Black Box Optimization of PID controllers for Micro Aerial Vehicles

Oswald Berthold and Verena V. Hafner

I. Abstract

***Abstract*— When operating flying robots we repeatedly face the necessity of tuning controllers for particular electronic and physical configurations. In the spirit of embodied robotic learning we want to tune the controllers during the system's closed-loop operation. We present an experiment applying a black box Bayesian optimization technique to the problem of tuning a hierarchical vertical position PID controller of an off-the-shelf Micro Aerial Vehicle flight controller and interpret it in the context of Policy Search.**

II. Introduction

For research in Micro Aerial Vehicles (MAV) we use off-the-shelf components for constructing the base platform of the flying machine. At the core of such a machine is the so called Auto-Pilot (AP) which usually consists of a circuit board carrying the sensors and a microcontroller and a piece of software that runs on the microcontroller which implements the stabilization functions of the AP. These functions, such as stabilization of angular rate and angle (for the pitch, roll and yaw axes) or vertical and lateral velocity and position, are usually realized by vendors with a set of PID controllers.

Repeatedly we find ourselves in the need of tuning the parameters of these controllers to match the electric and physical characteristics of the systems, most importantly motor / motor controller / propeller combinations as well as mass and geometry of the airframe. Usually a working set of parameters can be found by using heuristics (recipes) or intuition, but more often than not, the performance regarding the basic motion capabilities of the robotic platform are not entirely satisfying.

Control theory provides methods for designing and parameterizing controllers, yet we want to circumvent the effort needed for implementing and testing custom controller code as well as the need for building a system model required in the canonical control theoretic approach.

Here we present a systematic approach to tuning parameters for arbitrary controller architectures, which does not require a system model. We apply a black box optimization method designed for optimizing hyperparameters of machine learning architectures directly to the base level parameters of PID controllers. This, in fact, implements Policy Search in the space spanned by the underlying controller family. We describe a preliminary yet exemplary experiment tuning a hierarchical controller stack with six parameters on a small quadrotor platform.

III. Related work

The fields of Control Theory and Reinforcement Learning (RL) are directly relevant to the problem. In Control Theory one often considers systems where exploratory experiments are expensive. In this case, the effort of building the mathematical model is clearly justified as still being less expensive than an actual experiment. In our case experiments are cheap and we wish to exploit this fact, which goes well together with the concept of embodied learning in Robotics.

In RL and Sensorimotor Learning on the other hand the intention is the same with the difference, that the underlying representations which implement the controller family, in these contexts called policies or inverse models respectively, can be chosen freely. This choice can have an effect on the type of learning algorithms that are applicable.

In our case, we are not free to specify the underlying structures but instead we want to use those provided by a given AP vendor, usually sets of hierarchical PID controllers. From this arises the need for using a model agnostic optimization method. This is provided, among others, by hyperparameter optimizers originating from the model search problem in connection with machine learning for vision pipelines (Bergstra, Yamins, and Cox, 2013) which is available as a Python implementation (hyperopt).

IV. Robot and methods

For a photogrammetry application, we are using a small and low-cost quadrotor that carries an equally low-cost camera (808 car key camera). The brushless motors, electronic speed controllers and the frame are standard R/C components. The AP used here is the Naze32 which is a quadratic board with an edge length of 3.5 cm and an STM32 microcontroller. Exchanging data, such as telemetry measurements and configuration commands, with the stock firmware is done via a bluetooth serial connection using a custom MAVLink to Multiwii Serial Protocol converter [1].

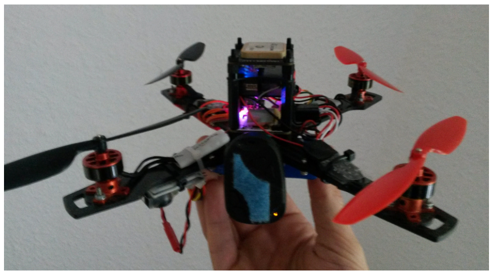

Fig. 1. Quadrotor MAV with camera.

V. Experiment

For the application, we require altitude stabilization, which is implemented using two PID controllers. The inner loop regulates the vertical velocity while the outer loop regulates the position by applying setpoints to the inner controller. In total we have a six-dimensional parameter space where the search is to be conducted.

The following procedure describes a single evaluation of one set of parameters:

1) Draw parameters and configure the AP.
2) Take-off by human pilot.
3) Activate altitude stabilization at a safe distance from ground (~4-6m) using the altitude at the moment of activation as setpoint.
4) Evaluate controller for a maximum of 30s.
5) Landing by human pilot.
6) Compute the Mean Squared Error (MSE) between the altitude setpoint and the actual altitude for the interval the stabilization was active.
7) If the evaluation was terminated earlier due to instabilities, set the MSE to a large constant.
8) Return MSE as cost for hyperopt's function minimizer

An example of some episodes showing these phases can be inspected in Figure 2. The optimizer's estimator of the parameter dependent performance is initialized with 20 random samples and then starts yielding suggestions based on the observed sample history Figure 3. The prior distribution for each of the parameters is specified as input to the optimizer, in this case integer-quantized uniform distributions. The procedure is repeated for a maximum number of 100 evaluations, since there is no intrinsic termination criterion in the optimizer. We find two configurations with sufficient performance with respect to the constraints of the intended application.

[1]https://github.com/koro/python-multiwii

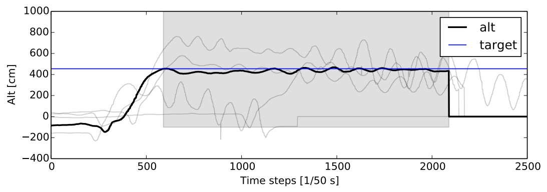

Fig. 2. Evaluation episode examples. The thick black line corresponds to the best performing parameter set (Ep. #87) and the shaded rectangle indicates the evaluation interval. The gray lines correspond to altitude trajectories for other trials to provide a point of comparison in addition to the MSE values.

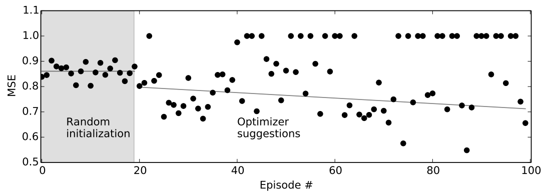

Fig. 3. MSE for all hundred episodes plotted in chronological order. The minimum for this run is at episode #87. The MSE values have been normalized to the interval [0, 1]. All points that exactly equal 1 refer to episodes that were prematurely terminated due to instabilities. The increasing occurence of these failures are due to exploration and large variance.

VI. Conclusion

We presented a method for online PID optimization for a hierarchical MAV altitude controller with 6 parameter dimensions using model agnostic black box optimization. The procedure finds two good solutions within 100 iterations (~1.5 hours experimental time). These are preliminary results which need to be substantiated further by reducing the performance variance, establishing a full random search baseline, comparing different estimators and optimizing the attitude control loop.

References

Bergstra, James, Daniel Yamins, and David Cox (2013). "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures". In: *Proceedings of The 30th International Conference on Machine Learning*, pp. 115–123.