Risk-averse Batch Active Inverse Reward Design

Panagiotis Liampas

Abstract

Autonomous agents optimize the reward function we give them. What they don’t know is how hard it is for us to design a reward function that actually captures what we want. When designing the reward, we might think of some specific training scenarios, and make sure that the reward will lead to the right behavior in those scenarios. Inevitably, agents encounter new scenarios (e.g., new types of terrain) where optimizing that same reward may lead to undesired behavior. Our insight is that reward functions are merely observations about what the designer actually wants, and that they should be interpreted in the context in which they were designed. We introduce inverse reward design (IRD) as the problem of inferring the true objective based on the designed reward and the training MDP. We introduce approximate methods for solving IRD problems, and use their solution to plan risk-averse behavior in test MDPs. Empirical results suggest that this approach can help alleviate negative side effects of misspecified reward functions and mitigate reward hacking.

# Introduction

## Heading 2

Robots are becoming more capable of optimizing their reward functions. But along with that comes the burden of making sure we specify these reward functions correctly. Unfortunately, this is a notoriously difficult task. Consider the example from Figure 1. Alice, an AI engineer, wants to build a robot, we’ll call it Rob, for mobile navigation. She wants it to reliably navigate to a target location and expects it to primarily encounter grass lawns and dirt pathways. She trains a perception system to identify each of these terrain types and then uses this to define a reward function that incentivizes moving towards the target quickly, avoiding grass where possible. When Rob is deployed into the world, it encounters a novel terrain type; for dramatic effect, we’ll suppose that it is lava. The terrain prediction goes haywire on this out-of-distribution input and generates a meaningless classification which, in turn, produces an arbitrary reward evaluation. As a result, Rob might then drive to its demise. This failure occurs because the reward function Alice specified implicitly through the terrain predictors, which ends up outputting arbitrary values for lava, is di…

## More heading 2

Blah blah blah…

### Heading 3.

Normal text after it

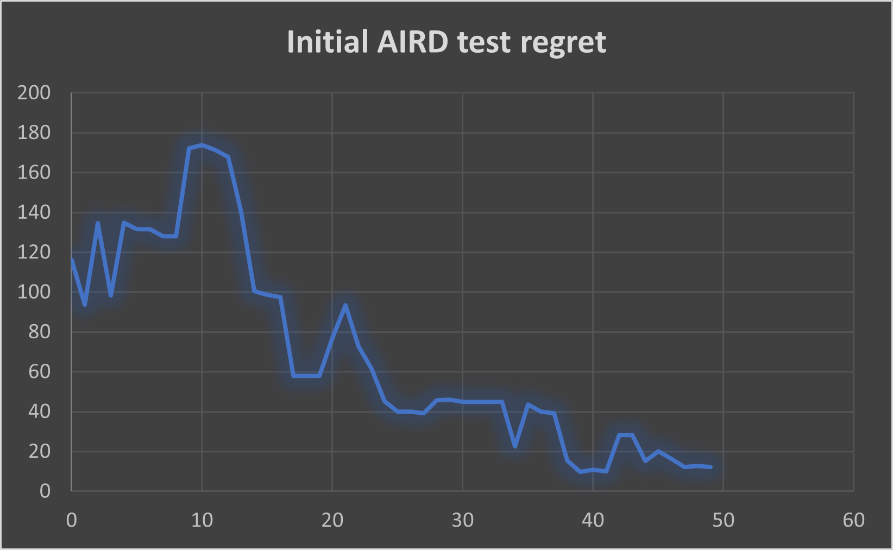


Figure 1: Sample figure description

Sample inline math: and text after it

Sample equation:

(press #(whatever number) after the equation)

Sample figure with multiple pictures

|  |  |  |
| --- | --- | --- |
| (a) S0, S1 and S2 trajectories, given the 1st randomly selected set of test parameters and the same start state, calculated using the time-1 map model, the derivative model, and the Runge-Kutta method. | (b) S0, S1 and S2 trajectories, given the 2nd randomly selected set of test parameters. | (c) S0, S1 and S2 trajectories, given the 3rd randomly selected set of test parameters. |

Figure 2: Sample multi-image figure

# How to cite

Example **paraphrased** sentence from another paper (Amin et al., 2017).

as seen in (Amin et al., 2017)

In the same document: as we see from Figure 1, or Equation (or inequality (1) or whatever), add cross references. (but I cannot do that for Zotero references??)

References

Amin, K., Jiang, N., & Singh, S. (2017). *Repeated Inverse Reinforcement Learning* (arXiv:1705.05427). arXiv. https://doi.org/10.48550/arXiv.1705.05427