RL & GA optimizaztion obstacle avoidance recap.

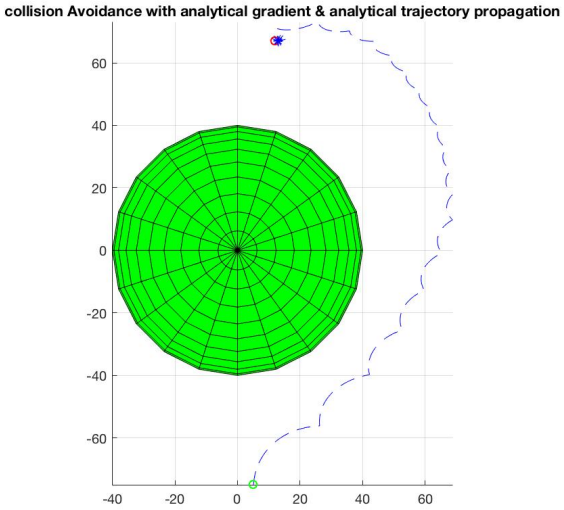
Matteo Baiguera

Abstract

Aim of this document is to summarize results concerning reinforment learning in the simple obstcle avoidance problematics. Main ingredients:

1. **Obstacle:** a circle of radius $50m$
2. **Robo-sat:** autonomus 2d free-flyer, force to move inside a control-box of dimension L starting from r_0 to a goal position r_{goal} avoding the obstacle
3. **neural-network:**
   (a) normalized inputs: current velocity: v/V, current postion: r/L. Where V has been taken reasonably high to ensure $[-1, 1]$
   !! Possible source of bad performance in training !!
   (b) constituted by a single hidden layer. Activation function: **tanh** (all possible values)

1 Baseline:



2 Fitness, Neural-Network, Dynamic: Individual.m (class)

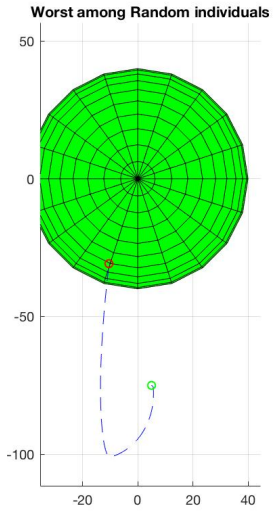
Attempts:

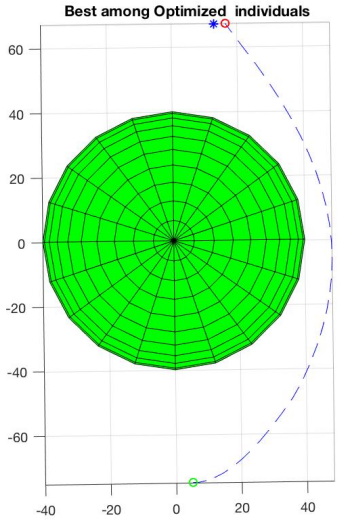
- 1: 8HL,tanh, regressor , $fitness = \{goal + 0.25, collision = -norm(v)/V, out\_of\_box = -norm(r - r\_goal)/L - 0.25\}$
- 2: 16HL,tanh, regressor , $fitness = \{goal + 0.25, collision = -norm(v)/V, default = -norm(r - r\_goal)/L\}$

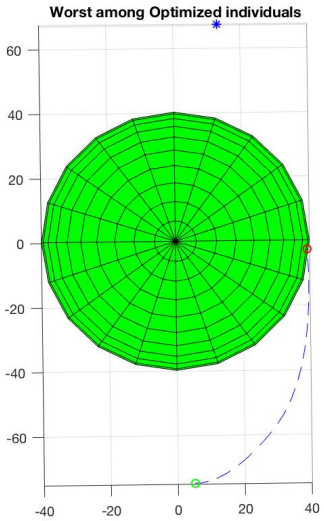
$$\Delta V = W_o(tanh(Whl * \{r/L, v/V\ \} + b_{hl}) + b_o \tag{2.1}$$

Action ΔV ***! evaluated every 10s!***
Simulation stops if $norm(r) > L$ ***or*** $norm(r - r\_goal) < 5m$
Thershold: $|\Delta V| <= 0.0529$

3 result1: [8HL-Neuron][DISCONTINUOUS FITNESS SCORE FUNCTION]: best_individual_until_now.m







This trajectory appears to simulate a continuous thrust trajectory.

4 Optimization considerations: createNextGeneration.m

pseudo GA Optimization:

BREEDER: *best_sample* Individuals among (sorted-by-score) evaluated population: **CHILD-GENERATION:** Half neuron from (random-Breeder) DAD Hal neuron form (random-Breeder) MOM among breeder
MUTATION: every weigth modified by the amount $rand([-0.001, +0.001])$ with a probability: *chance_of_mutation*
GOLDEN-RULE:

$$(best\_sample + lucky\_few)/2 * number\_of\_child = size\_population \tag{4.2}$$

Algorithm:

$$populationSorted = computePerfPopulation(individuals);$$

$$nextBreeders = selectFromPopulation(populationSorted, individuals, best\_samples, lucky\_few);$$

$$nextPopulation = createChildren(nextBreeders, n\_children);$$

$$nextGeneration = mutatePopulation(nextPopulation, chance\_of\_mutation);$$

Program:
generation = 10;

$$while(generation > 1)$$

$$populationNext = createNextGeneration(population, best\_samples, lucky\_few, number\_of\_child, chance\_of\_mutation);$$

$$population = populationNext;$$

$$generation = generation - 1;$$

$$end$$

Population size is kept the same along optimization =>?? Could be acceptable an elimination after a certain generation ??

5 Next steps:

- implement output neurons as 'classifiers' *sigmoid* or Biased output condition like *FIRE/NO-FIRE*
- Train populations on modifying environments
- Train from different initial conditions
- train on a complex obstacle

References

[1] *Reinforcement Learning for Spacecraft Maneuvering Near Small Bodies* Stefan Willis , Dario Izzo , Daniel Hennes AAS/AIAA Space Flight Mechanics Meeting, Napa, CA in February 14-18, 2016

[2] *Self-supervised learning as an enabling technology for future space exploration robots: ISS experiments* Kevin van Hecke*a, Guido C.H.E. de Croon*a, Daniel HennesbTimothy P. Setterfieldc, Alvar Saenz-Oteroc, and Dario Izzo. 67-th International Astronautical Congress (IAC), Guadalajara, Mexico, 26-30 September 2016.

[3] *Trajectory Optimization for Autonomous Flying Base Station via Reinforcement Learning* Harald Bayerlein, Paul de Kerret, and David Gesbert Communication Systems Department, EURECOM Sophia Antipolis, France

[4] *Reinforcement Learning-based Motion Planning of a Triangular Floating Platform under Environmental Disturbances* Konstantinos Tziortziotis, Kostas Vlachos, Member, 24th Mediterranean Conference on Control and Automation (MED) June 21-24, 2016, Athens, Greece