yasd

(yet another) self-driving car simulator

C. Caramaschi M. Biondi V. Armandi

Alma Mater Studiorum · Università di Bologna
Corso di Laurea Magistrale in Informatica
Fisica dei Sistemi Complessi

December 2020

yasd

(yet another) self-driving car simulator

What

An isolated simulation environment designed to study how autonomous cars could learn to properly drive and coexist without an initial well defined traffic law

With

- a convoluted looped road track
- multiple autonomous cars

Without

- road signs

How

the system will work

The autonomous cars should learn to:

Respect

- safety distance
- speed limits
- precedence

Avoid

- border collisions
- road accidents
- traffic congestion

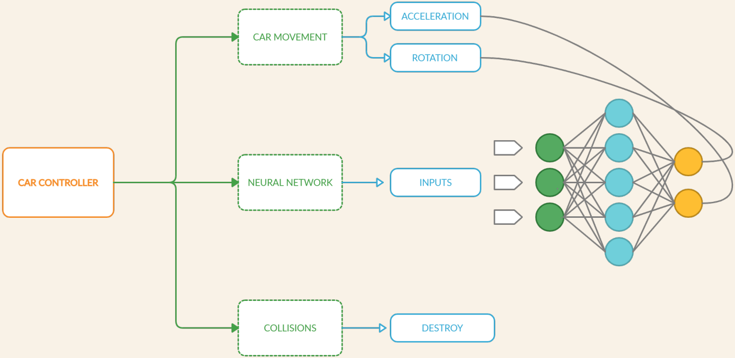
Why

are we developing it?

To study how self-driving cars could learn to coexist autonomously following traffic rules not previously defined

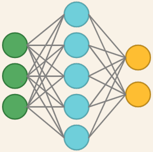
- To which side should priority be given to at a crossroad?
- On which side should an overtaking manoeuvre carried out?

Neural Network



Genetic Algorithm

ASSIGN WEIGHTS



DNA

CAR DNA

0.4 0.8 0.2 0.5 0.7 0.3 0.1 0.9 0.5 0.2 0.4 0.7 0.7 0.8 0.1 0.6 0.5 0.1 0.8 0.9 0.2 0.4 0.3 05 0.1 0.3 0.1 0.7 0.6 0.1 0.7 0.4

New epoch cars DNA

CHOICE OF THE BEST DNA'S

BEST SURVIVOR DNA

0.8 0.1 0.3 0.9 0.6 0.2 0.1 0.8 0.4 0.1 0.5 0.2 0.4 0.1 0.7 0.8 0.1 0.2 0.6 0.4 0.2 0.9 0.8 0. 3 0.6 0.4 0.5 0.9 0.7 0.1 0.6 0.2

SECOND BEST SURVIVOR DNA

0.3 0.2 0.6 0.1 0.2 0.3 0.3 0.7 0.4 0.7 0.5 0.1 0.2 0.3 0.8 0.7 0.9 0.2 0.8 0.7 0.1 0.3 0.5 0.6 0.4 0.5 0.9 0.4 0.5 0.4 0.5 0.9

NEW EPOCH CARS DNA MANIPULATION

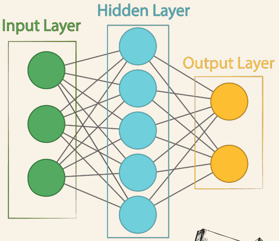
CROSSOVER

0.8 0.1 0.6 0.1 0.2 0.2 0.3 0.8 0.4 0.7 0.5 0.1 0.2 0.1 0.7 0.8 0.1 0.2 0.8 0.4 0.1 0.3 0.5 0. 6 0.4 0.5 0.9 0.4 0.7 0.1 0.6 0.9

MUTATION

0.8 0.2 0.6 0.1 0.6 0.9 0.3 0.8 0.4 0.7 0.5 0.1 0.7 0.1 0.7 0.8 0.1 0.2 0.3 0.4 0.1 0.3 0.5 0. 6 0.4 0.5 0.9 0.4 0.1 0.1 0.6 0.9

Neural Network & Genetic Algorithm



Car DNA

0.4 0.8 0.2 0.5 0.7 0.3 0.1 0.9 0.5 0.2 0.4 0.7 0.7 0.8 0.1 0.6 0.5 0.1 0.8 0.9 0.2 0.4 0.3 0.5 0.1 0.3 0.1 0.7 0.6 0.1 0.7 0.4

Adding BIAS

Activation Function

$$1 \quad \begin{bmatrix} 0.4 & 0.3 & 0.4 \\ 0.8 & 0.1 & 0.7 \\ 0.2 & 0.9 & 0.7 \\ 0.5 & 0.5 & 0.8 \\ 0.7 & 0.2 & 0.1 \end{bmatrix} \cdot \begin{bmatrix} 0.7 \\ 0.4 \\ 0.2 \end{bmatrix} = \sigma\left(\begin{bmatrix} 0.48 + b0 \\ 0.74 + b1 \\ 0.64 + b2 \\ 0.71 + b3 \\ 0.59 + b4 \end{bmatrix} \right) = \begin{bmatrix} 0.7 \\ 0.2 \\ 0.3 \\ 0.4 \\ 0.2 \end{bmatrix}$$

Acceleration

$$2 \quad \begin{bmatrix} 0.6 & 0.1 & 0.9 & 0.4 & 0.5 \\ 0.5 & 0.8 & 0.2 & 0.3 & 0.1 \end{bmatrix} \cdot \begin{bmatrix} 0.7 \\ 0.2 \\ 0.3 \\ 0.4 \\ 0.2 \end{bmatrix} = \sigma\left(\begin{bmatrix} 0.97 + b5 \\ 0.71 + b6 \end{bmatrix} \right) = \begin{bmatrix} 0.21 \\ 0.34 \end{bmatrix}$$

Rotation

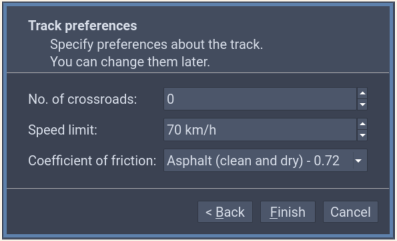
Activation Function = Sigmoid

$$f(x) = \frac{1}{1 + e^{-(x)}}$$

Track

generation process

1. input track parameters
   - number of crossroads
   - speed limit
   - coefficient of friction
2. Lissajous curve algorithm
3. track drawing using graphic libraries



Track preferences

Specify preferences about the track.
You can change them later.

No. of crossroads: 0

Speed limit: 70 km/h

Coefficient of friction: Asphalt (clean and dry) - 0.72

Lissajous

curve algorithm

Use

- simple curves
- good positioning of crossroads

Formulas

- $$x = A_x \sin(\omega_x t + \phi_x)$$
- $$y = A_y \sin(\omega_y t + \phi_y)$$

Car

types

Red car

- driving speed $\approx$ speed limit
- low proximity sensors sensitivity

Green car

- driving speed $\leq$ speed limit
- normal proximity sensors sensitivity

Blue car

- driving speed $\ll$ speed limit
- high proximity sensors sensitivity

Editing preferences

from the second epoch onwards

Yes - Track preferences

- number of crossroads
- speed limit
- coefficient of friction

No - Set of cars

- number of red cars
- number of green cars
- number of blue cars

.yasd

file extension

```
{
    "cars": [{
        "id": 0,
        "color": "red",
        "dna": [0.1,0.6, … ,0.7]
    }],
    "track": {
        "crossroads": 7,
        "friction": 0,
        "limit": 70
    }
}
```

Technical details

about the project

Programming language

- C++

GUI

- Qt5 + OpenGL

Building process

- CMake

Source Code

- GitHub
- GPL-3.0

Future developments

- Reinforcement learning (RL)
- Pictures as sensors input