Learning Visual Servoing with Deep Features and Fitted Q-Iteration

Alex X. Lee[1],

Sergey Levine[1], Pieter Abbeel[2,1,3]

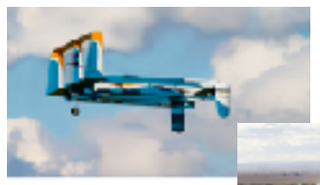
[1]UC Berkeley, [2]OpenAI,

[3]International Computer Science Institute

Hi, I’m Alex Lee, a PhD student at UC Berkeley. In this talk, I’ll present how to learn visual servoing with deep features and fitted Q-iteration. This is joint work with Sergey Levine and Pieter Abbeel.

Motivation







Robots have the potential to greatly improve human lives. In robotics manipulation, robots can help us in the house, at a factory, or in surgical procedures. In mobile robotics, self-driving cars can reduce accidents and traffic, delivery drones can reduce delivery times and costs, and ambulance drones can deliver critical medical equipment to a patient going through an emergency.

All of these robotic tasks share the same challenges: the robot needs to process complex visual information, and the robot needs to figure out how to act based on what it observes.

In order to process visual information, we can benefit from the success that CNNs have achieved in computer vision.

Deep Neural Networks in Computer Vision

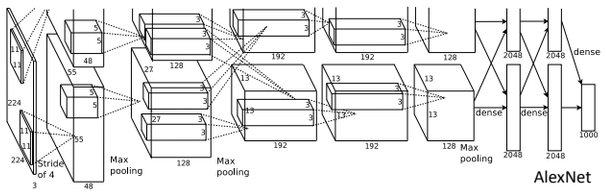
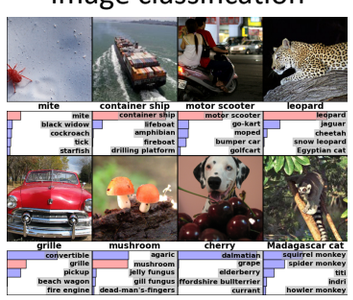
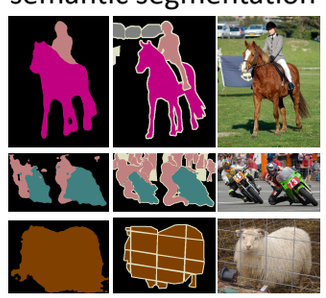


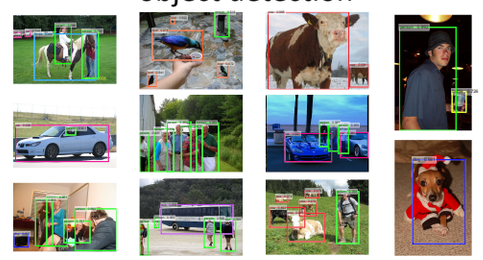
image classification



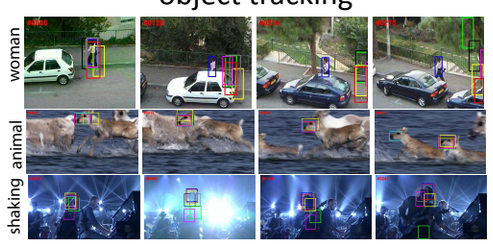
semantic segmentation



object detection



object tracking



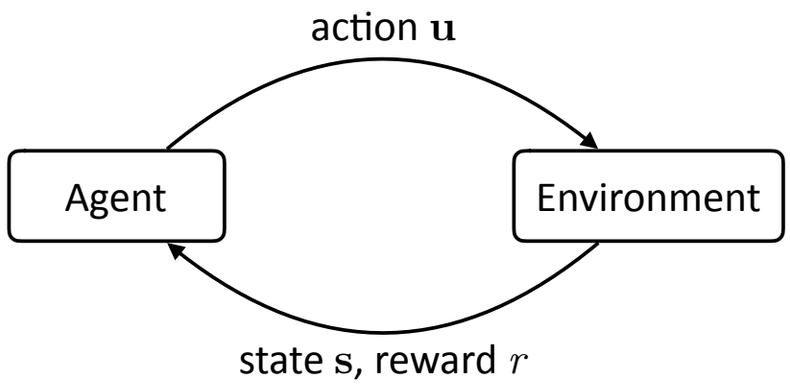
Deep CNNs are multi-layer computation graphs that takes in images and output some desired information of the images.
Deep learning has allowed to achieve impressive results in computer vision tasks, such as image classification, semantic segmentation, object detection, and object tracking.
We can leverage the success in computer vision to enable robots to understand the visual world.
What about enabling robots to learn how to act? For that, we can use reinforcement learning.

Outline

- Introduction
- **Reinforcement learning and deep reinforcement learning**
- Visual servoing
- Learn visual servoing with reinforcement learning
  - Policy optimization
  - Combine value and model based RL
    - Learn visual feature dynamics
    - Learn servoing policy with fitted Q-iteration
- Comparison to prior methods
- Conclusion

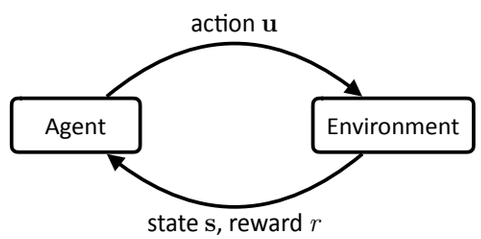
With that in mind, I'll first briefly explain what is reinforcement learning.

What is Reinforcement Learning?

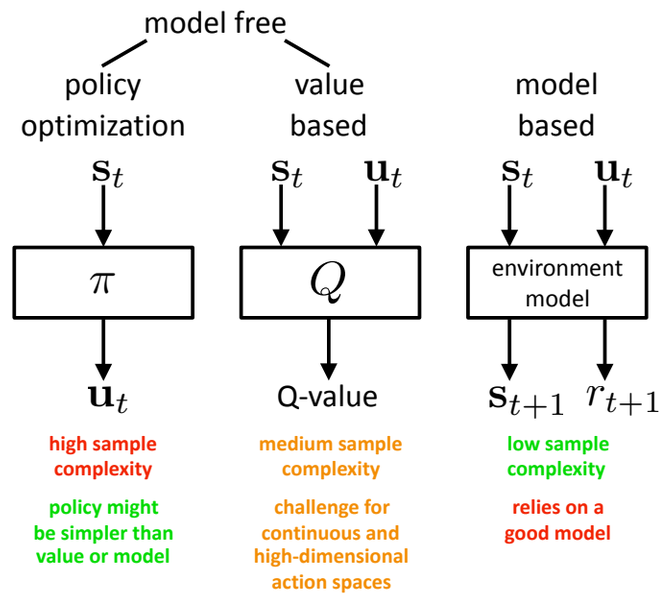


Reinforcement learning is a branch of machine learning that is concerned with taking a sequence of actions.
It is usually described in terms of an agent interacting with an environment whose goal is to maximize cumulative rewards.
For a given state, the agent acts by choosing an action according to a policy. Then, the state of the agent changes, and the agent receives a scalar reward.
In robotics, the state can be, for example, camera images or the joint angles of the robot, and the action can be the joint torques or high-level motor commands.
The reward can indicate progress towards a goal, with higher rewards being better.

Reinforcement Learning Approaches



$$\pi(\mathbf{u}_t|\mathbf{s}_t) = \arg\max_{\mathbf{u}} Q(\mathbf{s}_t, \mathbf{u})$$



There are 3 main approaches for reinforcement learning.

The first one is policy optimization, which directly learns a policy that maps from states to actions. The policy is denoted by the greek letter pi.

The second approach learns a value function Q, which maps state-action pairs to Q-values. The Q-value indicates the best cumulative sum of rewards the agent can get, conditioned on the current state s and action u. The optimal policy can readily be obtained by greedily optimizing for the action that will achieve the best Q-value from the current state.

These two approaches are called model-free reinforcement learning since it doesn't use a model of the environment.

In contrast, model-based reinforcement learning learns a model of the environment. For example, it might learn a dynamics function of how states change over time or it might learn a model of how the rewards are generated.

These approaches vary widely in terms of the number of samples needed to learn a policy. Among the RL approaches, policy optimization requires the most amount of samples, while model-based methods requires the least, and value-based methods are somewhere in between.

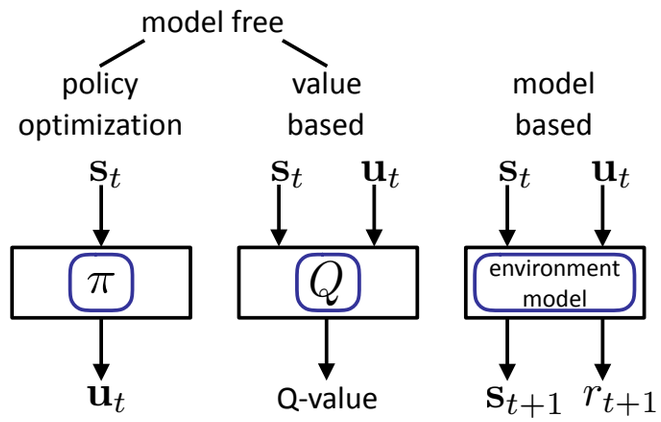
For robotic tasks, sample complexity is important because interaction in the real world is expensive and time-consuming.

However, one advantage of policy optimization is that learning the policy directly might be simpler than learning the value function or a model of the environment. For example, in robotic grasping, the value function or the model of the environment can be very complicated due to the discontinuities caused by contacts.

Value-based RL can be challenging for continuous and high-dimensional action spaces, since we need to solve for the optimal action to get the policy. This challenge can be mitigated by restricting the parametrization of the Q-function.

Model-based RL relies on a good model of the environment, since a bad model can lead to compounding errors.

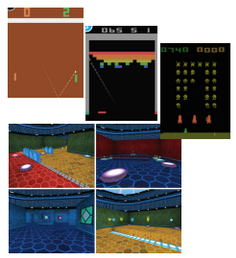
What is Deep Reinforcement Learning?



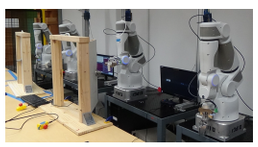
Deep neural networks can process complex sensory input and also compute really complex functions. These deep neural networks can be used for the policy or the value function in model-free RL, or for the models in model-based RL.

Examples of Deep Reinforcement Learning

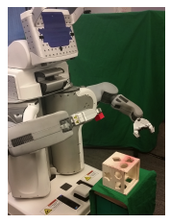
Silver et al, 2014
(DPG)
Lillicrap et al, 2015
(DDPG)



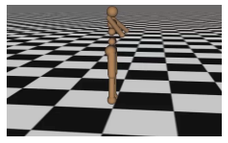
Mnih et al, 2015 (DQN)
Mnih et al, 2016 (A3C)



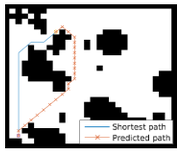
Gu*, Holly*, et al, 2016



Levine*, Finn*,
et al, 2016
(GPS)



Schulman et al, 2016
(TRPO + GAE)



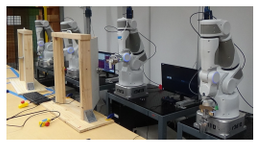
Tamar et al, 2016
(VIN)

Sadeghi et al, 2017 $(CAD)^2RL$

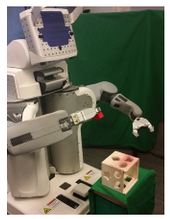
Previous work have used policy optimization for locomotion tasks in simulation, and value-based RL to learn how to play in Atari games.

Deep RL has also been used with robots in the real world. In order to make the learning feasible, previous work combine model-based RL with policy optimization or value-based RL. Or, they train in simulation, and then transfer the policy to the real-world system.

Deep Reinforcement Learning for Robotics



Gu*, Holly*, et al, 2016



Levine*, Finn*, et al, 2016 (GPS)

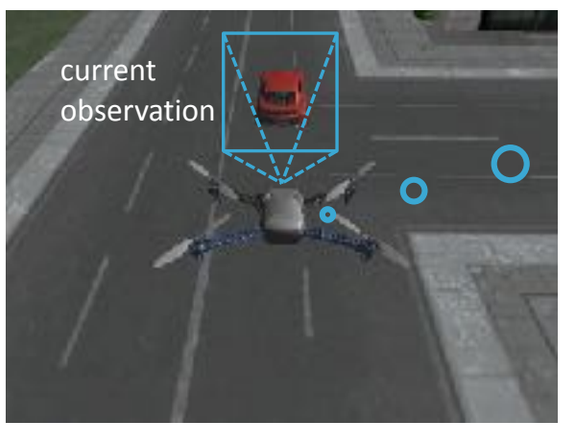
Sadeghi et al, 2017 $(CAD)^2RL$

So, can we just train a deep neural network to learn an arbitrary robotic task without having any prior knowledge of the structure of the task?
We can but… it would require a lot of samples.
If we have prior knowledge of the structure of the problem that we are trying to solve, we can learn with less samples.
In this talk, I'll show an example application to illustrate how we can incorporate structure of a task for deep RL.

Outline

- Introduction
- Reinforcement learning and deep reinforcement learning
- **Visual servoing**
- Learn visual servoing with reinforcement learning
  - Policy optimization
  - Combine value and model based RL
    - Learn visual feature dynamics
    - Learn servoing policy with fitted Q-iteration
- Comparison to prior methods
- Conclusion

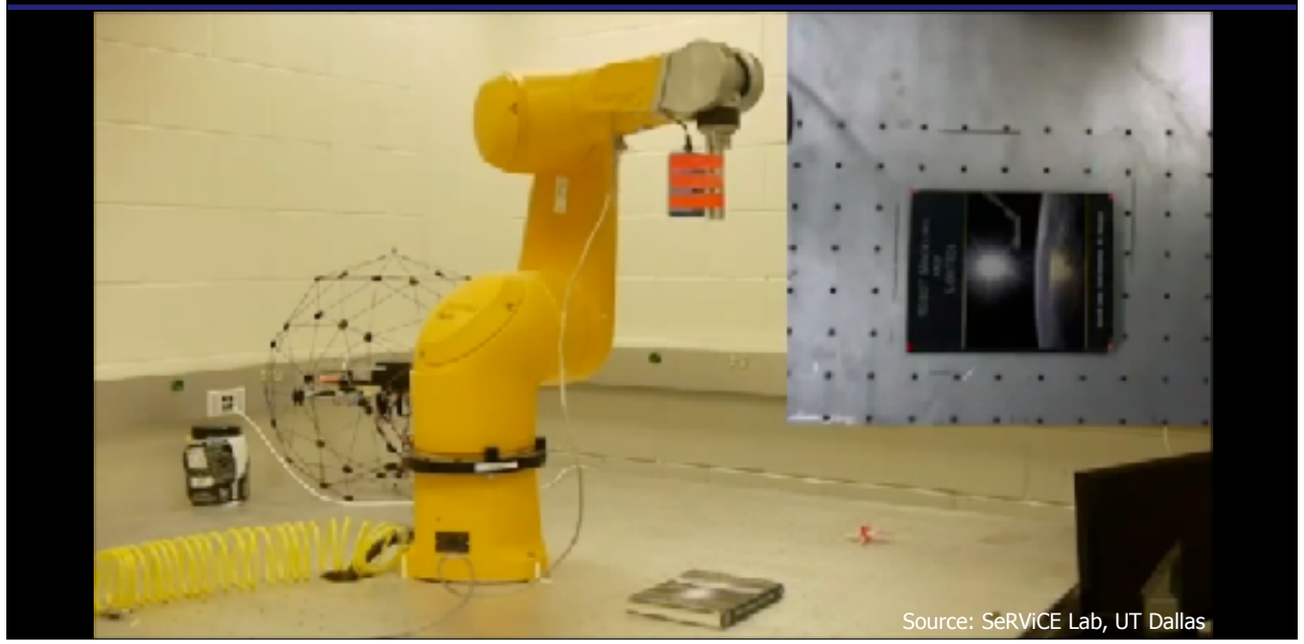
Visual Servoing



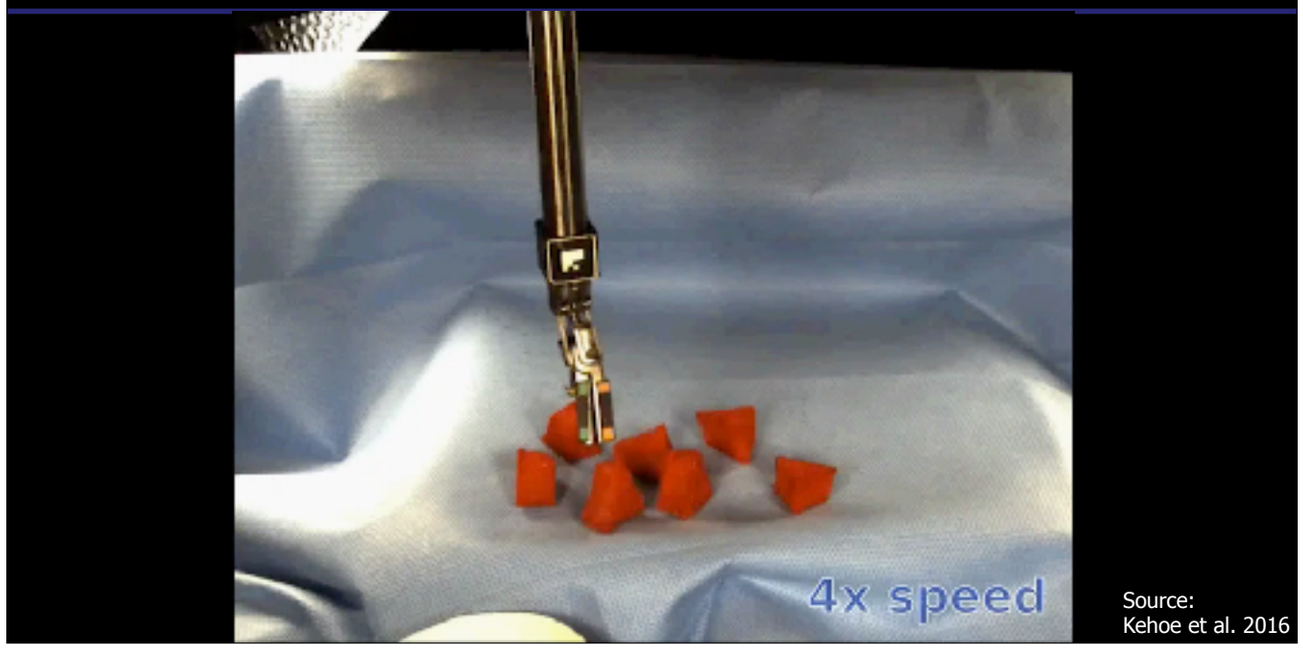
goal
observation

[Explain the setup]
In the literature, this is an example of a classical problem in robotics, called visual servoing.
Visual servoing involves choosing actions that move a robot in response to camera observations in order to reach a goal configuration.

Examples of Visual Servoing: Manipulation



Examples of Visual Servoing: Surgical Tasks



Source:
Kehoe et al. 2016

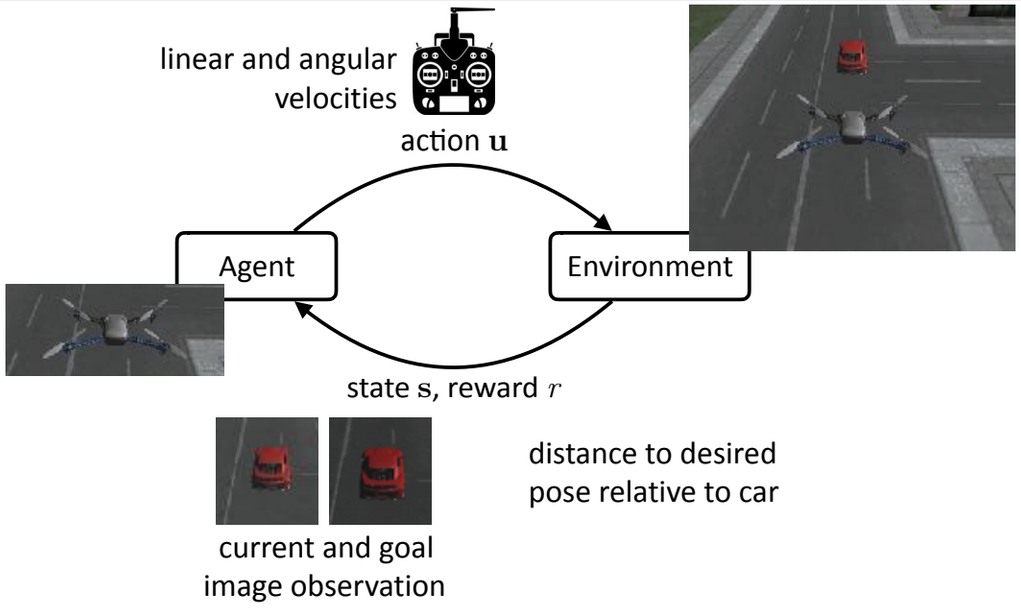
Examples of Visual Servoing: Space Docking

Source: NASA

Outline

- Introduction
- Reinforcement learning and deep reinforcement learning
- Visual servoing
- **Learn visual servoing with reinforcement learning**
  - Policy optimization
  - Combine value and model based RL
    - Learn visual feature dynamics
    - Learn servoing policy with fitted Q-iteration
- Comparison to prior methods
- Conclusion

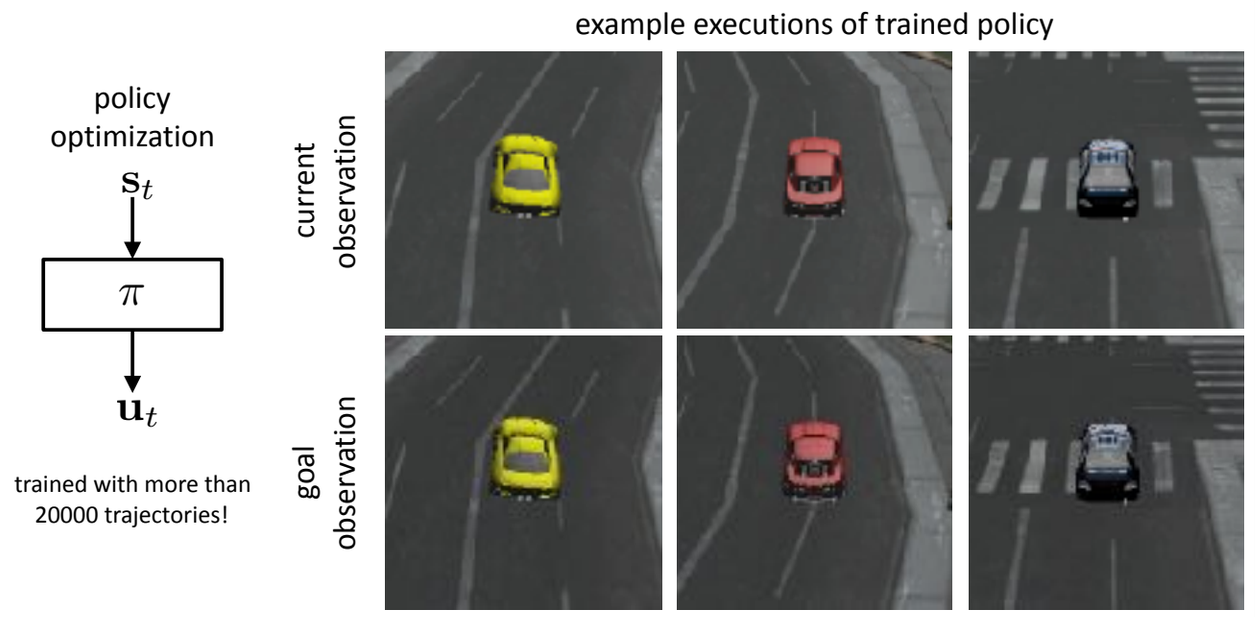
Learning Visual Servoing with Reinforcement Learning



Outline

- Introduction
- Reinforcement learning and deep reinforcement learning
- Visual servoing
- Learn visual servoing with reinforcement learning
  - **Policy optimization**
  - Combine value and model based RL
    - Learn visual feature dynamics
    - Learn servoing policy with fitted Q-iteration
- Comparison to prior methods
- Conclusion

Learning Visual Servoing with Policy Optimization



Outline

- Introduction
- Reinforcement learning and deep reinforcement learning
- Visual servoing
- Learn visual servoing with reinforcement learning
  - Policy optimization
  - **Combine value and model based RL**
    - Learn visual feature dynamics
    - Learn servoing policy with fitted Q-iteration
- Comparison to prior methods
- Conclusion

Combining Value and Model Based Reinforcement Learning

State-action value based RL: $\pi(\mathbf{s}_t) = \arg\max_{\mathbf{u}} Q(\mathbf{s}_t, \mathbf{u})$

Combining Value and Model Based Reinforcement Learning

State-action value based RL:

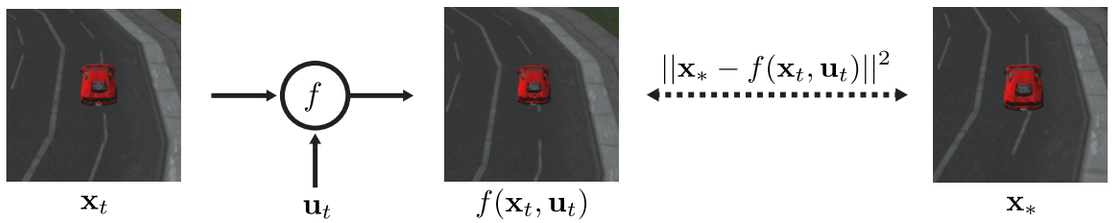
$$\pi(\mathbf{s}_t) = \arg\min_{\mathbf{u}} -Q(\mathbf{s}_t, \mathbf{u})$$

Visual servoing:

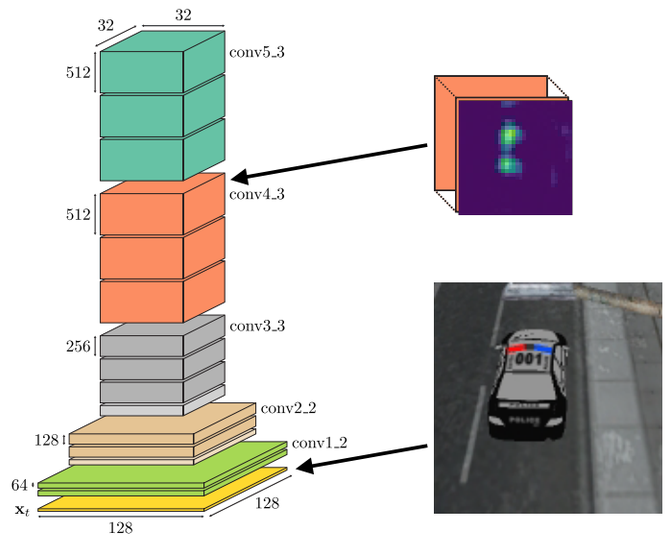
$$\pi(\mathbf{s}_t) = \arg\min_{\mathbf{u}} \underbrace{||\mathbf{x}_* - f(\mathbf{x}_t, \mathbf{u}_t)||^2}_{-Q(\mathbf{s}_t, \mathbf{u})}$$

dynamics function

Servoing with Visual Dynamics Model

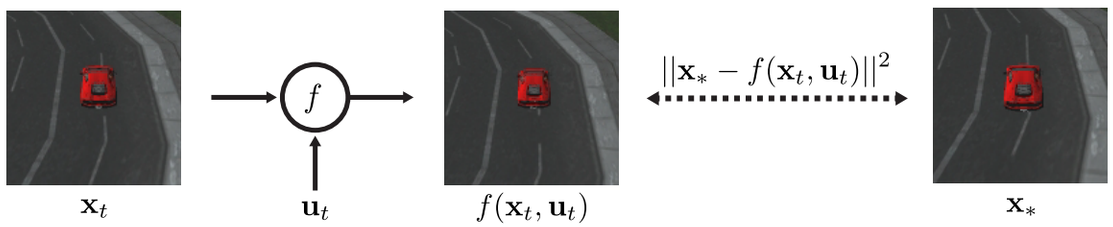


Features from Dilated VGG-16 Convolutional Neural Network



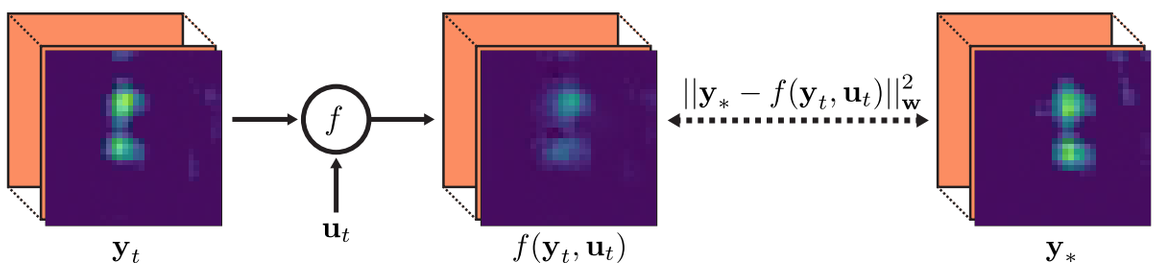
K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In ICLR, 2015.
F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In ICLR, 2016.

Servoing with Visual Dynamics Model



Servoing with Visual Dynamics Model

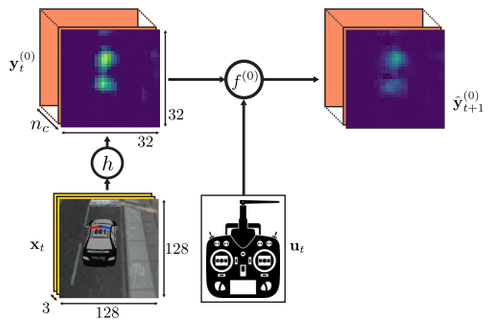
$$\pi(\mathbf{x}_t, \mathbf{x}_*) = \arg\min_{\mathbf{u}} \underbrace{||\mathbf{y}_* - f(\mathbf{y}_t, \mathbf{u}_t)||^2_{\mathbf{w}}}_{-Q_{\mathbf{w}}(\mathbf{s}_t, \mathbf{u})}$$



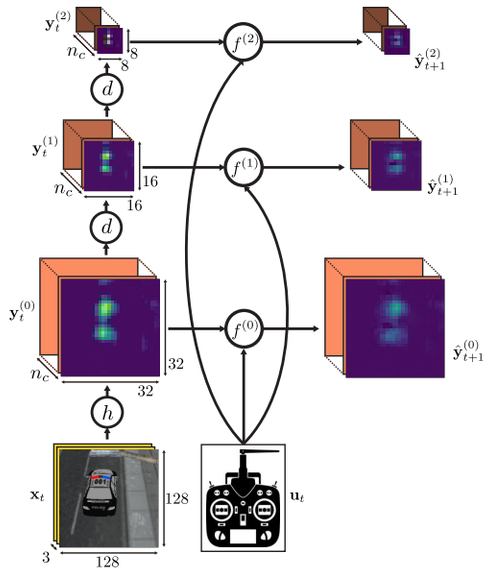
Outline

- Introduction
- Reinforcement learning and deep reinforcement learning
- Visual servoing
- Learn visual servoing with reinforcement learning
  - Policy optimization
  - Combine value and model based RL
    - **Learn visual feature dynamics**
    - Learn servoing policy with fitted Q-iteration
- Comparison to prior methods
- Conclusion

Feature Dynamics: Multiscale Bilinear Model



Feature Dynamics: Multiscale Bilinear Model



Outline

- Introduction
- Reinforcement learning and deep reinforcement learning
- Visual servoing
- Learn visual servoing with reinforcement learning
  - Policy optimization
  - Combine value and model based RL
    - Learn visual feature dynamics
    - **Learn servoing policy with fitted Q-iteration**
- Comparison to prior methods
- Conclusion

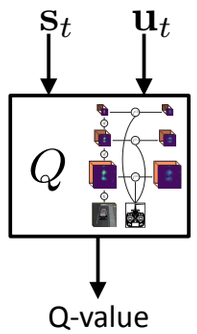
Learning Model Based Policy with Fitted Q-Iteration

$$\pi(\mathbf{s}_t) = \arg\min_{\mathbf{u}} \underbrace{||\mathbf{y}_* - f(\mathbf{y}_t, \mathbf{u}_t)||^2_{\mathbf{w}}}_{-Q_{\mathbf{w}}(\mathbf{s}_t, \mathbf{u})}$$

Not all features are equally relevant; relevance is task-dependent
To capture long-term dependencies, use Q-values
Learn Q-function approximator with fitted Q-iteration

Learning Visual Servoing with Deep Feature Dynamics and FQI

value based +
visual dynamics model



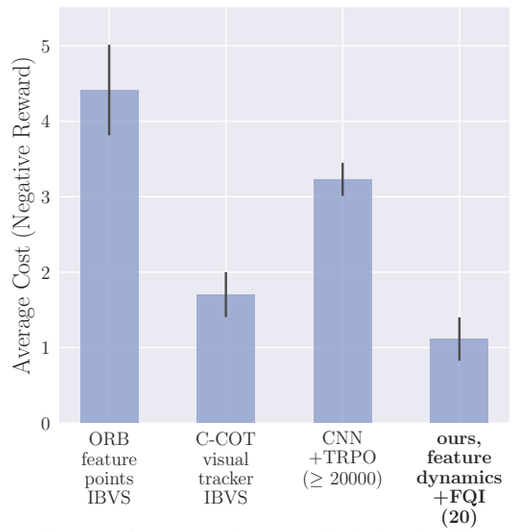
trained with only
20 trajectories!

example executions of trained policy

Outline

- Introduction
- Reinforcement learning and deep reinforcement learning
- Visual servoing
- Learn visual servoing with reinforcement learning
  - Policy optimization
  - Combine value and model based RL
    - Learn visual feature dynamics
    - Learn servoing policy with fitted Q-iteration
- **Comparison to prior methods**
- Conclusion

Comparison to Prior Methods



Conclusion

- Deep reinforcement learning allows us to learn complex robot policies that can process complex visual inputs
- Combine value based and model based for better sample complexity
- Visual servoing
  - Learn visual feature dynamics
  - Learn Q-values with fitted Q-iteration

In this talk, I'll showed an example application to illustrate how we can incorporate structure of a task for deep RL, in the context of visual servoing.

I described an approach that combines learned visual features with learning predictive dynamics models and reinforcement learning to learn visual servoing mechanisms.

Our experiments demonstrate that standard deep features, in our case taken from a model trained for object classification, can be used together with an action-conditional predictive model to learn an effective visual servo that is robust to visual variation, changes in viewing angle and appearance, and occlusions.

For control we propose to learn Q-values, building on fitted Q-iteration, which at execution time allows for one-step lookahead calculations that optimize long term objectives.

Our method can learn an effective visual servo on a complex synthetic car following benchmark using just 20 training trajectory samples for reinforcement learning. We demonstrate substantial improvement over a conventional servoing approach, and we show an improvement in sample-efficiency of more than two orders of magnitude over standard model-free deep reinforcement learning algorithms.

Thank You

Acknowledgements

NSF NVIDIA ARL DeepDrive

Resources

Paper: arxiv.org/abs/1703.11000
Code: github.com/alexlee-gk/visual_dynamics
Servoing benchmark code: github.com/alexlee-gk/citysim3d
More videos: rll.berkeley.edu/visual_servoing