Quadrotor Waypoint Tracking with   
Deep Reinforcement Learning

DS551 - Final Project - 2023

Manoj Velmurugan, Swati Shirke, Ramana Kiran, Rigved Sanku

# Problem Statement:

A blue line with stars

Description automatically generated

* Objective of this project is to design an RL agent for following waypoints in a simulated quadrotor.
* Here, the current state of the quadrotor will be [position, velocity, orientation, angular rate].
* The agent will learn the task of motion planning and control by interacting with the environment. The agent will give Angular velocities of the 4 rotors and cumulative thrust of the quadrotor as output. These values will be imported in another simulation software to verify if a quadrotor is following waypoints with accuracy.

# Related Work

A circular light in a room

Description automatically generated with medium confidence

A team from “University of Zurich” recently demonstrated expert level performance in drone racing with Deep Reinforcement Learning. The drone must pass through a series of windows autonomously solely with onboard sensors and camera.

For more information, please refer to,

Kaufmann, E., Bauersfeld, L., Loquercio, A. et al. Champion-level drone racing using deep reinforcement learning. Nature 620, 982–987 (2023). <https://doi.org/10.1038/s41586-023-06419-4>

In this work, we are going to attempt a simplified version of this problem in simulation. Our scope is limited and more focused on the RL instead of the sensing and hardware implementation aspects!

|  |  |
| --- | --- |
| Their work | Our scope |
| Deep RL using A3C | Deep RL using A3C (or any other method suited for continuous systems) |
| State estimation from sensors and camera | NA Quadrotor’s position and other states are known and fed directly from the environment into agent |
| Window detection from Camera | NA Window locations are known in advance and computer vision is abstracted out |
| Sim2Real and hardware implementation | NA Scope is only simulation |

# Environment and visualization

# 

## Inputs:

1D double vector (23x1) containing,

1. Previous agent outputs - desired angular rates - [wx, wy, wz] – 4x1
2. Position waypoints for next 3 waypoints [x1, y1, x2, y2, x3, y3] – 6x1
3. States of quadrotor – [position, velocity, orientation, angular rate] 13x1

## Output:

1. 1D double vector containing desired angular rates and desired thrust – 4x1
2. Reward. A reward function will be implemented as provided in the referenced paper.

## Termination:

Simulation will be terminated if the drone,

1. Goes below ground level.
2. Reaches last waypoint/goal.

## Environment and Visualization:

We have a Python based quadrotor simulation model built by one of our team members for the “Hands-On Autonomous Aerial Robotics Course” (RBE595).

The simulator has worked robustly for the course. So, we will use the same simulator for this work.

A wooden box with a drone

Description automatically generated

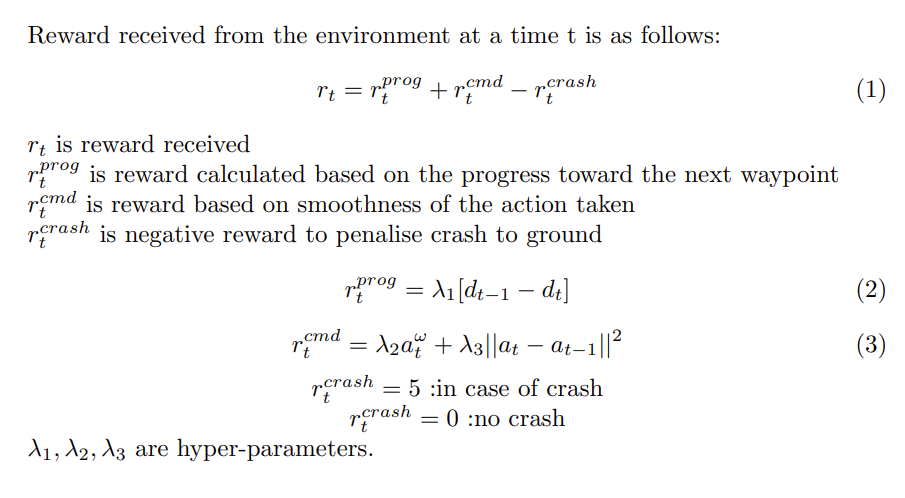
Visualization portions will be stripped out to speed up training.

Trained agents will be visualized on our Blender based visualization utility.

# Agent:

## Agent Class - A3C

# Reward System:



## Network architecture

Idea is to start off with a simple MLP based architecture to fit the which takes in the trajectory inputsand gives out the outputs as mentioned above. In essence this MLP is approximating a controller like Model Predictive Controller (MPC). Based on this performance, we also plan to try out a variant of temporal convolutional networks designed for time series forecasting. The idea of this network is to feed in a set time series inputs to determine the outputs for the current step.

# References

Kaufmann, E., Bauersfeld, L., Loquercio, A. *et al.* Champion-level drone racing using deep reinforcement learning. *Nature* **620**, 982–987 (2023). https://doi.org/10.1038/s41586-023-06419-4