Control of a Quadrotor with Reinforcement Learning

Jemin Hwangbo, Inkyu Sa, Roland Siegwart, and Marco Hutter

Robotic Systems Lab, ETH Zurich

Presented by Nicole McNabb

University of Waterloo

June 27, 2018

Overview

1. Introduction
2. The Method
3. Empirical Results
4. Summary and Future Work

What is a quadrotor?

Figure: Quadrotor [1]

What is a quadrotor?

Figure: Quadrotor [1]

High-level goal:

Train the quadrotor to perform tasks with varying initializations

A policy optimization problem.

Related Approaches

Deep Deterministic Policy Gradient (DDPG)

- Actor-critic architecture
- Off-policy, model-free
- Deterministic

- Insufficient exploration
- Very slow (if any) convergence

Trust Region Policy Optimization (TRPO)

- Actor-critic architecture
- Off-policy, model-free
- Stochastic

- Computationally intensive
- Slow, unreliable convergence

A New Approach

Goal:
A deterministic model with

- Fast and stable convergence
- Model-free training
- Extensive exploration

Solution:
A method combining the actor-critic architecture with an on-policy deterministic policy gradient algorithm and a new exploration strategy.

Setup

Continuous State-Action Space

State Space

18-D states, model:

- Orientation (or rotation)
- Position
- Linear velocity of system
- Angular velocity of system

Action Space

4-D actions, dictate rotor thrust for each rotor

Exploration

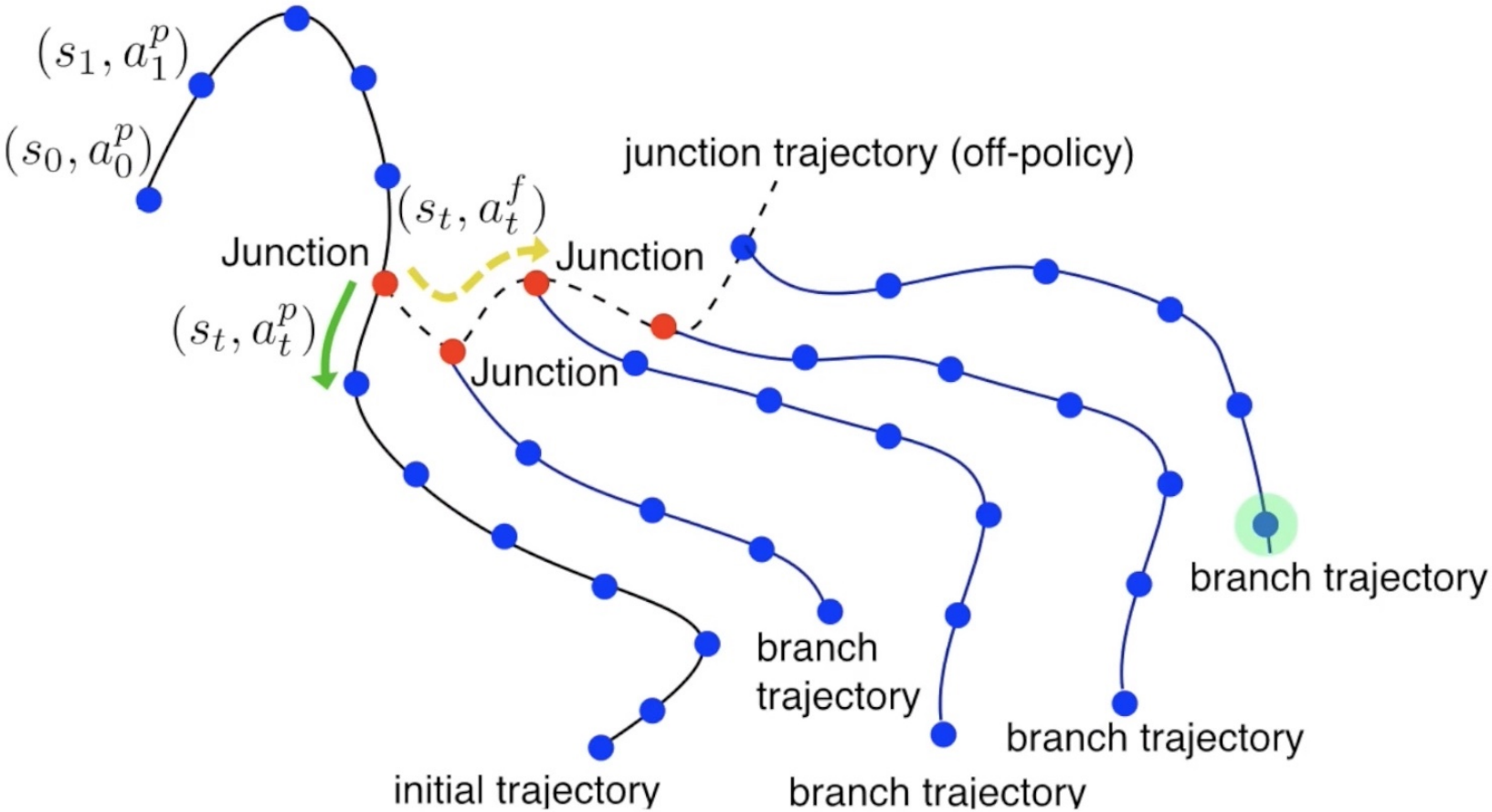

Figure: Exploration Strategy [2]

Network Training

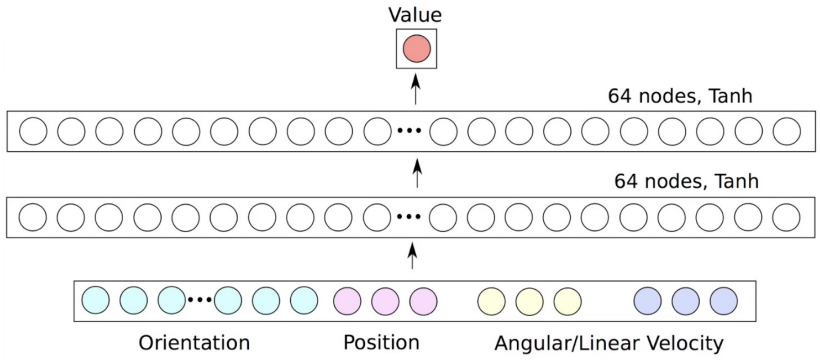

Figure: Value Network [2]

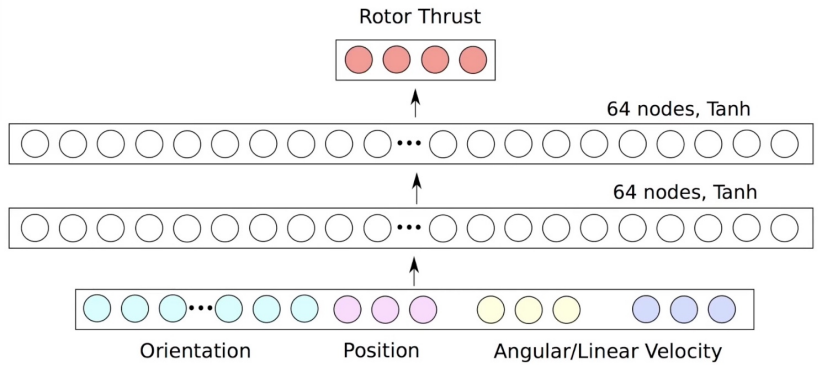

Figure: Policy Network [2]

Value function training:
Approximate with Monte-Carlo samples obtained from current trajectory

Policy optimization:
Same idea as TRPO, replacing KL-divergence with Mahalanobis metric

Learning Algorithm

Algorithm 1 Policy optimization

1: Input: Initial value function approximation, initial policy
2: **for** j = 1,2,. . . **do**
3: Perform exploration, take action
4: Compute MC estimates from current trajectory
5: Do approximate value function update
6: Do policy gradient update
7: **end for**

Empirical Results

- Training done in simulation
- Testing on two main tasks done on a real quadrotor

Summary

Primary contributions:

- A new deterministic, model-free neural network policy for training a quadrotor
- Stable and reliable performance on hard tasks, even under harsh initial conditions

Future Research

- Also compare model against PPO
- Introducing more accurate model of the system into simulation
- Train an RNN to adapt to model errors automatically

References


https://www.seeedstudio.com/Crazyflie-2.0-p-2103.html


Jemin Hwangbo, Inkyu Sa, Roland Siegwart, and Marco Hutter
Control of a Quadrotor with Reinforcement Learning
IEEE Robotics and Automation Letters, June 2017.

Questions?