Research report
Author: eng. Bazan Michał

Table of contents

Change sheet

Date	Description
04/14/2024	Basic informations and document structure, glossary, description of PID experiment.
04/15/2024	Experiment description update.

Glossary

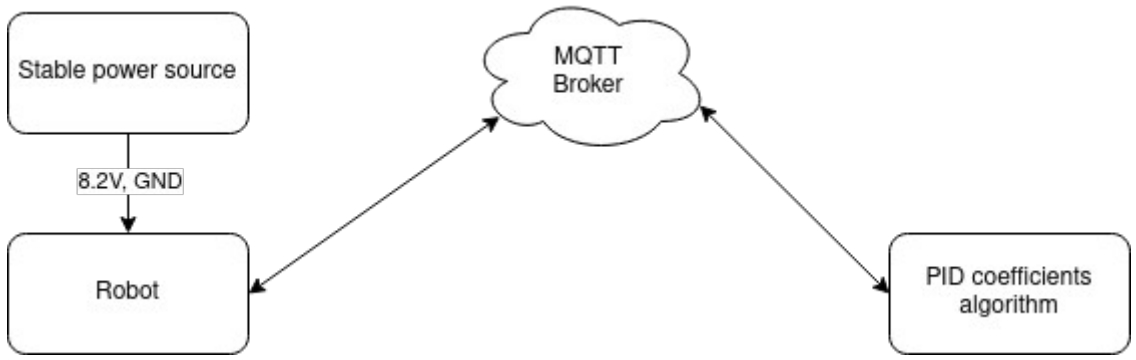
PID - PID is an acronym for Proportional-Integral-Derivative. It's a popular control algorithm used in automation to regulate various dynamic systems. The PID controller operates based on three main components: proportional (P), which responds to the current error, integral (I), which compensates for systematic error, and derivative (D), which predicts changes in error over time. The coefficients of these components are tuned to achieve the desired system response.

Genetic Algorithm - A Genetic Algorithm is a heuristic technique used to solve optimization problems and search through large solution spaces. Inspired by the principles of natural selection and genetics, it iteratively evolves a population of potential solutions, applying crossover, mutation, and selection operations to find the best solution or approximate it.

MCU – Master Control Unit.

Speed control experiment

Given a physical robot, it is possible to create a learning loop for any algorithm which will introduce new PID coefficients and test them on real hardware. General PID algorithm and communications are handled by software flashed into robot's MCU. Block diagram of the test setup has been provided below.



Finding PID coefficients in use of Genetic Algorithm

The dc motor from the point of view of automatic control is a simple object, so many sets of parameters will be suitable for automatic speed control. Therefore, it was decided to use a genetic algorithm to select the optimal set of pid coefficients.

Experiment description

Python script containing implementation of genetic algorithm [1] will be run on the local server. Every individual in the population will be tested on the physical robot to check it's fitness.

Algorithm will be run 3 times with different set of parameters:

a) first run

- **population size** → 200
- **max epoch** → 25
- **crossover probability** – 50%
- **parents of new node** – 2
- **mutation probability** – 15%

b) second run

- **population size** → 200
- **max epoch** → 25
- **crossover probability** – 40%
- **parents of new node** - 5
- **mutation probability** – 15%

c) third run

- **population size** → 200
- **max epoch** → 25
- **crossover probability** – 20%
- **parents of new node** - 10
- **mutation probability** – 40%

Evaluation conditions (**common for both cases**):

- **settling time** → min,

- **max error of regulation** → - 2 <= error <= 2 [%]

Fitness function – since algorithm maximizes this value, below formula will be used to achieve optimal PID coefficients:

$$\frac{1}{settling\,time + relative\,error}$$

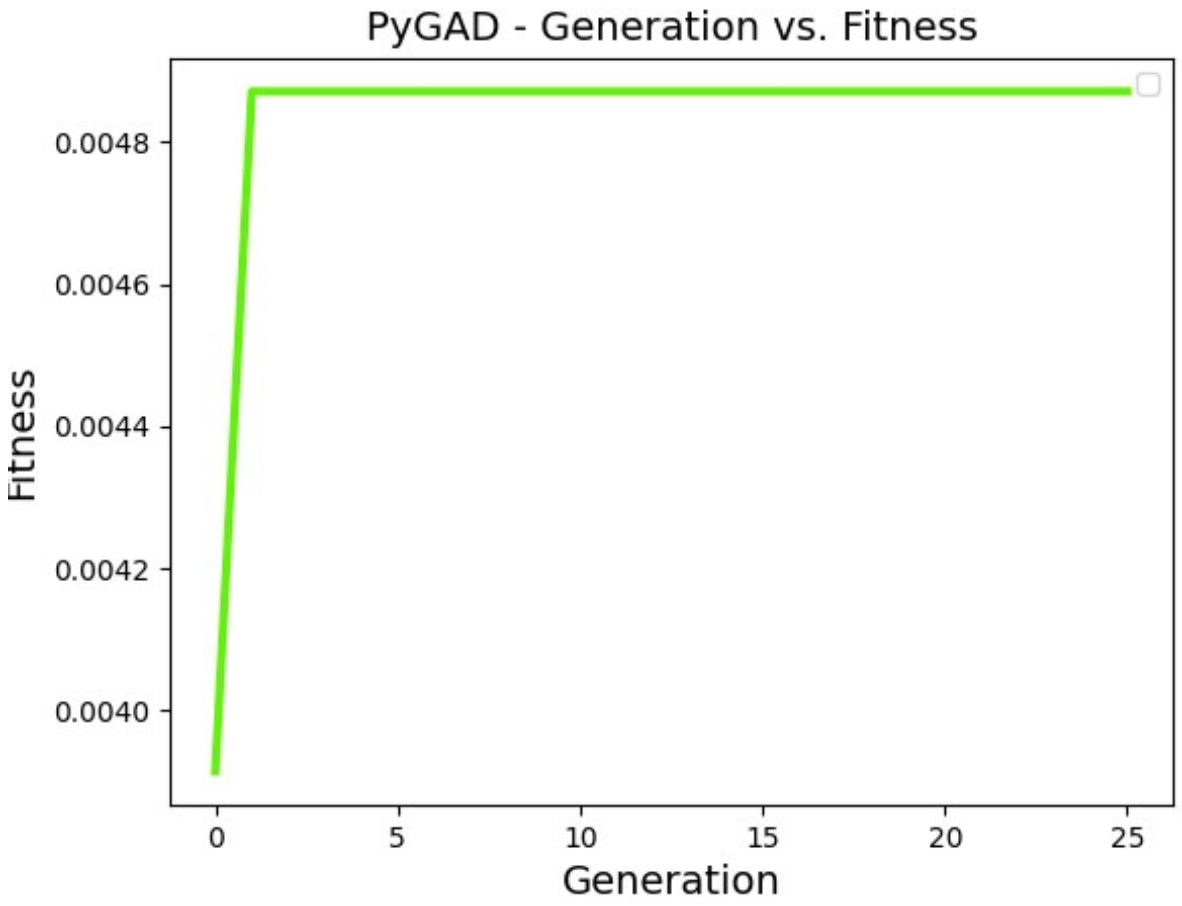
Stop condition – max epoch reached

Estimated experiment duration: 19h.

Experiment results and conclusions

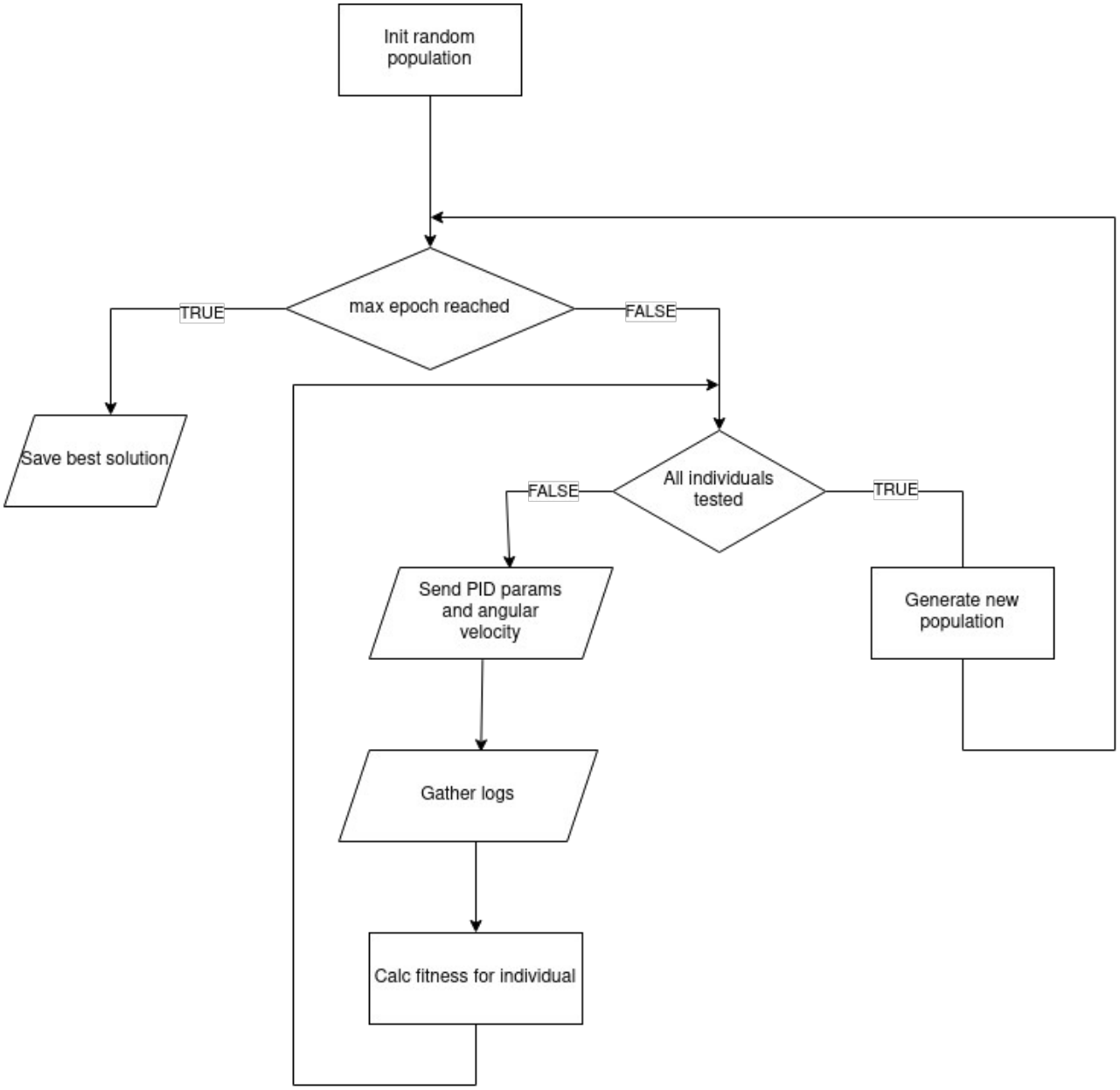
a) first run

The results of the algorithm run indicate stagnation, so the next experiments assume a higher probability of gene mutation and a higher number of parents of the new individual.



Run_ID	Parents mating	Mutation probability	Fitness	Parameters (Kp, Ti, Td)
1	2	15%	0.00487	14.64800285, 11.39891259, 34.22699805
2	5	40%		
3	10	40%		

Algorithm block diagram



References

[1]. https://pygad.readthedocs.io/en/latest/

[2]. https://github.com/DevxMike/master_degree/tree/master/research