Deployable Robotics (Part I)

Russ Tedrake
If a Robotic Hand Solves a Rubik’s Cube, Does It Prove Something?

A five-fingered feat could show important progress in A.I. research. It is also a stunt.
“For the Rubik’s cube task, we use $8 \times 8 = 64$ NVIDIA V100 GPUs and $8 \times 115 = 920$ worker machines with 32 CPU cores each. … The cumulative amount of experience ... is roughly 13 thousand years.”
Table 6: Performance of different policies on the Rubik’s cube for a fixed fair scramble goal sequence. We evaluate each policy on the real robot (N=10 trials) and report the mean ± standard error and median number of successes (meaning the total number of successful rotations and flips). We also report two success rates for applying half of a fair scramble (“half”) and the other one for fully applying it (“full”). For ADR policies, we report the entropy in nats per dimension (npd). For “Manual DR”, we obtain an upper bound on its ADR entropy by running ADR with the policy fixed and report the entropy once the distribution stops changing (marked with an “*”).

<table>
<thead>
<tr>
<th>Policy</th>
<th>Sensing</th>
<th>ADR Entropy</th>
<th>Successes (Real)</th>
<th>Success Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pose</td>
<td>Face Angles</td>
<td></td>
<td>Mean</td>
</tr>
<tr>
<td>Manual DR</td>
<td>Vision</td>
<td>Giiker</td>
<td>−0.569* npd</td>
<td>1.8 ± 0.4</td>
</tr>
<tr>
<td>ADR</td>
<td>Vision</td>
<td>Giiker</td>
<td>−0.084 npd</td>
<td>3.8 ± 1.0</td>
</tr>
<tr>
<td>ADR (XL)</td>
<td>Vision</td>
<td>Giiker</td>
<td>0.467 npd</td>
<td>17.8 ± 4.2</td>
</tr>
<tr>
<td>ADR (XXL)</td>
<td>Vision</td>
<td>Giiker</td>
<td>0.479 npd</td>
<td>26.8 ± 4.9</td>
</tr>
<tr>
<td>ADR (XXL)</td>
<td>Vision</td>
<td>Vision</td>
<td>0.479 npd</td>
<td>12.8 ± 3.4</td>
</tr>
</tbody>
</table>

Pablo’s lectures

Introduction to (robust) control and Lyapunov; we’ll do a bit more of that here...

Billion dollar question:

What will be the *epistemology* of deployable ML?
Challenge #1: System Complexity
Deploying Autonomous/Learning Systems

The complexity of perception breaks our existing tools…

- Sensors include cameras ⇒ sensor model is a photo-realistic rendering engine
- Perception components (especially) include deep neural networks; but verifying planning algorithms also nontrivial.
- Plant model has to capture distributions over natural scenes (numbers/types of objects, material properties, lighting conditions)
Challenge #2: Distributional Robustness and Black Swans
What do you see in this Picture?

Courtesy: John Leonard

Coolidge Corner, Brookline MA
My lesson in robustness

In a garage at MIT just days before the competition…

Now passionate to understand how to get robustness from these complex systems.
Challenge #3: High expectations

(how safe do we have to be to deploy?)
Analysis (via a very simple coin flipping model): To estimate to within 20% of assumed rate (1.09/100 million), with 95% confidence, requires ~ 8.8 billion miles.
Releases of technology in the airline industry

Goal: Toolkit for reasoning about about uncertainty in closed-loop systems
Uncertainty representations

In controls (polytopic/ellipsoidal, etc)

Developing autonomous systems in the real world.

Domain randomization in reinforcement learning

Abbeel et al.

IEEE Spectrum
Common Lyapunov Functions and Invariant Sets

On the board.

Relevant course notes (from 6.832) are [here](#).
Region of attraction for the (time-reversed) van der Pol oscillator
Verification with Inertia Variation of 10 Percent

- Nominal Verified Basin
- Uncertain Verified Basin
Can we make a control system for a fixed-wing airplane to land on a perch like a bird?
Nonlinear (post-stall) dynamics described well by polynomial diff eq.

\[ \dot{x} = f(x, u) \]
Plan trajectories with sequential quadratic programming (SQP)

Invariant sets as a **sums-of-squares (SOS)** optimization
Wind disturbances: colored “noise” drawn from ellipsoidal uncertainty set.

Robust control via bilinear SOS alternating.
ONR MURI: Provable-safe high-speed flight through forests
Some final thoughts (on the board)