My little document

August 19, 2016

Contents

1 root

1.1 Reinforcement Learning

1.1.1 Q Learning

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)$$

1.2 The video

`https://www.youtube.com/watch?v=zOgSC---rgM`

The video illustrates a car learning to avoid obstacles. As shown in Figure 1, the environment is a 2D scenario. The whole scenario is surrounded by fences. There are 4 irregularly shaped obstacles. To show the learned ability generalizes well with different

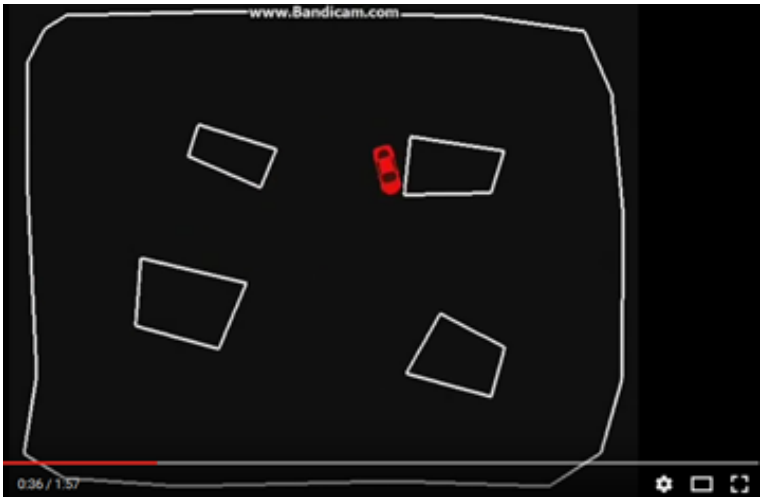

Figure 1: youtube$_{\text{screenshot}}$

layouts of obstacles, the obstacles will revolve about the center of the room slowly, at a constant speed, during running.

For each period, the car starts from the center of the room, with a randomly chosen direction. Its action is controlled by a reinforcement learning algorithm. During the early periods, the actions are like randomly decided. When the car runs into obstacles or fences, it will be repositioned to the center of the room, and a new period begins. The car learns over time.

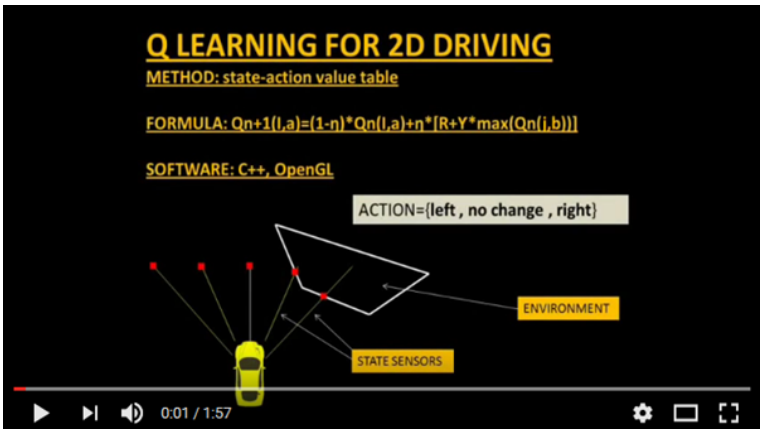

Figure 2: youtube$_{\text{structure}}$

The illustration comprises two parts, a 2D environment described above, and a learning algorithm which controls the action of the car. The algorithm learns from and makes decisions on specified data provided by the environment. The environment emulates 5 state sensors, corresponding to 5 different direction in front of the car. These sensors

find the nearest obstacle, and measure the distance from the obstacle to the car. The distance information is transferred to the car in the form of a 5D vector. During running, the algorithm will predict an action based on the distance information and then feed it back to the environment, to control the movement of the car.

1.3 Re-Implementation

For our further research, we need a verified code base to start with. So we want to implement the application in this video.

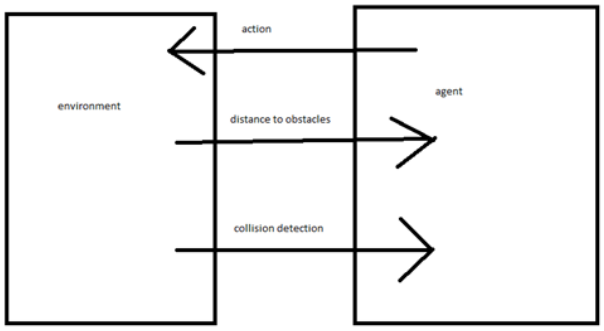

Figure 3: structure

It includes 3 steps.

- Build an environment.
- Implement a reinforcement learning algorithm to control the car.
- Provide and transfer learning data and actions between the environment and the algorithm.

1.3.1 The Environment

We choose to build this 2D environment on a 3D rendering engine. It is for the convenience of later transition. Because our later goal is to train the car to avoid obstacles in 3D environments.

Panda3D

Layout A cube room. A (round) pole at the left bottom corner. Another (square) pole at the top right Randomly positioned chess pieces (200-400) (maybe overlapped)

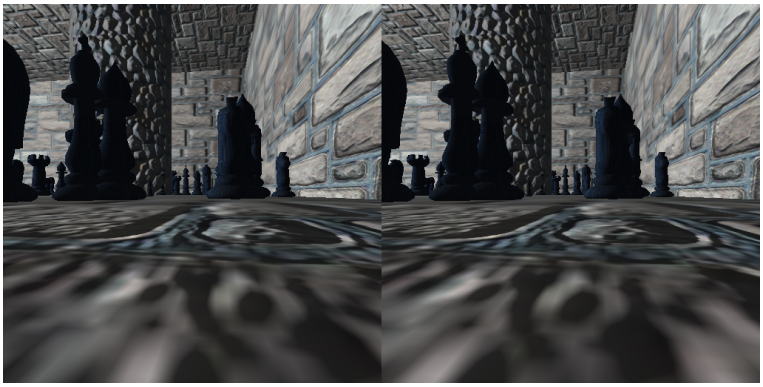

Figure 4: A 3D view

1.3.2 Reinforcement Learning

We started this part with a project on Github.com

Paper `http://arxiv.org/abs/1312.5602`

Code `https://github.com/spragunr/deep_q_rl`

1.3.3 Learning Data

a Extracting depth maps (Goal: E (distance) =¿ Q) Convenient Method: depth map =¿ distance ,(`https://en.wikipedia.org/wiki/Depth_map`) (Figure 2). What is depth map? Figure ??? How to generate from 3d models How to extract distance from depth map. Figure ???

,(Figure 3),,agent ,agent,agent. Figure 1: agent, Figure 2: depth map generated from 3d models Figure 3: o Reimplement q-learning o Feeding distance to q-learning Different Implementaion o Data: volume??

1.3.4 Results and discussion:

o ,agent. . o 1.2.5.1 o . ,agent.. o ,agent

2 a

agent. ,,(Figure 1), ,

3 a

DQN githubDQNAtari. ,Atari. Atari,DQNcnn. 1.2.4 1.2.4.1 ,agent,.

Q learning Describe: illustrated in Figure 1.2: from youtube Action Action value function

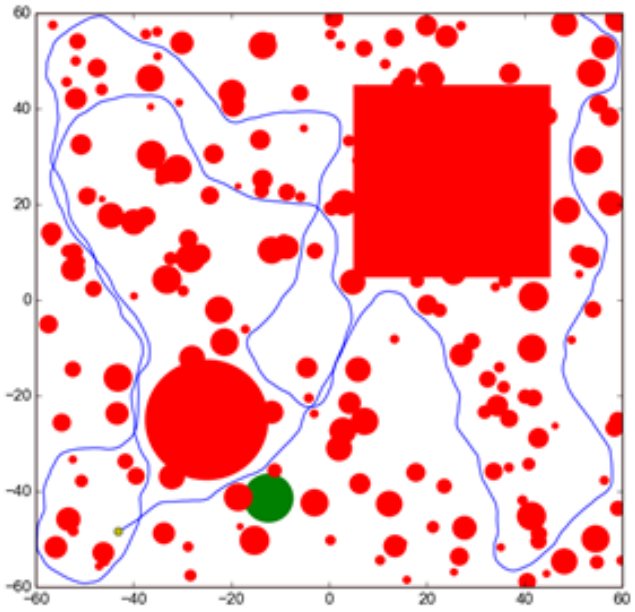

Figure 5: (mark the poles)(Green circle/blue lines remove later) Different Red shapes are obstaclesi Green destination(remove)Blue line routes (remove)

Figure 1.2 Describe the method used in the video