End-to-End Training of Deep Visuomotor Policies

Sergey Levine*, Chelsea Finn*, Trevor Darrell, Pieter Abbeel

Presenter: Ankur Handa
What’s the goal here?

● Making a robot learn to do mundane real-world tasks e.g.
  ○ Screwing on a bottle cap
  ○ inserting a block into shape sorting cube
  ○ hanging a coat hanger on a clothes rack
  ○ simple grasping

● Mapping directly from RGB images to motor torques.
  ○ Similar in spirit to Visual Servoing
A demo video
How is it done traditionally?

- A vision module detects some key feature locations in the image.
- A PD controller is tuned heuristically to map from the visual input to motor torques.
- Detection of key locations and the driving of motors share only one way communication.
Limitations

- Detected features are not task dependent.
- The performance of vision system is never improved over the whole experiment.
- Different modules are chained together. Every module is heuristically tuned and therefore there is no end-to-end mapping from input to output space.
Solution proposed in this paper

- End-to-end mapping from visual input (RGB image) to motor torques.
  - No camera calibration needed.
- Offers flexibility in how to choose the visual signal.
- Treated as a policy search using reinforcement learning.
Mathematical introduction

$u_t$ are motor torques.

$x_t$ known robot configuration as well as target position for object placement task.

$o_t$ are observations e.g. the camera image as well configuration of the robot.

$\ell(x_t, u_t)$ cost function to be minimised over a trajectory.

$\pi_\theta(u_t | o_t)$ specifies a distribution from images to torques given some parameters theta.
Mathematical introduction

\( u_t \) are motor torques.

\( x_t \) known robot configuration as well as target position for object placement task.

\( o_t \) are observations e.g. the camera image as well configuration of the robot.

\( \ell(x_t, u_t) \) cost function to be minimised over a trajectory.

\( \pi_\theta(u_t | o_t) \) specifies a distribution from images to torques given some parameters theta.

This is done using CNNs in this work.
Mathematical introduction

\[ \theta = \arg \min_{\theta} J(\theta) \]

\[ J(\theta) = E_{\pi_\theta} \left[ \sum_{t=1}^{T} c(x_t, u_t) \right] \]

\( \pi_\theta(u_t|x_t) - \text{control policy} \)
The task now is do alternating trajectory optimisation and policy search.
Note about Trajectories

- Use a distribution on trajectories and not just trajectories in the optimisation.
- If the policy returns a position that it not in the trajectory it will still be able to recover if we have samples obtained from the distribution of trajectories.
**Policy learning**

- Use trajectory-centric reinforcement learning method with unknown dynamics to sample trajectories.
- Use these trajectories to do supervised learning with CNNs to map from RGB to motor torques.
- Use the policy within the trajectory sampling and iterate.
Note: Bound the trajectory distribution
Some results
Trajectory optimisation to policy search
Feedback loop
Overall Algorithm
Test time assumptions

- We will not have access to the position of the target object; this is inferred from the image. This is what they call the unobserved variables.
- We will not have access to the dynamics model; this is also inferred.
- Same network will be used for variety of tasks.
Cost

Iterate until the distribution of actions returned by the policy matches the trajectory distribution.

\[
\min_{p, \pi_\theta} E_p[\ell(\tau)] \quad \text{s.t.} \quad p(u_t|x_t) = \pi_\theta(u_t|x_t) \quad \forall x_t, u_t, t,
\]