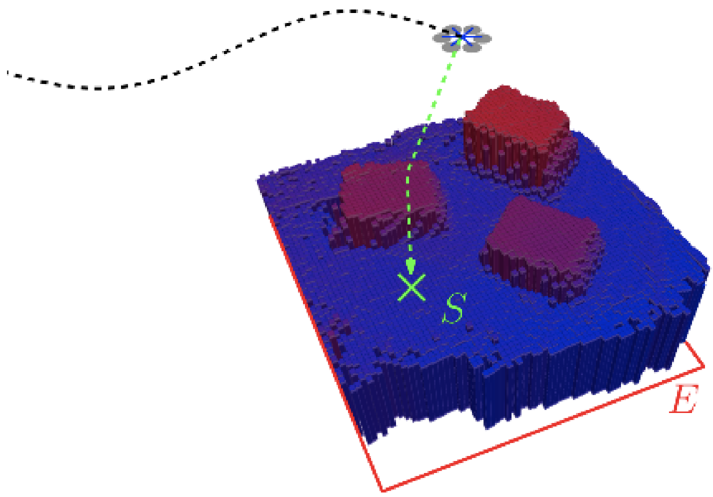

Fig. 1: Example image of a drone landing.

References

[1] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

[2] Epic Games. Unreal Engine simulator. https://github.com/EpicGames/UnrealEngine.git.

[3] Yi Zhang Siyuan Qiao Zihao Xiao Tae Soo Kim Yizhou Wang Weichao Qiu, Fangwei Zhong and Alan Yuille. Unrealcv: Virtual worlds for computer vision. *ACM Multimedia Open Source Software Competition*, 2017.