REAP: A Flexible Real-World Simulation Framework for AI-Planning of UAVs

Oliver Kraus[*1], **Lucas Mair**[*1], **Jane Jean Kiam**[1]

[1]University of the Bundeswehr Munich, Neubiberg, Germany
{oliver.kraus, lucas.mair, jane.kiam}@unibw.de

Abstract

Both AI-planning and aerial robotics have seen a surge in applications in the real-world, which requires removing textbook assumptions on idealized environments. This in turn necessitates a simulation framework that sufficiently supports these applications in terms of realism and deployability. We present REAP (Real-world environment for aerial AI-planning), a framework that integrates state-of-the-art AI-planning tools and open-access real-world data into a modular system. It allows for AI-planning with discretized geo-referenced data, including validation with realistic control algorithms and visualization utilizing 3D map data. The framework is easily extensible since it seamlessly connects to the ROS2 ecosystem and all of its components are open source. A system demonstration can be found at: https://www.youtube.com/watch?v=QtMjnMD5zzE

Motivation

Simulators are important for validating and debugging planning methods. However, existing simulators for AI-planning use highly abstracted environments, which deter their use for real-world applications (De Pellegrin and Petrick 2022; Andreasen et al. 2022). Meanwhile, more realistic frameworks such as (Ma, Zhou, and Li 2020) are mostly based on hand-built models, requiring a lot of effort to reconstruct reality, and do not provide an interface for automated planners. To bridge this gap, we present the simulation framework REAP (Real-world environment for aerial AI-planning) that leverages state-of-the-art libraries and uses real-world data to construct a simulation environment (see Figure 1), while still providing a realistic approximation of flight physics. The advantages of our proposed system are multifold. Firstly, it is highly modular with clearly defined interfaces, making it easy to interchange individual components. Furthermore, we use only open source components so that the system is reproducible and extendable. This also enables easy deployment on real hardware. Lastly, our framework leverages real-world data from Geographic Information Systems (GIS), facilitating the use of AI-planning for real-world applications.

Figure 1: Imported LIDAR environment in UE 4.

REAP System Description

REAP[1] is an end-to-end planning and simulation framework that enables visualization and validation of plans for unmanned aerial vehicles (UAVs) in a realistic 3D environment. As depicted in Figure 2, REAP consists of two subsystems, one for validation and visualization, the other for AI-planning. Communication between both subsystems is handled by the ROS2 (Macenski et al. 2022) action client-server model.

The core of the AI-planning subsystem is PlanSys2 (Martín et al. 2021) which provides a module for translating plans (i.e. sequences of actions) into ROS2 messages. PlanSys2 integrates the UPF4ROS2-plugin[2] for generating plans using the Unified Planning Framework (UPF) (Micheli et al. 2022), through which many state-of-the-art automated planners are made available. Real-world GIS data in shapefile format are parsed and abstracted using the GeoPandas library[3] as geo-referenced polygons in an automated manner. These are subsequently encoded using the Planning Domain Definition Language (PDDL) into planning problems to be solved by planners of the UPF. In REAP, we use geospatial vector data in shapefile format that describe the land use of an area. The geo-referenced polygons are labelled according to land use (e.g. “urban areas”, “woods”, “waters”). Based on these defined labels, a flight mission can be automati-

[*]These authors contributed equally.

[1]https://github.com/UniBwM-IFS-AILab/REAP
[2]https://github.com/PlanSys2/UPF4ROS2
[3]https://geopandas.org/en/stable/index.html

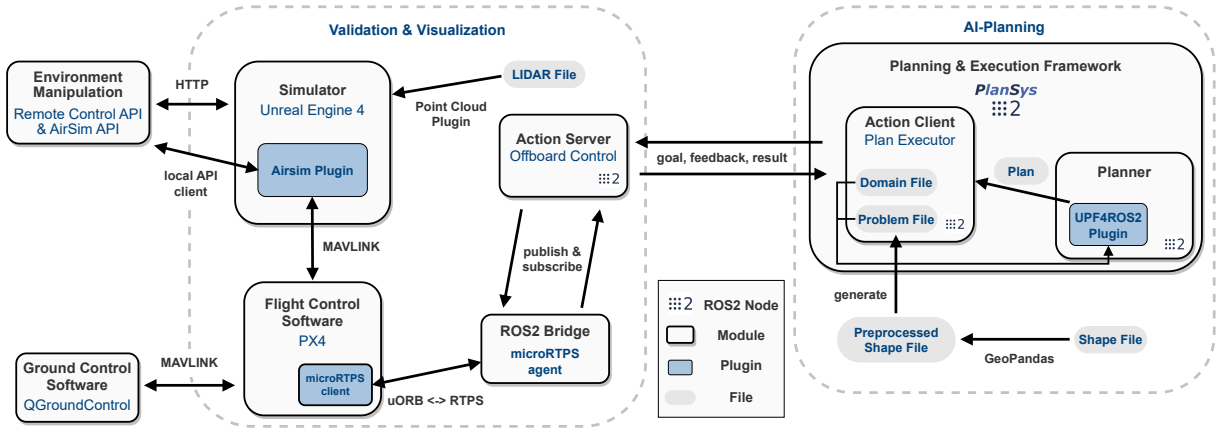

Figure 2: Overview of the REAP system architecture.

cally generated (e.g., "Explore all 'woods' in an area") or the UAV's exploration mode (e.g., altitude) can be dynamically adapted to the area it is exploring.

The core component of the validation and visualization subsystem is Unreal Engine 4 (UE 4)[4] which integrates the Airsim plugin (Shah et al. 2017). AirSim is an open-source simulation platform for autonomous vehicles. The simulated UAV is directly controlled by a flight control software (e.g. PX4 or ArduPilot). This software can also be deployed on an onboard micro-controller, enabling therefore the transfer of software-validated plans to real flights. The flight control software in our framework receives commands via a ROS2 action server, which translates the symbolic arguments received from the ROS2 action client of the AI-planning subsystem into the corresponding continuous control commands. The geodetic_utils library[5] is used to transform coordinates in WGS84 into the local NED reference frame used by the flight control software. To create a realistic simulation, the 3D environment in which the UAV navigates is automatically generated from imported georeferenced LIDAR files. Figure 1 shows a view of such an imported LIDAR file in UE 4. To further increase the resolution of single objects, user-defined 3D meshes (e.g. created with photogrammetry) can be imported as well. In addition, a remote control API for the UE 4 and the AirSim API allow for the dynamic manipulation of the simulation environment. This can be used to test the calculated plan in the presence of unforeseen events. For example, new objects can be spawned in the engine or properties (like wind or weather) can be modified. Finally, manual control of the UAV via a graphical user interface is also possible. This is achieved by connecting software like QGroundControl to the simulation.

The shape and LIDAR files used in our demonstration were made available as open-access data by the State of Bavaria[6]. However, there are many other sources that publish such open-access data.

Future Work

As the validation and visualization subsystem of the framework already supports the simulation of multiple concurrent UAVs, we plan to include this in the REAP-framework for multi-agent planning. Furthermore, we plan to support the dynamic replanning of broken plans in our framework.

Acknowledgments

This work was supported by WIWeB under the grant number E/E210/AM016/KF085.

References

Andreasen, M.; Holler, P.; Jensen, M.; and Albano, M. 2022. MAES, a Realistic Simulator for Multi Agent Exploration and Coverage. In *ICAPS 2022 System Demonstrations*.

De Pellegrin, E.; and Petrick, R. 2022. Plan Visualisation with PDSim. In *ICAPS 2022 System Demonstrations*.

Ma, C.; Zhou, Y.; and Li, Z. 2020. A new simulation environment based on AirSim, ROS, and PX4 for quadcopter aircrafts. In *2020 6th International Conference on Control, Automation and Robotics (ICCAR)*. IEEE.

Macenski, S.; Foote, T.; Gerkey, B.; Lalancette, C.; and Woodall, W. 2022. Robot Operating System 2: Design, architecture, and uses in the wild. *Science Robotics*, 7(66).

Martín, F.; Ginés, J.; Rodríguez, F. J.; and Matellán, V. 2021. PlanSys2: A Planning System Framework for ROS2. In *IEEE/RSJ IROS*.

Micheli, A.; et al. 2022. Unified Planning: A Python Library Making Planning Technology Accessible. In *ICAPS 2022 System Demonstrations*.

Shah, S.; Dey, D.; Lovett, C.; and Kapoor, A. 2017. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. In *Field and Service Robotics*.

[4]https://www.unrealengine.com

[5]https://github.com/ethz-asl/geodetic utils

[6]https://geodaten.bayern.de/opengeodata/index.html