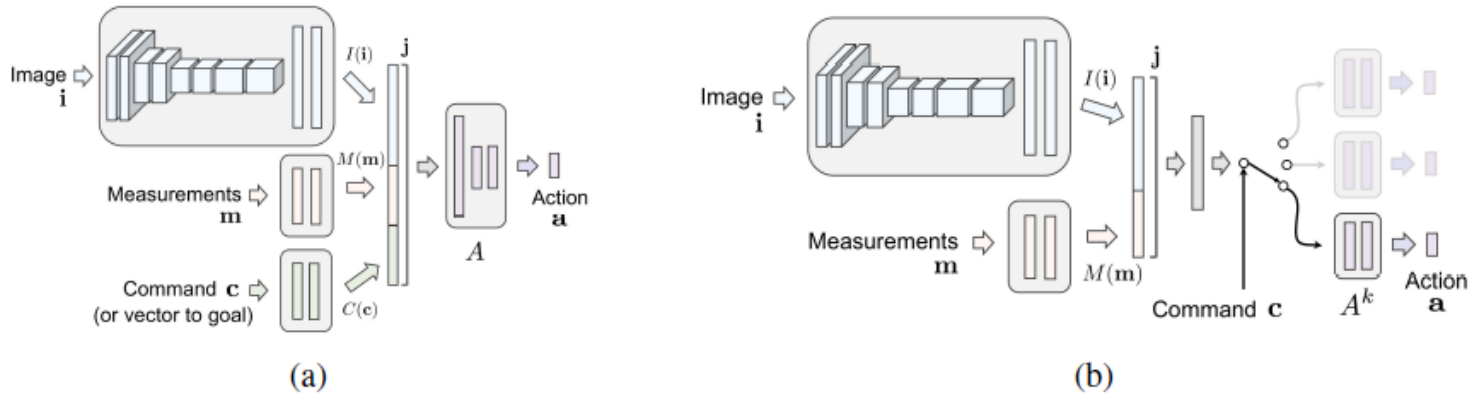
Literature Review -

End-to-end Learning and Reinforcement Learning in Obstacle Avoidance

End-to-end learning with image input

- **End-to-End Driving via Conditional Imitation Learning (2018)**: At training time, the model is given not only the perceptual input and the control signal, but also a representation of the expert's intention.



Link: https://arxiv.org/abs/1710.02410

End-to-end learning with image input

- **Off-Road Obstacle Avoidance through End-to-End Learning (2005)**: using two cameras as input of the neural network.

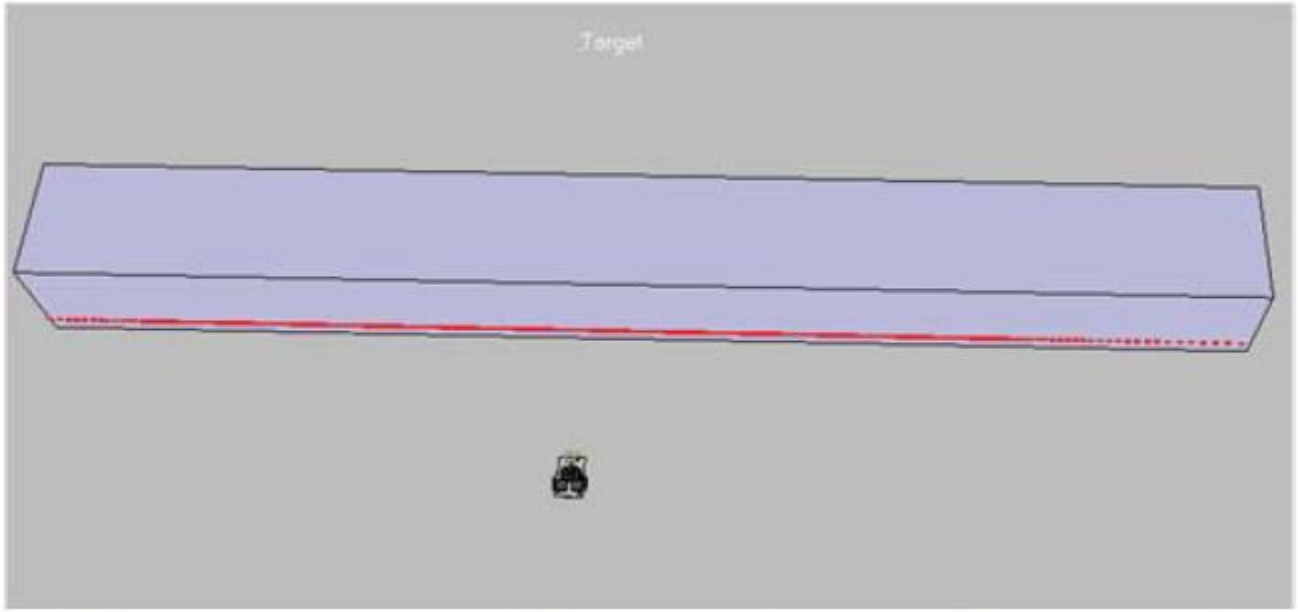
- **End-to-End Deep Learning for Autonomous Navigation of Mobile Robot (2018)**: regular methods of using single image. Nothing original.

- **Vision-Based Mobile Robot Navigation through Deep Convolutional Neural Networks and End-to-End Learning (2017)**: Nothing original.

- **CNN-Based Vision Model for Obstacle Avoidance of Mobile Robot (2017)**: Nothing original.

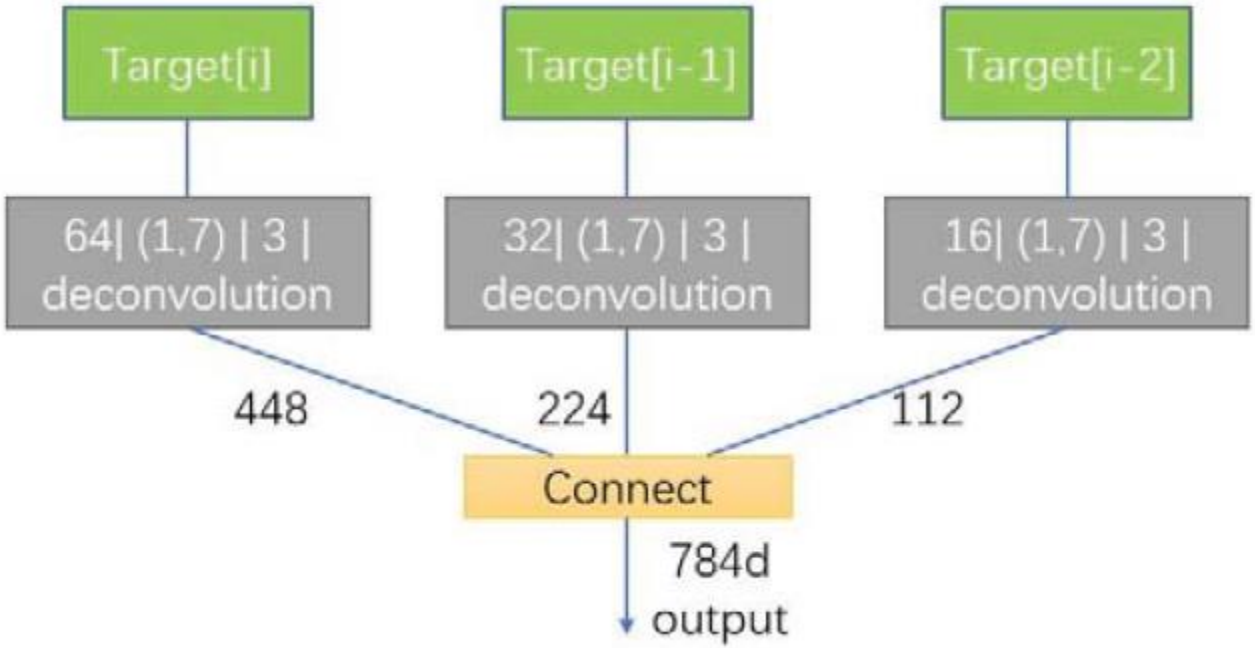
End-to-end learning with laser scan input

- **A LIDAR based end to end controller for robot navigation using deep neural network (2017)**: A sliding window of past information is incorporated to add the memory to controller, so that the hesitation is reduced when ambiguity occurs.



Link: https://ieeexplore.ieee.org/document/8278417/

End-to-end learning with laser scan input

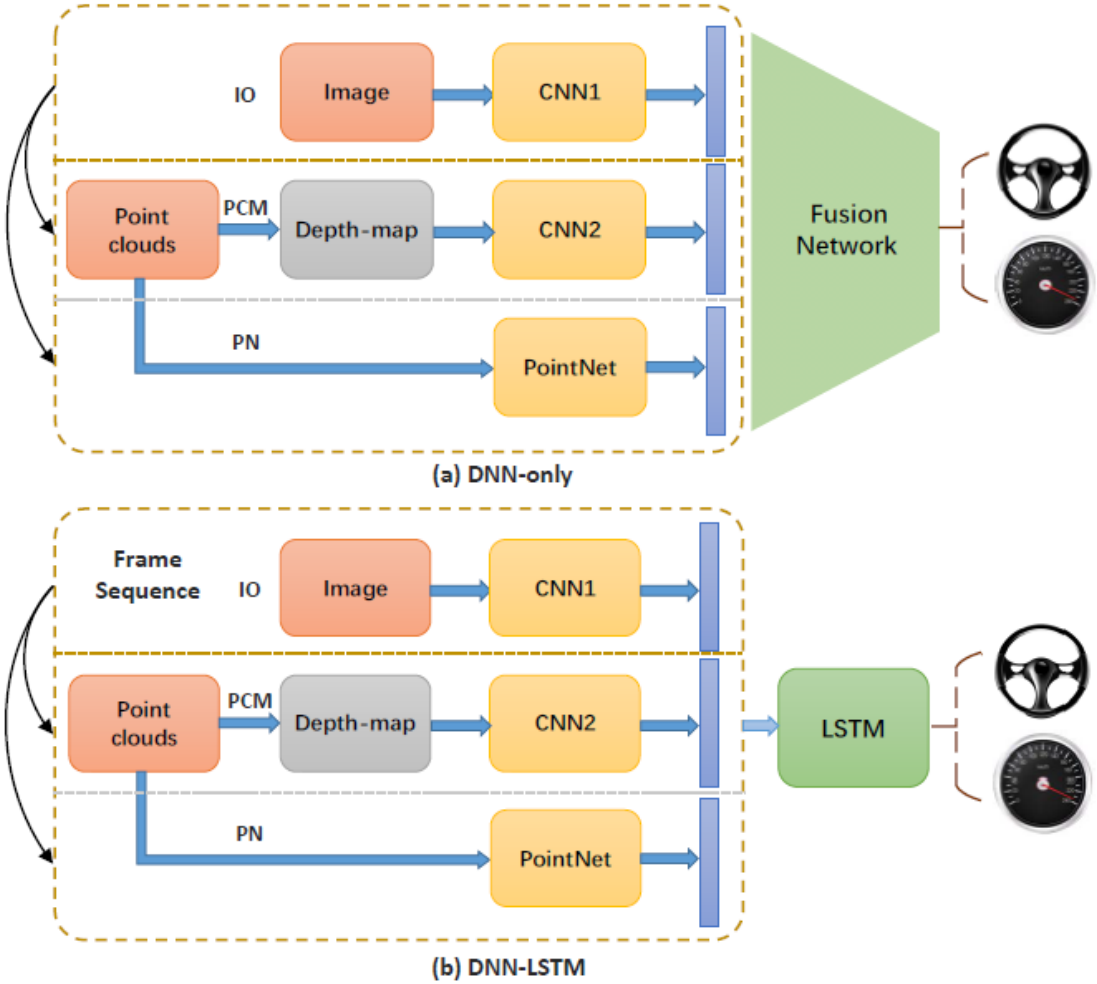


The target information is a combination of the last three frames of target information. Then they would be inserted into fully-connected layers to output the control commands.

End-to-end learning with laser scan and images input

- **LIDAR-Video Driving Dataset: Learning Driving Policies Effectively (2018)**: a dataset which is the first policy learning benchmark composed of driving videos, LIDAR data and corresponding driving behaviors.

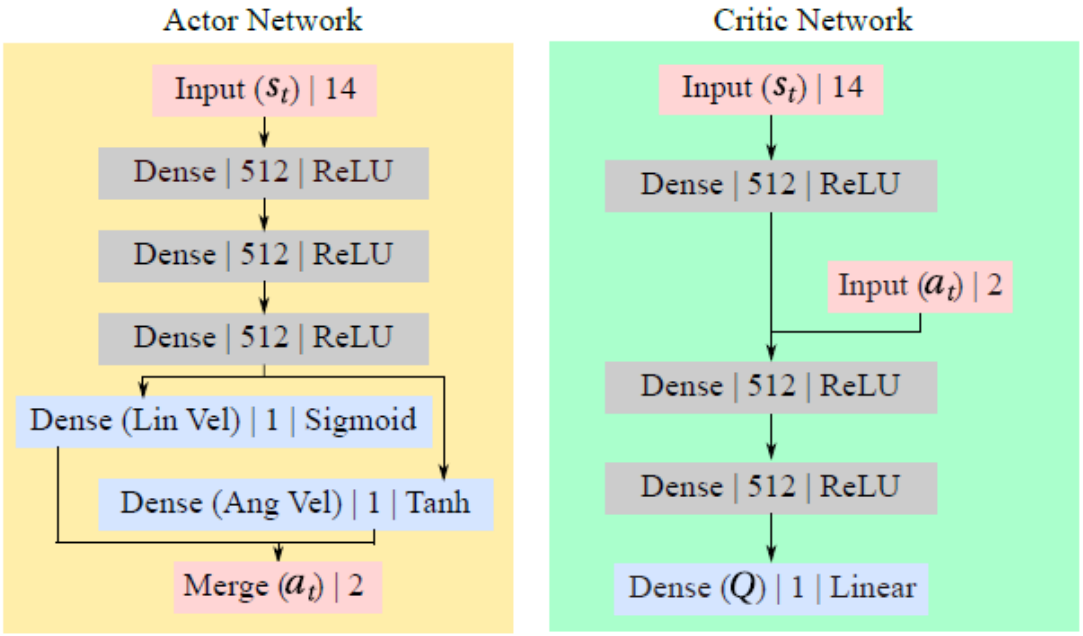
- DNN + LSTM. Long short-term memory (LSTM) is a well-improved recurrent neural network by introducing memory gates.
- Depth representation: Point Clouds Mapping: LIDAR points are divided into grids on XOZ plane and we could get the nearest points of Y coordinate in each grid. PointNet: It directly take disorder pointes as the input of neural networks and finally output the representation features.
- Vehicle speed and steering angle prediction models are trained individually.



Link: http://openaccess.thecvf.com/content_cvpr_2018/html/Chen_LiDAR-Video_Driving_Dataset_CVPR_2018_paper.html

End-to-end reinforcement learning with laser scan input

- **Virtual-to-real Deep Reinforcement Learning: Continuous Control of Mobile Robots for Mapless Navigation (2017)**: taking LIDAR data and target position as input and steering commands as output. Using asynchronous DDPG method.



End-to-end reinforcement learning with laser scan input

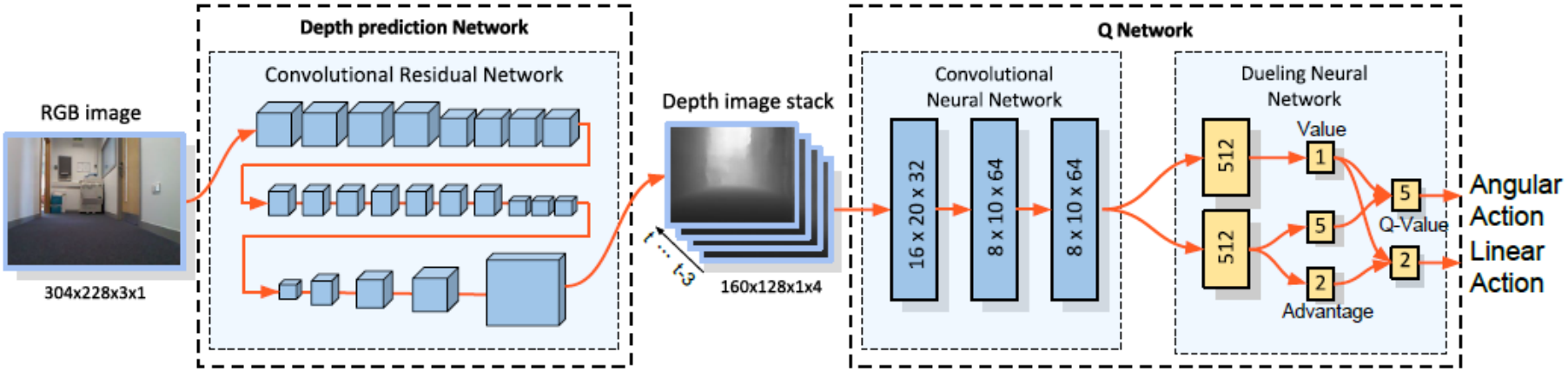
- Reward function definition

$$r(s_t, a_t) = \begin{cases} r_{arrive} \text{ if } d_t < c_d \\ r_{collision} \text{ if } min_{x_t} < c_o \\ c_r(d_{t-1} - d_t) \end{cases}$$

Link: https://ieeexplore.ieee.org/document/8202134/

Reinforcement learning with image input

- **Towards Monocular Vision based Obstacle Avoidance through Deep Reinforcement Learning (2017)**: a dueling architecture based deep double-Q network is proposed, using only monocular RGB vision.



Link: https://arxiv.org/abs/1706.09829

Reinforcement learning with image/laser scan input

- **Uncertainty-Aware Reinforcement Learning for Collision Avoidance (2017)**: presenting an uncertainty-aware model-based learning algorithm that estimates the probability of collision together with a statistical estimate of uncertainty.

- The collision prediction model is a fully connected neural network (policy gradient method).
- Inputs are current state, the sensor observation, and a sequence of controls.
- Outputs are the probability of collision, the uncertainty which is the variance, and a collision cost which is used in MPC.
- In order to obtain accurate uncertainty, they use two techniques: bootstrapping and dropout.

Reinforcement learning with image/laser scan input

Algorithm 1 Neural net training with bootstrapping and dropout

input: dataset $\mathcal{D} = \{\mathbf{x}_t^{(i)}, \mathbf{u}_{t:t+H}^{(i)}, \mathbf{o}_t^{(i)}\}$, neural network model NN
for $b = 1$ to B **do**
  Sample a dataset of subsequences $\mathcal{D}^{(b)}$ from the full dataset $\mathcal{D}$ with replacement
  Initialize neural network $\text{NN}^{(b)}$ with random weights
  for number of SGD iterations **do**
    Sample datapoint $(\mathbf{x}_t, \mathbf{u}_{t:t+H}, \mathbf{o}_t)$ from $\mathcal{D}^{(b)}$
    Sample $\text{NN}_d^{(b)}$ by masking the units in $\text{NN}^{(b)}$ using dropout
    Run forward pass on $\text{NN}_d^{(b)}$ using $(\mathbf{x}_t, \mathbf{u}_{t:t+H}, \mathbf{o}_t)$
    Run backward pass on $\text{NN}_d^{(b)}$ to get gradient $g_d^{(b)}$
    Update model $\text{NN}^{(b)}$ parameters using $g_d^{(b)}$
  end for
end for

Reinforcement learning with image/laser scan input

Algorithm 2 RL with Risk-Averse Collision Estimates

1: Initialize empty dataset $\mathcal{D}$
2: Initialize collision prediction model $\widetilde{P}_\theta$
3: **for** iter=1 to max_iter **do**
4: Sample trajectories $\{\tau_i\}$ using MPC with cost $\mathcal{C}$
5: Add samples $\{\tau_i\}$ to $\mathcal{D}$
6: Train $\widetilde{P}_\theta$ using $\mathcal{D}$ (Alg. 1)
7: **end for**

Link: https://arxiv.org/abs/1702.01182

Reinforcement learning

- **Motion Planning Among Dynamic, Decision-Making Agents with Deep Reinforcement Learning (2018)**: an algorithm that learns collision avoidance among a variety of types of dynamics agents without assuming they follow any particular behavior rules. It is a kind of modified actor-critic method.

- **Neural networks based reinforcement learning for mobile robots obstacle avoidance (2016)**: providing mobile robots a collision-free trajectory within an uncertain workspace which contains both stationary and moving entities. It is a modified Q-Learning approach.

Reinforcement learning

- **End-to-End Race Driving with Deep Reinforcement Learning (2018)**: including hand brake command in the learning algorithm to enforce drifting. An Asynchronous Actor-Critic framework is used. RGB Image input.

- **Avoiding Moving Obstacles with Stochastic Hybrid Dynamics using PEARL: PrEference Appraisal Reinforcement Learning (2016)**: PEARL projects the high-dimensional continuous robot state space to a low dimensional preference feature space resulting in efficient and adaptable planning.