Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

MIN Faculty
Department of Informatics

Mobile robot avoid obstacles using Monocular Vision

Hongzhuo Liang

University of Hamburg
Faculty of Mathematics, Informatics and Natural Sciences
Department of Informatics
Technical Aspects of Multimodal Systems

28. Nov. 2017

Outline

Outline

Motivation: Why use Monocular Vision

In the paper *Deep Learning* from Nature: [LeCun et al., 2015][1]

> *The future of deep learning: Unsupervised learning had a catalytic effect in reviving interest in deep learning, we expect unsupervised learning to become far more important in the longer term. Human and animal learning is largely unsupervised: we discover the structure of the world by observing it, not by being told the name of every object.*

[1] https://www.nature.com/articles/nature14539

Motivation: Why use Monocular Vision

In the paper *Deep Learning* from Nature: [LeCun et al., 2015][1]

> *The future of deep learning: Unsupervised learning had a catalytic effect in reviving interest in deep learning, we expect unsupervised learning to become far more important in the longer term. Human and animal learning is largely unsupervised: we discover the structure of the world by observing it, not by being told the name of every object.*

[1] https://www.nature.com/articles/nature14539

Motivation

Advantage

- Cheap
- Easy to get
- Flexible to complicated environments

Disadvantage

- Not as accuracy as depth sensor
- Need a lot of calculate power
- Need a lot of training
- Not stable in current time

Motivation

Advantage

- Cheap
- Easy to get
- Flexible to complicated environments

Disadvantage

- Not as accuracy as depth sensor
- Need a lot of calculate power
- Need a lot of training
- Not stable in current time

Motivation

Advantage

- Cheap
- Easy to get
- Flexible to complicated environments

Disadvantage

- Not as accuracy as depth sensor
- Need a lot of calculate power
- Need a lot of training
- Not stable in current time

Motivation

Advantage

- Cheap
- Easy to get
- Flexible to complicated environments

Disadvantage

- Not as accuracy as depth sensor
- Need a lot of calculate power
- Need a lot of training
- Not stable in current time

Motivation

Advantage

- Cheap
- Easy to get
- Flexible to complicated environments

Disadvantage

- Not as accuracy as depth sensor
- Need a lot of calculate power
- Need a lot of training
- Not stable in current time

Motivation

Advantage

- Cheap
- Easy to get
- Flexible to complicated environments

Disadvantage

- Not as accuracy as depth sensor
- Need a lot of calculate power
- Need a lot of training
- Not stable in current time

Motivation

Advantage

- Cheap
- Easy to get
- Flexible to complicated environments

Disadvantage

- Not as accuracy as depth sensor
- Need a lot of calculate power
- Need a lot of training
- Not stable in current time

Motivation

Advantage

- Cheap
- Easy to get
- Flexible to complicated environments

Disadvantage

- Not as accuracy as depth sensor
- Need a lot of calculate power
- Need a lot of training
- Not stable in current time

Motivation

Advantage

- Cheap
- Easy to get
- Flexible to complicated environments

Disadvantage

- Not as accuracy as depth sensor
- Need a lot of calculate power
- Need a lot of training
- Not stable in current time

Related Work: Use Traditional Ways

In the traditional way of doing obstacle avoidance task, one should use the camera to detect the point flow to get the information about the environment. For example,the position of the floor and obstacles. Then we can use some fixed way to plan and executes.

- Use Depth camera to get Point Cloud
- Use the Point Cloud to fitting a function about the ground
- Calculate the distance between every point with the fitting ground h
- If h is larger than a setting threshold, then marke it as an obstacle
- Transfer the Point Cloud data of the obstacle to the robot coordinate space
- Combine the current robot moving stats with the obstacle to decide how to avoid the obstacles

Related Work: Use Traditional Ways

In the traditional way of doing obstacle avoidance task, one should use the camera to detect the point flow to get the information about the environment. For example,the position of the floor and obstacles. Then we can use some fixed way to plan and executes.

- Use Depth camera to get Point Cloud
- Use the Point Cloud to fitting a function about the ground
- Calculate the distance between every point with the fitting ground h
- If h is larger than a setting threshold, then marke it as an obstacle
- Transfer the Point Cloud data of the obstacle to the robot coordinate space
- Combine the current robot moving stats with the obstacle to decide how to avoid the obstacles

Related Work: Use Traditional Ways

In the traditional way of doing obstacle avoidance task, one should use the camera to detect the point flow to get the information about the environment. For example,the position of the floor and obstacles. Then we can use some fixed way to plan and executes.

- Use Depth camera to get Point Cloud
- Use the Point Cloud to fitting a function about the ground
- Calculate the distance between every point with the fitting ground h
- If h is larger than a setting threshold, then marke it as an obstacle
- Transfer the Point Cloud data of the obstacle to the robot coordinate space
- Combine the current robot moving stats with the obstacle to decide how to avoid the obstacles

Related Work: Use Traditional Ways

In the traditional way of doing obstacle avoidance task, one should use the camera to detect the point flow to get the information about the environment. For example,the position of the floor and obstacles. Then we can use some fixed way to plan and executes.

- Use Depth camera to get Point Cloud
- Use the Point Cloud to fitting a function about the ground
- Calculate the distance between every point with the fitting ground h
- If h is larger than a setting threshold, then marke it as an obstacle
- Transfer the Point Cloud data of the obstacle to the robot coordinate space
- Combine the current robot moving stats with the obstacle to decide how to avoid the obstacles

Related Work: Use Traditional Ways

In the traditional way of doing obstacle avoidance task, one should use the camera to detect the point flow to get the information about the environment. For example,the position of the floor and obstacles. Then we can use some fixed way to plan and executes.

- Use Depth camera to get Point Cloud
- Use the Point Cloud to fitting a function about the ground
- Calculate the distance between every point with the fitting ground h
- If h is larger than a setting threshold, then marke it as an obstacle
- Transfer the Point Cloud data of the obstacle to the robot coordinate space
- Combine the current robot moving stats with the obstacle to decide how to avoid the obstacles

Related Work: Use Traditional Ways

In the traditional way of doing obstacle avoidance task, one should use the camera to detect the point flow to get the information about the environment. For example,the position of the floor and obstacles. Then we can use some fixed way to plan and executes.

- Use Depth camera to get Point Cloud
- Use the Point Cloud to fitting a function about the ground
- Calculate the distance between every point with the fitting ground h
- If h is larger than a setting threshold, then marke it as an obstacle
- Transfer the Point Cloud data of the obstacle to the robot coordinate space
- Combine the current robot moving stats with the obstacle to decide how to avoid the obstacles

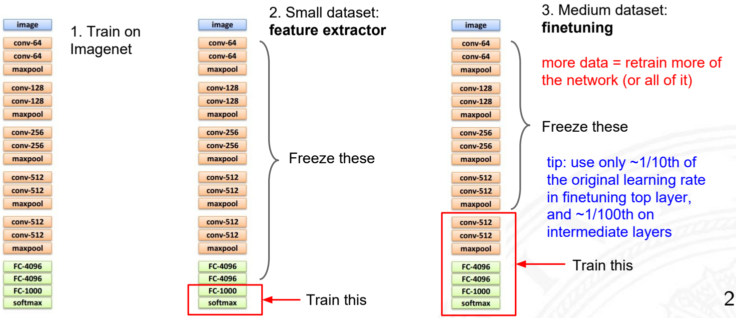
Related Work: Use Traditional Ways

In the traditional way of doing obstacle avoidance task, one should use the camera to detect the point flow to get the information about the environment. For example,the position of the floor and obstacles. Then we can use some fixed way to plan and executes.

- Use Depth camera to get Point Cloud
- Use the Point Cloud to fitting a function about the ground
- Calculate the distance between every point with the fitting ground h
- If h is larger than a setting threshold, then marke it as an obstacle
- Transfer the Point Cloud data of the obstacle to the robot coordinate space
- Combine the current robot moving stats with the obstacle to decide how to avoid the obstacles

Related Work: Use CNN and transfer learning

Transfer Learning with CNNs



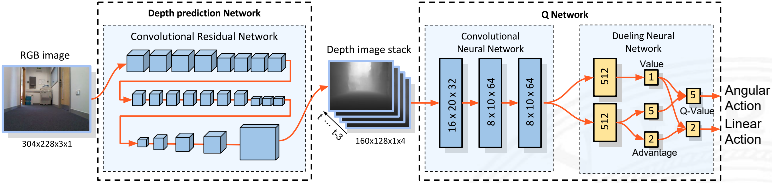
[2]

Use transfer learning (Fine tune) is also a good way to improve the training process.

[2] http://cs231n.stanford.edu/slides/2016/winter1516_lecture11.pdf

Related Work: Using CNN and reinforcement learning

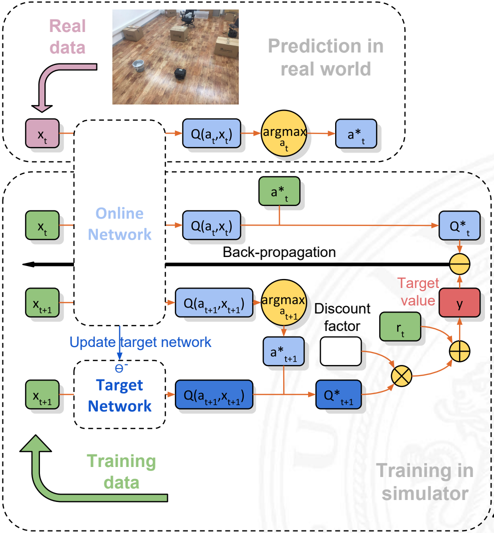
[Xie et al., 2017] Towards Monocular Vision based Obstacle Avoidance through Deep Reinforcement Learning. [3]



This paper uses CNN to get a "Depth Image stack", then uses D3QN to get action.

[3] https://arxiv.org/abs/1706.09829

Related Work: Using CNN and reinforcement learning



[4]

[4] https://arxiv.org/abs/1706.09829

Outline

Try to use CNN

Pipeline:

- Collect data and labeling
- Build CNN to train
- Get the result in test set

Try to use CNN

Pipeline:

- Collect data and labeling
- Build CNN to train
- Get the result in test set

Try to use CNN

Pipeline:

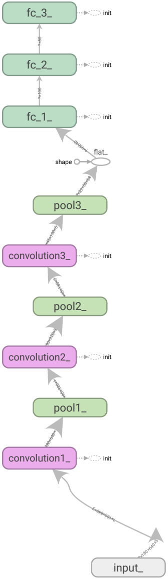
- Collect data and labeling
- Build CNN to train
- Get the result in test set

Try to use CNN

Pipeline:

- Collect data and labeling
- Build CNN to train
- Get the result in test set

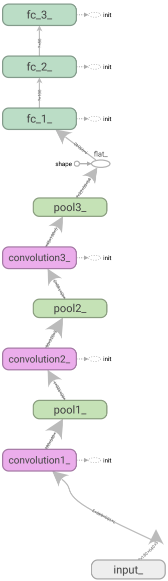
Try to use CNN: Structure of my CNN



Framwork of the CNN

- Convolutional Networks
  - Conv 1: filter 16, kernel size 5, strides 1
  - Conv 2: filter 32, kernel size 5, strides 1
  - Conv 3: filter 54, kernel size 5, strides 1
- Fully connected Networks

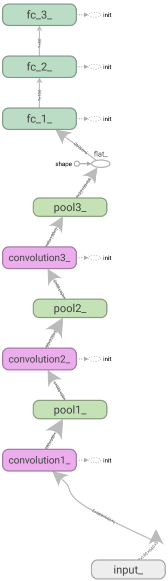
Try to use CNN: Structure of my CNN



Framwork of the CNN

- Convolutional Networks
  - Conv 1: filter 16, kernel size 5, strides 1
  - Conv 2: filter 32, kernel size 5, strides 1
  - Conv 3: filter 54, kernel size 5, strides 1
- Fully connected Networks

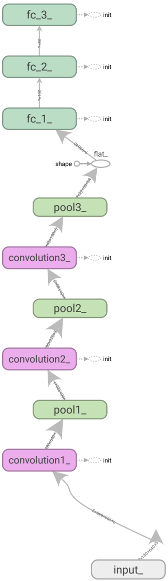
Try to use CNN: Structure of my CNN



Framwork of the CNN

- Convolutional Networks
  - Conv 1: filter 16, kernel size 5, strides 1
  - Conv 2: filter 32, kernel size 5, strides 1
  - Conv 3: filter 54, kernel size 5, strides 1
- Fully connected Networks

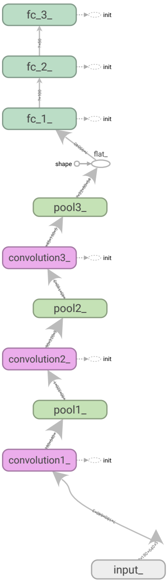
Try to use CNN: Structure of my CNN



Framwork of the CNN

- Convolutional Networks
  - Conv 1: filter 16, kernel size 5, strides 1
  - Conv 2: filter 32, kernel size 5, strides 1
  - Conv 3: filter 54, kernel size 5, strides 1
- Fully connected Networks

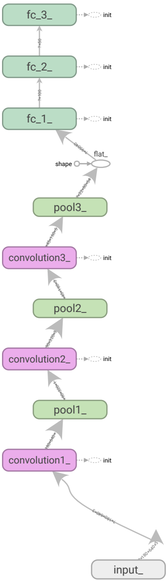
Try to use CNN: Structure of my CNN



Framwork of the CNN

- Convolutional Networks
  - Conv 1: filter 16, kernel size 5, strides 1
  - Conv 2: filter 32, kernel size 5, strides 1
  - Conv 3: filter 54, kernel size 5, strides 1
- Fully connected Networks
  - Flat 1: 100 nodes
  - Flat 2: 50 nodes
  - Flat 3: 2 nodes

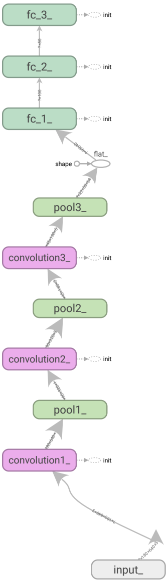
Try to use CNN: Structure of my CNN



Framwork of the CNN

- Convolutional Networks
  - Conv 1: filter 16, kernel size 5, strides 1
  - Conv 2: filter 32, kernel size 5, strides 1
  - Conv 3: filter 54, kernel size 5, strides 1
- Fully connected Networks
  - Flat 1: 100 nodes
  - Flat 2: 50 nodes
  - Flat 3: 2 nodes

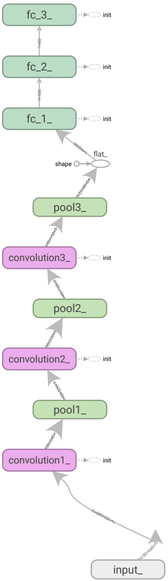
Try to use CNN: Structure of my CNN



Framwork of the CNN

- Convolutional Networks
  - Conv 1: filter 16, kernel size 5, strides 1
  - Conv 2: filter 32, kernel size 5, strides 1
  - Conv 3: filter 54, kernel size 5, strides 1
- Fully connected Networks
  - Flat 1: 100 nodes
  - Flat 2: 50 nodes
  - Flat 3: 2 nodes

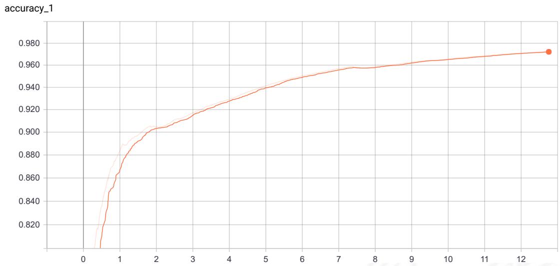
Try to use CNN: Structure of my CNN

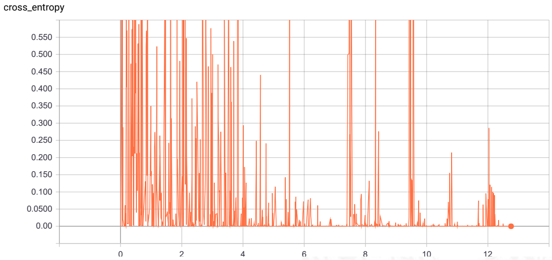


Framwork of the CNN

- Convolutional Networks
  - Conv 1: filter 16, kernel size 5, strides 1
  - Conv 2: filter 32, kernel size 5, strides 1
  - Conv 3: filter 54, kernel size 5, strides 1
- Fully connected Networks
  - Flat 1: 100 nodes
  - Flat 2: 50 nodes
  - Flat 3: 2 nodes

Try to use CNN: Results





Use self supervised learning

The basic idea of this is to let robot get labels by the additional sensing such as range finder or collision detection switch. But this way is highly relying on the labeling rules.

- Using depth sensor to get "ground truth"
- Using the bumper in front of the turtlebot

Use self supervised learning

The basic idea of this is to let robot get labels by the additional sensing such as range finder or collision detection switch. But this way is highly relying on the labeling rules.

- Using depth sensor to get "ground truth"
- Using the bumper in front of the turtlebot

Use self supervised learning

The basic idea of this is to let robot get labels by the additional sensing such as range finder or collision detection switch. But this way is highly relying on the labeling rules.

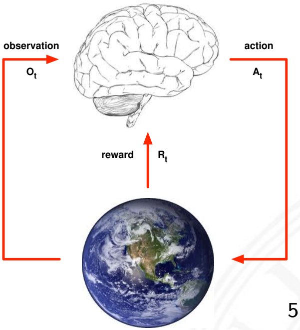
- Using depth sensor to get "ground truth"
- Using the bumper in front of the turtlebot

Use self supervised learning

The basic idea of this is to let robot get labels by the additional sensing such as range finder or collision detection switch. But this way is highly relying on the labeling rules.

- Using depth sensor to get "ground truth"
- Using the bumper in front of the turtlebot

Use Reinforcement Learning: What is RL



[5]

At each step t the agent:

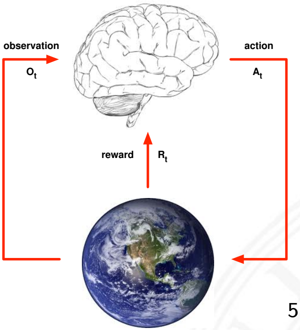
- Executes action A_t
- Receives observation O_t
- Receives scalar reward R_t

The environment:

- Receives action A_t
- Emits observation O_{t+1}
- Emits scalar reward R_{t+1}

[5] http://www0.cs.ucl.ac.uk/staff/D.Silver/web/Teaching.html

Use Reinforcement Learning: What is RL



[5]

At each step t the agent:

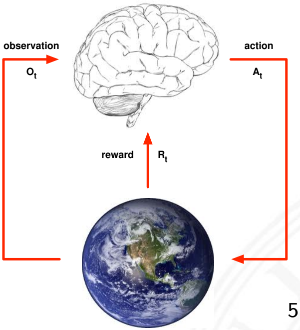
- Executes action A_t
- Receives observation O_t
- Receives scalar reward R_t

The environment:

- Receives action A_t
- Emits observation O_{t+1}
- Emits scalar reward R_{t+1}

[5] http://www0.cs.ucl.ac.uk/staff/D.Silver/web/Teaching.html

Use Reinforcement Learning: What is RL



5

At each step t the agent:

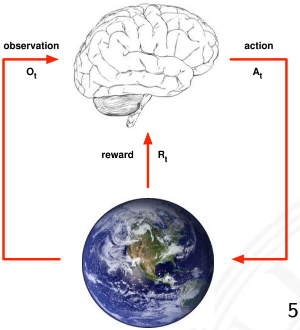
- Executes action A_t
- Receives observation O_t
- Receives scalar reward R_t

The environment:

- Receives action A_t
- Emits observation O_{t+1}
- Emits scalar reward R_{t+1}

[5] http://www0.cs.ucl.ac.uk/staff/D.Silver/web/Teaching.html

Use Reinforcement Learning: What is RL



[5]

At each step t the agent:

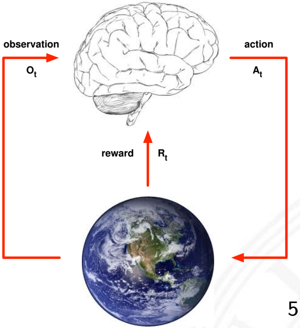
- Executes action A_t
- Receives observation O_t
- Receives scalar reward R_t

The environment:

- Receives action A_t
- Emits observation O_{t+1}
- Emits scalar reward R_{t+1}

[5] http://www0.cs.ucl.ac.uk/staff/D.Silver/web/Teaching.html

Use Reinforcement Learning: What is RL



[5]

At each step t the agent:

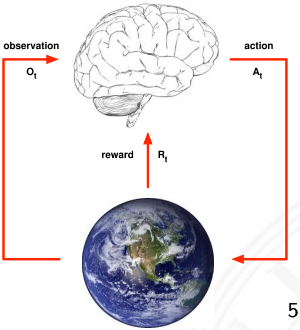
- Executes action A_t
- Receives observation O_t
- Receives scalar reward R_t

The environment:

- Receives action A_t
- Emits observation O_{t+1}
- Emits scalar reward R_{t+1}

[5] http://www0.cs.ucl.ac.uk/staff/D.Silver/web/Teaching.html

Use Reinforcement Learning: What is RL



[5]

At each step t the agent:

- Executes action A_t
- Receives observation O_t
- Receives scalar reward R_t

The environment:

- Receives action A_t
- Emits observation O_{t+1}
- Emits scalar reward R_{t+1}

[5] http://www0.cs.ucl.ac.uk/staff/D.Silver/web/Teaching.html

Use Reinforcement Learning: Q-Learning

The mission of RL is to find a *best policy* in order to make the reward more.

The value function(Bellman Function) describes how to get the value of current state, that is,

$$V(s) = \mathbb{E}[R_{t+1} + \lambda v(S_{t+1}) | S_t = s]$$

To update the Q value, that is,

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \lambda \max_a Q(S_{t+1}, a) - Q(S_t, A_t))$$

The mission of RL is to find a *best policy* in order to make the reward more.

The value function(Bellman Function) describes how to get the value of current state, that is,

$$V(s) = \mathbb{E}[R_{t+1} + \lambda v(S_{t+1}) | S_t = s]$$

To update the Q value, that is,

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \lambda \max_a Q(S_{t+1}, a) - Q(S_t, A_t))$$

Use Reinforcement Learning: Q-Learning

The mission of RL is to find a *best policy* in order to make the reward more.

The value function(Bellman Function) describes how to get the value of current state, that is,

$$V(s) = \mathbb{E}[R_{t+1} + \lambda v(S_{t+1}) | S_t = s]$$

To update the Q value, that is,

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \lambda \max_a Q(S_{t+1}, a) - Q(S_t, A_t))$$

Use Reinforcement Learning: DQN

In the Q-Learning we give a Q table:

	a_1	a_2	a_3	a_4
s_1	Q(1,1)	Q(1,2)	Q(1,3)	Q(1,4)
s_2	Q(2,1)	Q(2,2)	Q(2,3)	Q(2,4)
s_3	Q(3,1)	Q(3,2)	Q(3,3)	Q(3,4)
s_4	Q(4,1)	Q(4,2)	Q(4,3)	Q(4,4)

Due to store too many states in one Q table is unrealistic in DQN, we get Q table from a network.

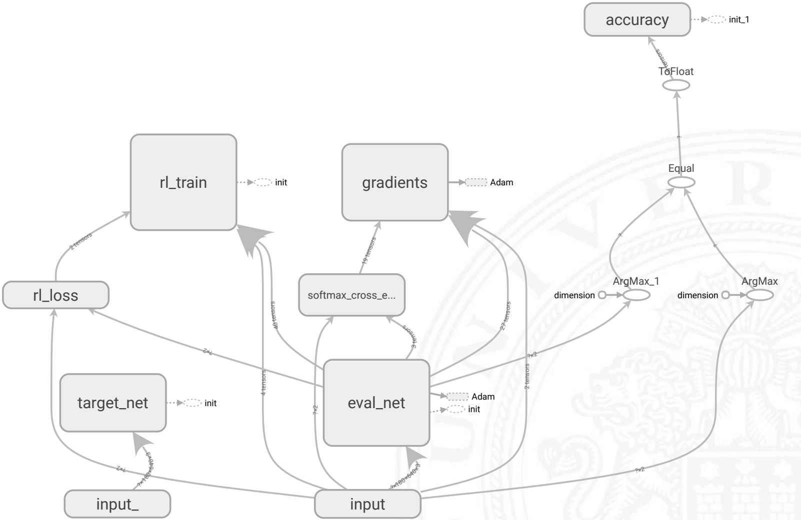
Use Reinforcement Learning: DQN

In the Q-Learning we give a Q table:

	a_1	a_2	a_3	a_4
s_1	Q(1,1)	Q(1,2)	Q(1,3)	Q(1,4)
s_2	Q(2,1)	Q(2,2)	Q(2,3)	Q(2,4)
s_3	Q(3,1)	Q(3,2)	Q(3,3)	Q(3,4)
s_4	Q(4,1)	Q(4,2)	Q(4,3)	Q(4,4)

Due to store too many states in one Q table is unrealistic in DQN, we get Q table from a network.

Use Reinforcement Learning: DQN Algorithm



Use Reinforcement Learning: Get Reward

```
def get_reward(self, action):
    if self.flag == action:
        if action == 0:
            reward = 1
        else:
            reward = 0
    else:
        reward = -1
    return reward
```

Flag is whether the robot is crashed a bumper. In the future, we need to add more actions.

Outline

Experiment using CNN

Experiment 2

Experiment using DQN

Outline

Conclusion

- Traditional way by using depth sensor to plan a robot is good enough, but this kind of method is highly relied on expensive sensor.
- By using CNN the Monocular camera can get very good result in both training data and test data. But it will get bad result in unknown environment.
- Self-supervised learning can reduce the backward of the pure CNN, by self collecting data on the run.
- By using CNN to get a predict depth image then use this as input to train the RL network is very interesting, and it avoid the backward of using bad simulated camera in simulation environment.
- Using DQN to train the network directly is more intelligence, need to do Some furthermore researching on it.

Conclusion

- Traditional way by using depth sensor to plan a robot is good enough, but this kind of method is highly relied on expensive sensor.
- By using CNN the Monocular camera can get very good result in both training data and test data. But it will get bad result in unknown environment.
- Self-supervised learning can reduce the backward of the pure CNN, by self collecting data on the run.
- By using CNN to get a predict depth image then use this as input to train the RL network is very interesting, and it avoid the backward of using bad simulated camera in simulation environment.
- Using DQN to train the network directly is more intelligence, need to do Some furthermore researching on it.

Conclusion

- Traditional way by using depth sensor to plan a robot is good enough, but this kind of method is highly relied on expensive sensor.
- By using CNN the Monocular camera can get very good result in both training data and test data. But it will get bad result in unknown environment.
- Self-supervised learning can reduce the backward of the pure CNN, by self collecting data on the run.
- By using CNN to get a predict depth image then use this as input to train the RL network is very interesting, and it avoid the backward of using bad simulated camera in simulation environment.
- Using DQN to train the network directly is more intelligence, need to do Some furthermore researching on it.

Conclusion

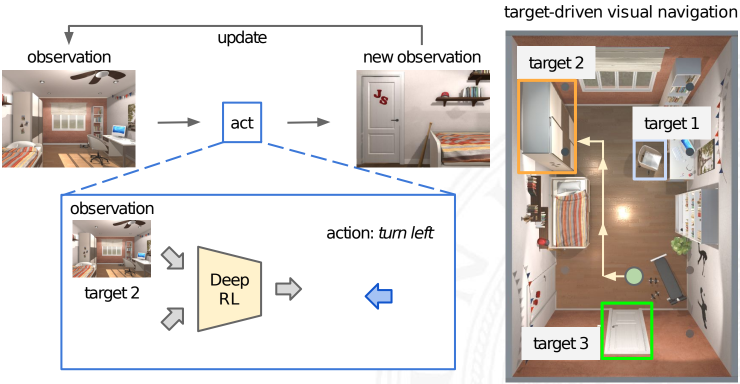
- Traditional way by using depth sensor to plan a robot is good enough, but this kind of method is highly relied on expensive sensor.
- By using CNN the Monocular camera can get very good result in both training data and test data. But it will get bad result in unknown environment.
- Self-supervised learning can reduce the backward of the pure CNN, by self collecting data on the run.
- By using CNN to get a predict depth image then use this as input to train the RL network is very interesting, and it avoid the backward of using bad simulated camera in simulation environment.
- Using DQN to train the network directly is more intelligence, need to do Some furthermore researching on it.

Conclusion

- Traditional way by using depth sensor to plan a robot is good enough, but this kind of method is highly relied on expensive sensor.
- By using CNN the Monocular camera can get very good result in both training data and test data. But it will get bad result in unknown environment.
- Self-supervised learning can reduce the backward of the pure CNN, by self collecting data on the run.
- By using CNN to get a predict depth image then use this as input to train the RL network is very interesting, and it avoid the backward of using bad simulated camera in simulation environment.
- Using DQN to train the network directly is more intelligence, need to do Some furthermore researching on it.

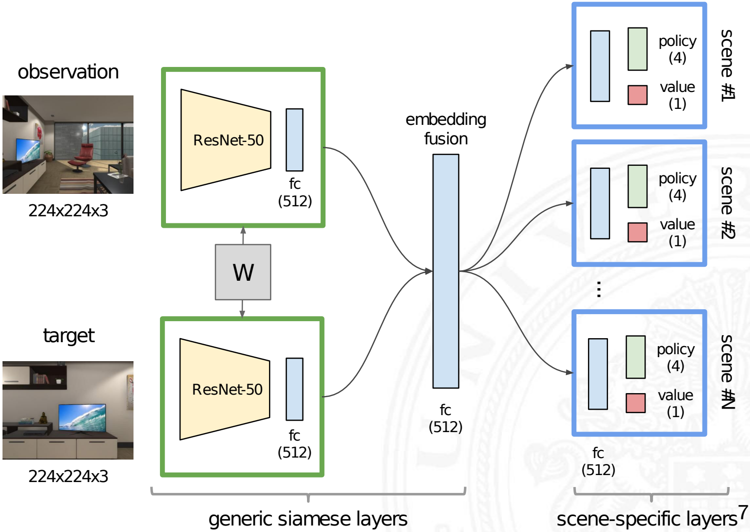
Outline

[Zhu et al., 2016] Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning [6]



[6] https://arxiv.org/pdf/1609.05143.pdf

Future Work



[7] https://arxiv.org/pdf/1609.05143.pdf

Outline

1. Introduction
2. Details
3. Experiments
4. Conclusion
5. Future Work
6. Bibliography

[LeCun et al., 2015] LeCun, Y., Bengio, Y., and Hinton, G. (2015).
Deep learning.
Nature, 521(7553):436–444.

[Xie et al., 2017] Xie, L., Wang, S., Markham, A., and Trigoni, N. (2017).
Towards Monocular Vision based Obstacle Avoidance through Deep Reinforcement Learning.

[Zhu et al., 2016] Zhu, Y., Mottaghi, R., Kolve, E., Lim, J. J., Gupta, A., Fei-Fei, L., and Farhadi, A. (2016).
Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning.
In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 3357–3364. IEEE.

[LeCun et al., 2015] LeCun, Y., Bengio, Y., and Hinton, G. (2015).
Deep learning.
Nature, 521(7553):436–444.

[Xie et al., 2017] Xie, L., Wang, S., Markham, A., and Trigoni, N. (2017).
Towards Monocular Vision based Obstacle Avoidance through Deep Reinforcement Learning.

[Zhu et al., 2016] Zhu, Y., Mottaghi, R., Kolve, E., Lim, J. J., Gupta, A., Fei-Fei, L., and Farhadi, A. (2016).
Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning.
In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 3357–3364. IEEE.

[LeCun et al., 2015] LeCun, Y., Bengio, Y., and Hinton, G. (2015).
Deep learning.
Nature, 521(7553):436–444.

[Xie et al., 2017] Xie, L., Wang, S., Markham, A., and Trigoni, N. (2017).
Towards Monocular Vision based Obstacle Avoidance through Deep Reinforcement Learning.

[Zhu et al., 2016] Zhu, Y., Mottaghi, R., Kolve, E., Lim, J. J., Gupta, A., Fei-Fei, L., and Farhadi, A. (2016).
Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning.
In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 3357–3364. IEEE.

Thank you for your attention,
Suggestion, Questions and
Commands are highly welcome.