CapstoneProposalReinforcedEngineer

February 17, 2017

1 Machine Learning Engineer Nanodegree

1.1 Capstone Proposal

Martin Simon
February 6th, 2017

1.2 Proposal

(approx. 2-3 pages)

1.2.1 Domain Background

(approx. 1-2 paragraphs)

Finding optimal solutions in Mechanical Engineering requires thorough theoretical understanding of physics and practical experience in the domain to find good solutions. However, even skilled professionals experiment with a number of theories before finding the most suitable solution. Machine Learning algorithms can be used to find hidden patterns and automate the engineering process. Genetic Algorithms have already proven useful - proposing technical solutions, which humans would probably not find with human intuition. A great example of this is the design of a NASA satellite antenna: https://ti.arc.nasa.gov/m/pub-archive/1244h/1244%20(Hornby).pdf. An examples of applying Machine Learning is Simulation Data Mining.

I am currently designing a Blended Wing Body Unmanned Aerial Vehicle and am interested in experimenting with Machine Learning algorithms to help make the solvers more "intelligent" to speed up the development process. Handing tasks that can be automated and are based on a lot of "trial and error" is a significant improvement of an engineers workflow. However, physics simulation driven policies can act in other applications as well, such as real-time decision making. The policy will be using open-source softwares to generate geometry and get evaluations for the proposed geometries: XFLR5, XFoil, OpenVSP and AeroPython.

1.2.2 Problem Statement

(approx. 1 paragraph)

Train a Reinforced AI Engineer-Policy to find aerodynamically optimal geometric parameters for a Blended Wing Body Unmanned Aerial Vehicle design. The optimal solution is determined by evaluating the UAV-s Coefficient of Lift, Drag and stability factors.

1.2.3 Datasets and Inputs

(approx. 2-3 paragraphs)

The preliminary geometric model is built by myself, making some broad assumptions about where the useful model should be. Some geometric limits will be set to make sure it is manufacturable and can carry electronics. The initial mutations of the model will be random. The preliminary input model will be based on airfoil sections, thus giving the policy the option to remember what worked on one section and use that knowledge to make decisions on the next.

For modelling and analysis open-source airfoil/wing platforms will be used, such as XFoil, XFLR5 and OpenVSP.

1.2.4 Solution Statement

(approx. 1 paragraph)

Based on decisions, the policy gets feedback. Small changes will be penalized in the beginning to get moving. Also, annealing should be used to avoid local minimas.

1.2.5 Benchmark Model

(approximately 1-2 paragraphs)

The benchmark model is the human engineer who tinkers the initial geometric model of the UAV to have the best attributes of the stability, drag, lift and weight.

1.2.6 Evaluation Metrics

(approx. 1-2 paragraphs)

Evaluation metrics are stability, drag, lift and weight values.

1.2.7 Project Design

(approx. 1 page)

Finding an aerodynamically optimized body for the UAV begins with constructing an initial model and sets of constraints by the human engineer. The policy is given a set of parameters it can modify and is expected to learn which combinations will give the best effect when constrained by payload size, and weight, UAV range, endurance (time), stall margin (m/s), minimum airspeed (m/s) and available power etc. The policy will take leverage of 2D and 3D aerodynamic modeling to find optimal solutions with the least amount of GPUh/CPUh. The project workflow: 1. Set up aerodynamic analysis programs and make a decision which of them are most beneficial for the project. 2. Build a simple Python-API to mediate with the selected programs. 3. Create a set of constraints and initial model. 4. Test a DNN based Reinforcemed ML Policy with a simple game from OpenAI Gym. 5. Connect the modifiable parameters and reward function to the Reinforced learner. 6. Fine-tune the set-up of the ML system. 7. Train final policy and evaluate results.

```
In [ ]:
```