How to Specify Aligned Reinforcement Learning Problems

Brad Knox



Outline

- 1. Aligned reward functions
- 2. Finding misalignment in a reward function
- 3. Misalignment by shaping
- 4. Misalignment by trial-and-error design
- 5. Misalignment by discounting
- 6. Designing aligned reward functions
- 7. What should we work towards?

Aligned reward functions



Gridworld



States: cell location

Actions: move in each cardinal direction





State: board configuration

Action: a legal move





Observation: joint positions, joint velocities, and a camera image of its task area

Action: acceleration on each joint



Reinforcement learning



Reinforcement learning

Gridworld



State: cell location

Action: move in each cardinal direction

Reward: -1 per time step / action



Reinforcement learning



State: board configuration

Action: a legal move

Reward: 1 upon winning, 0 otherwise (assume no stalemates)



REWARD AND RETURN

T $G(\tau) = \sum \gamma^t R(s_t, a_t, s_{t+1})$ t = 0reward return

REWARD AND RETURN

<u>Field</u>

reinf. learning motion planning control theory evolutionary algs. utility theory optimization

$G(\tau) = \sum_{t=0}^{T} f$	$\gamma^t R(s_t, a_t, s_{t+1})$
return	Tewara
-1 × cost	-1 × cost
-1 × cost	-1 × cost
fitness	-
utility	-
objective function*	-
performance metric	-
score	-

* "objective function" more precisely refers to the expectation of $G(\tau)$



EXPECTED RETURN

$J(\pi) = E_{\pi}[G(\tau)]$

(Implicit in the expectation is the distribution over start states and state transitions.)



EXPECTED RETURN

$$J(\pi) = E_{\pi}[G(\tau)]$$

(Implicit in the expectation is the distribution over start states and state transitions.)

A more precise characterization of RL: attempt to find a behavior policy π that maximizes expected return.

Rewards vs goals



Main alternatives to RL problems

- Learning from demonstrations (imitation learning)
- Learning from preferences (RLHF, RLAIF, DPO, CPL, etc.)

Learning an intermediate reward function and doing RL on it is not the only way to these methods.

BACKGROUND ON REWARD

RL oversimplified: a set of problems and corresponding algorithmic solutions, in which *experience in a task is used to improve an agent's behavior such that it gets more reward*.

More specifically, most RL problems focus on increasing the *expectation* of $G(\mathbf{T})$, the utility of a trajectory:

$$G(\tau) = \sum_{t=1}^{(T-1)} R(s_t, a_t, s_{t+1})$$

(Assumes undiscounted/episodic setting and an unstated distribution over starting states)

An aligned reward function

- A **reward function** creates a preference ordering over possible trajectories (by $G(\tau)$) and probability distributions over trajectories.
- These trajectories can simplified to only the outcomes that matter (e.g., winning/losing or time until reaching a goal.)
- We assume **humans** also have such an ordering.

A perfectly aligned reward function creates an ordering over outcome distributions that matches that of the human stakeholder.

An aligned reward function



A perfectly aligned reward function creates an ordering over outcome distributions that matches that of the human stakeholder.

An aligned reward function



A perfectly aligned reward function creates an ordering over outcome distributions that matches that of the human stakeholder.

An aligned reward function?

<u>Human stakeholder</u>

Expected return



An aligned reward function?

<u>Human stakeholder</u>

Expected return



An aligned reward function?

<u>Human stakeholder</u>

Expected return



Gridworld



State: cell location

Action: move in a cardinal direction

The agent's purpose is to reach the goal in the minimum mean time from any state.

Design a reward function by designating what the reward should be for each action from each cell (state).

There is no discounting and the goal state is terminal (or transitions to absorbing state, if you prefer).

Gridworld



An aligned reward function

-1 reward until goal

The return is the time to goal (* -1). The **expected return** is the mean time to goal (* -1). So the task is to minimize the mean time to goal.

Gridworld



A misaligned reward function

1 reward for an action towards the goal 0 reward otherwise

The return is the number of goal-approaching actions. The **expected return** is the mean number of goal-approaching actions. So the task is to maximize the mean number of goal-approaching actions.

Gridworld



A misaligned reward function

1 reward for an action towards the goal 0 reward otherwise

The return is the number of goal-approaching actions. The **expected return** is the mean number of goal-approaching actions. So the task is to maximize the mean number of goal-approaching actions.

Will an agent that's maximizing its expected return terminate?

Gridworld



A misaligned reward function

0 reward for an action towards the goal -1 reward otherwise

The return is the number of suboptimal actions (* -1). The **expected return** is the mean number of suboptimal actions (* -1). So the task is to minimize the mean number of suboptimal actions.

Gridworld



A misaligned reward function

0 reward for an action towards the goal -1 reward otherwise

The return is the number of suboptimal actions (* -1). The **expected return** is the mean number of suboptimal actions (* -1). So the task is to minimize the mean number of suboptimal actions.

What if the goal is moved?

What if you misunderstood which actions go to the goal?

AI safety terminology

Precise definitions couldn't be found, so my versions:

outer alignment - the problem given to an AI optimizer to solve is aligned

• regardless of whether the resultant solution is aligned in practice

outcome-based learning - optimizing decisions based on future rewards or goals

Is RL unfixably unsafe?



POLICY FORUM

ARTIFICIAL INTELLIGENCE

Regulating advanced artificial agents

Governance frameworks should address the prospect of AI systems that cannot be safely tested

By Michael K. Cohen^{1,2}, Noam Kolt^{3,4} Yoshua Bengio^{5,6}, Gillian K. Hadfield^{2,3,4,7}, Stuart Russell^{1,2}

echnical experts and policy-makers have increasingly emphasized the need to address extinction rick from

under control is also reflected in President Biden's 2023 executive order that introbe controlle duces reporting requirements for AI that achieve com could "eva[de] human control or oversight rewards ap through means of deception or obfuscacapable RL tion" (3). Building on these efforts, now is rewards, wh the time for governments to develop reguto secure me

So long

"Giving an advanced AI system the objective to maximize its reward (LTPAs)..." leads to concerns that include reward tampering, removing humans as obstacles to reward, and power seeking.

"both safety and validity cannot be ensured when testing sufficiently capable LTPAs"

"Developers should not be permitted to build sufficiently capable LTPAs, and the resources required to build them should be subject to stringent controls."

Is RL unfixably unsafe?



POLICY FORUM

ARTIFICIAL INTELLIGENCE

Regulating advanced artificial agents

Governance frameworks should address the prospect of AI systems that cannot be safely tested

By Michael K. Cohen^{1,2}, Noam Kolt^{3,4}, Yoshua Bengio^{5,6}, Gillian K. Hadfield^{2,3,4,7}, Stuart Russell^{1,2}

echnical experts and policy-makers have increasingly emphasized the need to address extinction risk from under control is also reflected in President Biden's 2023 executive order that introduces reporting requirements for AI that could "eva[de] human control or oversight through means of deception or obfuscation" (3). Building on these efforts, now is that time for governments to devalor requi-

t So long be controlle t achieve com t rewards ap - capable RL s rewards, wh "If dangerously capable LTPAs are at some point permitted to be developed, rigorous technical and regulatory work would need to be done first..."

This talk covers such work.

Is RL impactful?



Observation: prompt + any previous text

Action: n-token response

Reward: ???



Is RL impactful?



Observation: prompt + any previous text

Action: n-token response

Reward: ???

Here, demonstrations and preferences are used. The "reward function" from RLHF is not a reward function in they way we normally use the concept.



Is RL impactful?

RL has often had a stigma of **not yet working well for important problems.** It has had some large successes though. A few:

- Games: Go, Chess, Poker, and Starcraft
- Data center cooling
- RL as search (LLM fine-tuning and AlphaFold)

Learning from long-horizon reward is **harder to wield than learning from demonstrations and preferences.**

Nonetheless, a well-formulated RL problem has the potential to lead to performance far beyond what humans can demonstrate or identify through preferences.

The set of optimal policies is invariant to rescaling of the reward function.
The set of optimal policies is invariant to shifts of the reward function if...

from each state, all possible trajectories have the same length

Includes continuing and finite horizon tasks.

Does not include typical episodic tasks, such as those with goal or failure states.

A change in perspective

Reward from the perspective of an RL algorithm \leftarrow the familiar perspective

An agent conducting policy improvement continually searches for policies that get higher mean return.

We start zoomed in and zoom out from there:

- Agent is a pursuer of reward.
- Agent estimates expected return from the reward it has experienced.
- Agent identifies actions (more precisely, changes in policy) that will increase estimated expected return.

A change in perspective

Reward from the perspective of an RL algorithm \leftarrow the familiar perspective

An agent conducting policy improvement continually searches for policies that get higher mean return.

We start zoomed in and zoom out from there:

- Agent is a pursuer of reward.
- Agent estimates expected return from the reward it has experienced.
- Agent identifies actions (more precisely, changes in policy) that will increase estimated expected return.

A change in perspective

A reward-centric perspective of policies

The choice of the reward function and discounting create an ordering over policies—given a start-state distribution—via their expected returns and over full trajectories.

Each change in a reward function may rearrange this ordering by changing each policy's expected return.

Learning is not a consideration.

Ordering by expected return:



Consider designing for interpretable return.

Two highly common reward functions have interpretable return.

reward function: -1 reward until goal The return is the time to goal (* -1). The **expected return** is the mean time to goal (* -1). So the task is to minimize the mean time to goal.



reward function: 0 upon losing, 1 upon winning, and 0 otherwise
The return is a binary indicator of winning.
The expected return is the probability of winning.
So the task is to maximize the probability of winning.



CC 2.0 license (source)

Finding Misalignment in a Reward Function

FIND MISMATCHES IN PREFERENCE ORDERINGS (The most powerful method I'm aware of.)

If all human stakeholders agree that trajectory τ_A is preferable to τ_B (i.e., $\tau_A > \tau_B$), then return of $\tau_A >$ return of τ_B should hold.



W. Bradley Knox, Alessandro Allievi, Holger Banzhaf, Felix Schmitt, and <u>Peter Stone</u>. **Reward (Mis)design for Autonomous Driving**. AlJ 2023.

















End-to-End Race Driving with Deep Reinforcement Learning

Maximilian Jaritz^{1,2}, Raoul de Charette¹, Marin Toromanoff², Etienne Perot² and Fawzi Nashashibi

Adstract—We present research using the latest reinforcement learning algorithm for end-to-end driving without any mediated preception (s)decir recognition, score understanding). The newly proposed reward and karning strategies lead (spether 60 latest convergence and more rebest driving using out) RoB sare from a forward facine camera. An As y replace tol with control to-end o percept tirect irect axed Lean work th For car were she dorceme We propose a matterial (trg. (p) constrainty, to a rectin-synchronous learning [13] and building on our predim-inary work [17] to train an end-to-end agent in World Rally Champional 5 (WRC6), a realistic-ser racing game with stochastic be is or cumunations, (ig). In addition to with stochastic be remain close to : image ar control and can methy sonverges some senalization compton wiroum vite modules by for primindions. The indicators the second the development lained h due to the 8 layer CNN to learn the lateral control from a front via mining (e.g. oncoming cars). Section III describes the few related works in end-to-end Section III describes the few related works in end-to-end carriers, using stering range from a raid driver as a ground driving. Section III details car methodology and kenning truth, huuss 726 of truining data, with homographic integrate variables in discussed in III and there fewerth-freque carriers to greater with the section of the puterblacknot on real videos is shown in ∑ 'ben, RTS Tano, 2 ne šenos IR: 7012. Po tano in the star in the star

> Xiaodan Liang^{1,2}, Tairui Wang², I na Yang¹, Eric P. Xing² Petrum Inc.2 . Aut, nous urben di ve navie in with complex multi-agent dy w. The traditional modular pipeline heavily relies on hand-designed rules and the inled Controllable Imitative Reinforcement Learning (CIRL) approach n only vision inputs in a high fidelity car simulator. To alleviate the low explo-ration efficiency for large con-tuous action space that often prohibits the use of lissur AL ceb, un agn train acti space g un ceb specialize adaptive policies and storing-angle reward designs for we propose to specialize adaptive policies and storing-angle reward designs for sive experiments on CARLA driving benchmark demo CIRL substantially outperforms all previous methods in terms of the percentage of successfully completed episodes on a variety of goal-directed driving tasks. We also show its superior generalization capability in unseen environments. To our knowledge, this is the first successful case of the learned driving policy by reinforcement learning in the high-fidelity simulator, which performs better than supervised imitation learning.

CIRL: Controllable Imitative Reinforcement Learning

for Vision-based Self-driving

Keywords: Imitative reinforcement learning, Autonomous driving

1 Introduction

Autonomous urban driving is a long-studied and still under-explored task [12] particularly in the crowded urban environments 3. A desirable system is required to be capable of solving all visual perception tasks (e.g. object and lane localization, drivable paths) and determining long-term driving strategies, referred as "driving policy" Although visual perception tasks have been well studied by resorting to supervised learning on large-scale datasets [45], simplistic driving policies by manually designed rules in the modular pipeline is far from sufficient for handling diverse real-world cases as discussed in 677. Learning a optimal driving policy that mimics human drivers is less explored but key to navigate in complex environments that requires understanding of multi-agent dynamics, prescriptive traffic rule, negotiation skills for taking left

Deep Distributional Reinforcement Learning Based High-Level Driving Policy Determination Kyushik Min¹⁰, Hayoung Kim¹⁰, and Kunsoo Huh¹⁰, Member, IEEE

HER TRANSACTIONS ON INTILLIGENT VEHICLES, VOL. 4, NO. 3, SEPTEMBER 20

brough some of the christ-maintain astronomes more than provides address and extensiones in the other provides address and extensiones in the other sensitive address to stand gets patient of the other sensitive address to stand gets patient of the other sensitive address and the other sensitive address and the other sensitive address the stand test and the other sensitive address the stand test and promote the difference in standard and the other sensitive address and the other sensitive address address the test address and the other sensitive address and the other sensitive address and the standard between the standard sensitive address and the distributional test address and the other sensitive address address address address address address address is the properior adjustments in standard theorem advantance, which is bargeresside by the Christian standards, a during advance address address address address advantance, address bargers address address address address advantance, address bargers address address address address advantance address bargers address address address advantance address bargers address address address advantance address bargers address address address address advantance address	anisate system we als a ACT and LSS. The is complete large thread points of points and points in the grand between the system of the system of the system of the Dorp Enformment Learning (DRE) is a tachenge which in the system of the system of the system of the system orthonic (S) (2) and many other fails. This is that any advance of the system of the system of the system in the interference in learning points affectively because the training of the system of the system of the system of the point of the system of th
L heraceurrow	expected future reward as a distribution.

forcement learning and the raw sensor output so that vehicle O IMPROVE safety and convenience, a pleasty of stud To approximate the second seco raffic problems in terms of safety and efficiency. Unlike driver scients systems which deal with relatively simple individual In end-to-end mannee can be dargerous, even if reinforcement learning algorithm is trained well. There is always a possibi-ity that the vehicle acts unexpectedly, so the controlled well-cle should include the sufery features. Thus, the agent includes collision novidance systems such as Automotores Barnegeesys. Booking (AEB) and lane change warning, which are already youls, the autonomous driving algorithm must consider high publi, the autoexmous driving algorithm must consider high level decisions in complex road environments. The main motivation for this paper is to utilize the already correspondence of the statistical systems for autoexmous driv-ing in a wellable manner. In the limited stratuctions in highway, autoexmous driving can be carried out by combining driver commercialized in many vehicles. In addition, to reinforce th

also a LIDAR sensor are used as inputs. A multi-input network Manuscript received September 15, 2018; proined January 11, 2019; accepted as 35, 2020; Date of publication May 25, 2019; date of earnerst seminor August 2009; (Corresponding andher Examon Bah), and and an example of the seminor and an example of the engineering Hyperbolic and an example of the seminor and engineering and an example of the seminor and and engineering. Hyperbolic engineering and an example of the seminor and an example of the engineering and an example of the seminor and an example of the engineering and an example of the seminor and an example of the engineering and an example of the seminor and an example of the engineering and an example of the seminor and an example of the engineering of the seminor and an example of the se architecture is designed to compress and combine camera inages and LIDAR data using different deep learning technique respectively. The test of the paper is organized as follows: In Section II, the related works on learning based approaches for autonomous driving are summarized. In Section III, the Markov deci-sion process is briefly explained as the basis of reinfeccement nore of the figures in this paper are available online

2379-9858 © 2019 EEE. Personal ans is permitted, but republicationshofser/hotion requires IEEE permission Soc http://www.inte.arthpublications_standard/publications/tights/ladet.html for more information.

A structured internant use invited by University of Tarvas at & active Descriptional or a larvase \$7,0001 at 00.5710 (10) two EEE Values, Bartierium and

APPEARED IN Proc. Robotics: Science & Systems (RSS), 2019

0.000. 20w 10.1109/TIV.2019.2929467

op and LIDAR d activity of deep in referen, collision a undrastiene. In a undrastiene, In a undrastiene in a undrastiene in a undrastiene view effect for autonom introduction. Final highway deriving ML-agents.

Index Terms-D

LeTS-Drive: Driving in a Crowd by Learning from Tree Search

trees of planning and the runtime efficiency of le ce the performance of both. Experimental root a show that LeTS-Drive outperforms either plann

Note that have no dirit is rook with a field with the state of the

TEXAS (H) BOSCH

Dynamic Input for Deep Reinforcement Learning in Autonomous Driving

Maria Huerle^{1,*}, Gabriel Kalweit^{1,*}, Branka Mirchevska², Moritz Werline², Joschka Boedecker^{1,3}



ing a tack. A decision

Learning hierarchical behavior and motion planning for

autonomous driving

Jingke Wang, Yue Wang, De an Zhany school a



039

arXiv:1711.

CARLA: An Open Urban Driving Simulator ev Dosovitskáv¹, German Ros^{2,1}, Felipe Codevilla^{1,3}, Autonio López¹, and Vladien Koltur

Model-free Deep Reinforcement Learning for Urban

Autonomous Driving

Jianyu Chen*, Bodi Yuan* and Masayoshi Tomizuka

the challensing orban autoenness year, upor ray tarion discup the dimension discup the discupation the discup the discup th

t devel-

you day any anon tri up, we noter cost a simula a r luctio

testing the second sec

Research in anomenous urban driving is hindeed by infrarestrute costs and the logitistical diffra-cision of training and hering systems in the hybrical weld frammening and operating even one orders or response significant from and mappener. And a single vehicle is for from sufficient too hering and welding and the second state of the second state of the second state of the haring and weldingstate. This is the off orders include replacities of states in the second state haring the deviation. This is the off orders including replaced to the second state of the haring and weldingst need in the second state of the second state of the second state of the haring and weldingst need in the second state to second program.

1st Conference on Robot Learning (CoRL 2017), Mountain View, United States



extend the abilities of autonomous agents and increase safety through driver assistance when a human driver is in control. through driver assistance when a human driver is in control. Soveral stranges have always been applied to inter-section hundling, including cooperative [1] and hearistic [2] approaches. Cooperative approaches require vehicle-to-vehicle corresponding. The correct state of the et is a null-heard method based on time-specificien (PTC) [3], [4], which is a widely used heuristic as a safety indicator in the automotive industry [5]. Variants of the TTC approach

reason behind a collision between to reases behind a collision between two antrenamess cars was "failure to anticipate vehicle intent" [100]. Even with higher coder dynamics, the often-supposiziable behavior of human drivers compliants the use of rule based algorithms. Secondi, in many cases an aspert using TFC is overly cantions, while is in balar angulad investion a durin on the second second



figs or Booling or "vield or comping behavior to actively sense

Navigating Occluded Intersections with Autonomous Vehicles

using Deep Reinforcement Learning

David Isele^{1,3}, Reza Rahimi², Akansel Cosgan³, Kaushik Sabramanian⁴ and Kikuo Fujimura

End-to-End Model-Free Reinforcement Learning

for Urban Driving using Implicit Affordances

Marin Toromanoff^{1,23}, Emilie Wirbel²³, Fabien Moutarde¹ ¹Cener for Robotics, MINES ParisTech, PSL ¹Valee Driving Assistance Research ¹Valee.ai

t of our method

cases do exist but they are currently mostly limited to inte keeping and intral control [1, 34]. Deep Reinforcement Learning (DRL) on the other side

lets the algorithm learn by itself by providing a reward sig-nal at each action taken by the agent and thus does not suffer from distribution mismatch. This reward can be sporse are

treen distribution measures. It is invariant can be sported and not describing exactly what the agent should have done but just how good the action taken is locally. The final goal of the agent is to maximize the sams of accumulated rewards and thus the agent needs to think about sequence of actions

rather than instantaneous ones. One of the major drawboc of DRL is that it can need a magnitude larger amount of da

than supervised learning to converge, which can lead to d

ficulties when training large networks with many param ters. Moreover many RL algorithms rely on a replay buffer

ters. Moreover many RL algorithms rely on a replay buffer [21, 25, 12] allowing to karm from past approximates but such buffers can limit the size of the input used (e.g. the size of the image). Thus is why mean hertworks and image size in DRL are usually intry compared to the ones used in su-pervised learning. This they may not be expressive enough to solve and complicated tasks an orbitm driving. Thurding:

ment DRL approaches to autonomous driving are applied

to simpler cases, e.g. only steering control for lang keering

Another drawback of DR1, shared with II, is that the

Abome drawteack or Dick, studed with its, is that his ap-prithm appears as a black box from which it is difficult to inderstand how the decision was taken. A premising way to solve both the data efficiency (pre-

ticularly for DRL) and the black box problem is to use

dances in some recent papers [2, 31]. The idea is to train

a network to prodict high lovel information such as surrap-tic segmentation maps, distance to center of the lumi, traffic-light state etc... This prediction can then be used in sou-ernl ways, either by a classic control are also faster et al. [31], either as anxiitary loss helping to find better [entrops to the main invitative task loss as in Mehra et al. [33] or alor in a model-based RL, approach as in the really recent veck-

of Pan et al. 1261 while also providing some interrectable

feedback on how the decision was taken.

privilered information as auxiliary losses also coin-

[18] or going as fast as possible in racing games [2

Abstract

Reinforcement Learning (RL) sins at learning an opt

nal behavior policy from its own experiments and not rale

based control methods. However, there is no RL algorithm out consider of handling a task or difficult on seban driv yet capable of handhing a sank ar difficult as arbane dri-ting. We present a morel techniques, coincil applicit affor-dancers, so affectively lowrange RL for arbane driving that sh-chahing kane kooping, producting and arbitrary driving that arbane and straffic light detection. To care householder we are the first to present a measurable RL application bounding and a complex to capacitally regarding the testfic light detection. Further,

by winning the Camera Only track of the CARLA challenge.

Urban driving is probably one of the hardest situations to solve for autonomous cars, particularly regarding the interaction on intersections with traffic lights, pedestrians

crossing and cars going on different possible lanes. Soly-

ing this task is still an open problem and it seems compli-

ing this task is still as open problem and it seems complexent to handle such difficult and highly variable situations which classic inface-based approach. This is why a significant part of the state of the art is autonomous divising [20, 4, 5] focuses on end-to-end systems, its. Learning driving policy from data without relying on hand-endfed rules.

Imitation learning (IL) [28] aims to reproduce the behav-

ior of an expert (a human driver for autonomous driving) by

Isoming to minic the control the human driver applied in the same situation. This leverages the massive amount of data annotated with human driving that most of automotive manufacture and supplier can obtain relatively assily. On the other side, as the human driver is always in an almost perfect situation, IL, algorithms suffer from a distribution of the state of the situation of the situati

mismatch, i.e. the algorithm will never encounter failing

cases and thus will not react appropriately in those condi-

tions. Techniques to assement the database with such failing

1. Introduction

an anomen phase (1). Young of the TC spress has a strain of the strain o



used for the intersection handling, such as imitation learning, othine planning, and offline learning. In instation learning, the policy is learned from a human driver [11], however this the policy is learned frem a human driver [11], however this policy does not offer a solvinor if the agent funds itself in state that is not part of the training data. Online planness compare the host action to size by your maining the future states frem the current time state. Online planness based on partially observable Moree Carlo Planning (POMCP) howe been shown to handle interactions [12], but rely on the estimetes of an accurate generative near (Al. Officie learning ackles the interaction problem, othen by using Markub Decision Processor (MDP) in the backet of [13], [14].

exploratory action to better understand the environment. The first contribution of this paper is demonstrating the

<text><text><text><text><text>

environment perception as well as behavior and motion planner, encoding the traffic situation as a hidden representation.

Formally, we state the driving as hierarchical behavior and metion planning (HBMP) problem, which is shown in Fig. 1. The behavior layer makes a decision based on the correct observation, e.g. lane change, while the motion planning layer pickth the migisteriory to complete this decision. Comparing with the existing works on learning behavior Bighe Wang, Yee Wang, Dengkon Zhang and Hong Xiong are the like State Key Lebenstery of Industrial Control and Technol-ry, Zacjang University, Hangdhoo, F.R. Chian, Yathou Yang is with chool of Computing, Informatics, and Decision Systems Engineer-g, Axiona State University, Yae Wang is the corresponding atherplanning [6]-[8] using RL, we set to achieve a policy that ering the coupled action space of motion and behavior, the main challenge is the search inefficiency, especially in the

Research in autonomous urban driving is hindered by infrastructure costs and the logistical diffi-

untig in an physical werk is keysed in terms of most material paper. As a dimension of the maximum of the structure of the s

While ad-hec use of simulation in autonomous driving research is widespread, existing simulation platforms are limited. Open-source racing simulators such as TORCS [28] do not present the com-

Altergi-characteristic definit in a consolid contribution, e.g., the prop (DBL bismochis, in a standord definition of the contribution of the standord definition of the standord defin -Astronomous driving in a crowded envi

metanana sustant of the default, music metanana, rain more than a sustant of the default and the sustainana, rain more than a substant of the default and the substant and the substant and the substant and the substant entire planning: perform a look-shoul suarch in a *belief tree* to to provide an initial plan, which is refined by another neural compute a polycy, execute the first action in the polycy, and re- metwork component.



Panpan Cai, Yuanfu Luo, Aseem Saxena, David Hsu, Wee Sun Lee School of Computing, National University of Singapore, 117417 Singap

A SAMPLING OF PUBLISHED REWARD FUNCTIONS

Paper

Navigating Occluded Intersections with Autonomous Vehicles using Deep Reinforcement Learning [Isele et al., 2018]

Unweighted sum of 3 attributes:

- 0.01 for every step
- 10 if a collision occurred(0 otherwise)
- + 1 when the agent successfully reaches the destination beyond the intersection
 (0 otherwise)

Deep Distributional Reinforcement Learning Based High-Level Driving Policy Determination [Min et al., 2019]

Unweighted sum of 4 attributes:

- + (v-40)/40, where v is speed in km/h within the allowed range [40,80] km/h
- 10 if the ego vehicle collides (0 otherwise)
- + 0.5 if the ego vehicle overtakes another vehicle (0 otherwise)
- 0.25 if the ego vehicle changes lane (0 otherwise)

CARLA: An open urban driving simulator [Dosovitskiy et al., 2017]

Weighted sum of **5** attributes:

- $r = (1) \triangle d + (0.05) \triangle v$
 - + $(-2*10^{-6}) \triangle c$ + $(-2) \triangle s$
 - + (2)∆o
- Δd , the change in distance along the shortest path from start to goal
- $\triangle v$, the change in speed in km/h
- $\triangle c$, the change in collision damage expressed in range [0, 1]
- Δs , the change in the proportion of the ego vehicle overlapping with the sidewalk
- △o, the change in the proportion of the ego vehicle overlapping with the sidewalk

Learning hierarchical behavior and motion planning for autonomous driving [Wang et al., 2020]

Defined separately:

- For transitions to terminal states, one of the following:
- + 100 if the goal was reached
- 50 upon a collision or running out of time
- 10 for a red-light violation
- 1 if the ego vehicle is in the wrong lane
- For transitions to non-terminal states, unweighted sum of **3 attr.**:
- $\Sigma_t t^2 [v_{ref} v(t)] / \Sigma_t t^2$, which rewards speeds close to the desired speed
- 1 / [1 + $\boldsymbol{\Sigma}_t \, \big| \, v(t) \, \big|$, which rewards based on distance traveled

+ $\Sigma_t [0.02*d_{olon}(t) + 0.01*d_{olat}(t)]$, which rewards keeping larger distances



FIND MISMATCHES IN PREFERENCE ORDERINGS

(The most powerful method I'm aware of.)

7 of 9 reward functions* incorrectly prefer T_{crash}.



W. Bradley Knox, Alessandro Allievi, Holger Banzhaf, Felix Schmitt, and <u>Peter Stone</u>. **Reward (Mis)design for Autonomous Driving**. AlJ 2023.

*9 exhaustively characterized papers' reward functions allow this analysis

Let τ_{dest} be a trajectory that successfully reaches the destination. $\tau_{crash} < \tau_{idle} < \tau_{dest}$.



Let r_{dest} be a trajectory that successfully reaches the destination. $r_{crash} < r_{idle} < r_{dest}$.

If $r_A < r_B < \tau_C$, then there is some probability p such that $G(\tau_B) = pG(\tau_C) + (1-p)G(\tau_A)$

p is the *indifference point*.

For each R, we calculate *p*, then convert it to km per collision at the indifference point.

Sanity check failure 3: UNDESIRED RISK TOLERANCE VIA INDIFFERENCE POINTS











Indifference points for collision frequency



Indifference points for collision frequency



Indifference points for collision frequency





OTHER GOTCHAS

• Clipping

• Example: -1,000,000 for collision, +1 for reaching the destination

• Learnable loopholes

• when an agent *increases* its utility/return through behavior that *decreases* its performance in the eyes of its designer(s)

Redundant attributes

Missing attributes \rightarrow negative side effects



Amodei et al., 2016

MINOR SANITY CHECK FAILURES (5-8)

Identify any of these red flags:

Incomplete description of the problem specification in research presentations

- [Speculation] missing details in a paper indicate not considering reward design to be a critical component of the research project
- 9 of 10 exhaustively characterized papers lacked details of their problem specification (that were learned via our correspondence with their authors)



Misalignment by Shaping

REWARD SHAPING

Reward shaping, def. – in addition to the true/environmental reward, providing reward to aid learning, e.g., *by providing behavioral hints or heuristics*

$$r_{shaped} = R(s_t, a_t, s_{t+1}) + R_{shaping}(s_t, a_t, s_{t+1})$$

In practice, most RL problems only have one, *shaped* reward function.

REWARD SHAPING

What leaders in AI say

Russell and Norvig: "As a general rule, it is better to design performance metrics according to what one actually wants to be achieved in the environment, rather than according to how one thinks the agent should behave."

Sutton and Barto agree in almost the same phrasing, adding that imparting knowledge about effective behavior is better done via the initial policy or initial value function.

My version: Specify how to measure outcomes, not how to achieve them.

I.e., *in general*, don't shape rewards.

REWARD SHAPING

What leaders in AI say

Russell and Norvig: "As a general rule, it is better to design performance metrics according to what one actually wants to be achieved in the environment, rather than according to how one thinks the agent should behave."

Sutton and Barto agree in almost the same phrasing.

My version: Specify how to measure *outcomes*, not how to achieve them.

I.e., *in general*, don't shape rewards.
REWARD SHAPING

"Safe" reward shaping

Safety here means that the reward shaping will not change the optimal policy (or ordering over policies).

Some *specific* methods for reward shaping are safe (under some assumptions), but their desirability is still controversial.

If no effort is made to establish that an instance of reward shaping is safe, then it's unsafe.

SAFE REWARD SHAPING METHODS

"Safe" here means that the reward shaping will not change the optimal policy (or ordering over policies).

- Potential-based reward shaping (Ng. et al, 1999) $R_{shaping}(s_t, a_t, s_{t+1}) = \gamma \Phi(s_{t+1}) - \Phi(s_t)$
 - Its assumptions are often overlooked tabular and a proper task
 - \circ Extension to $\Phi(s,a)$ (Wiewiora et al., 2023)
 - Equivalence of potential-based shaping and Q-value initialization (Wiewiora et al., 2023)
 - Dynamic potential-based reward shaping (Devlin and Kudenko, 2012)
- Annealing the shaped reward (Behboudian et al., 2020; Szoke et al. 2024)

REWARD SHAPING IN PRACTICE

Reward shaping in RL for AD papers

- 13 of 19 include reward shaping
- Some examples of behavior that shaped rewards encourage
 - staying close to the center of the lane [Jaritz et al., 2018]
 - increasing distances from other vehicles [Wang et al., 2020]
 - avoiding overlap with the opposite-direction lane [Dosovitskiy et al., 2017, Liang et al., 2018]

REWARD SHAPING IN PRACTICE

Reward shaping in RL for AD papers

- Of those 8 exhaustively characterized papers that include reward shaping ...
 - $\circ~$ 0 explicitly describe the separation of their shaping rewards and their true rewards
 - O use a recognized method of safe reward shaping or discuss safety of reward shaping
 - 2 acknowledge usage of reward shaping
 - 1 acknowledges its potential adverse effects

Recommendation: Create an aligned reward function without shaping, then optionally add a shaping reward function.

Why?

- Clarity
 - The reward function should create an aligned problem specification.
 - The shaping rewards give policy guidance and may change the problem specification.
- Debugging ("overfit" plot) -



Misalignment by Trial-and-Error Design

Imagine you want to design a new RL problem.

Imagine you want to design a new RL problem.

How might you approach this?

Step 1: Design a candidate RL problem, including R.

Step 1: Design a candidate RL problem, including R.

Step 2: Pick an RL algorithm for testing.

Step 1: Design a candidate RL problem, including R.

Step 2: Pick an RL algorithm for testing.

Step 3: Learn a behavior policy.

Step 1: Design a candidate RL problem, including R.

Step 2: Pick an RL algorithm for testing.

Step 3: Learn a behavior policy.

Step 4: If the policy isn't right, update the RL problem (especially the reward function) and repeat.

This trial-and-error process is the norm.

RL for AD

Of 8 papers whose authors shared their reward design process over email,

100% used trial-and-error to design their reward function.

General RL experts

We surveyed 24 expert RL practitioners. 92% used trial-and-error to design their most recent reward function.



reward design iterations



Trial-and-error reward design iterations

Training context - RL algorithm, hyperparameters, and tasks



Trial-and-error reward design iterations

Training context - RL algorithm, hyperparameters, and tasks



Trial-and-error reward design iterations

Hungry Thirsty Domain



Singh et al., 2009, Where Do Rewards Come From?

Finding the potential for overfitting.





Finding: Reward functions that achieve the best performance in one learning context can be suboptimal in another.



For all experiments, we find the best performing reward functions differ across learning contexts.

This shows potential for overfitting.

H2: The cumulative performances achieved with different reward functions are *uncorrelated* across different learning contexts.



We rank all reward functions for each experiment setting (D D D_2

We compare the ordering of these rankings using Kendall's tau.

Distribution Sample

Finding: The cumulative performances achieved with different reward functions are uncorrelated across different learning contexts.

\mathcal{D}_1	${\cal D}_2$	$ au_b$
$\gamma = 0.99$	$\gamma = 0.8$	0.07
$\gamma = 0.99$	$\gamma = 0.5$	0.04
$\gamma = 0.8$	$\gamma = 0.5$	0.12
$\alpha = 0.25$	$\alpha = 0.05$	0.11

We rank all reward functions for each experiment setting (D D D_2

We compare the ordering of these rankings using Kendall's tau.

We find that these rankings are **uncorrelated** ($|\tau_b| < 0.1$) or **slightly correlated** ($|\tau_b| < 0.2$).

User Study Conducted in Jupyter Notebooks





Experts overfit reward functions too

68% of users overfit reward functions

User P20 first tried a reward function which achieved a mean score of 138,092 with DDQN.

They ultimately selected a different reward function, which achieved a mean score of 1,031 with DDQN.

Experts' reward functions tend to not generalize.



Hard configuration

(15 steps between water & food)



Easy configuration

(5 steps between water & food)

Experts' reward functions tend to not generalize.



53% of RL experts submitted reward functions that had optimal policies which do not perform the hard configuration well.

Hard configuration

(15 steps between water & food)

Experts are currently bad at writing reward functions.



53% of RL experts wrote reward functions which **failed to encode the task** in the hard case.

For example, **P3'**s reward function:

$$egin{aligned} r(
eglinetwidtharpinetwid$$

Hard configuration

(15 steps between water & food)

Most experts (83%) used a *myopic* design strategy of using reward to order states by their desirability.

Example

"It's best to not be hungry and thirsty, so I'll set that to the max, 1. Being not thirsty is better than being not hungry [so 0.3 for only hungry /not thirsty and -0.35 for only thirsty / not hungry]. Worst is at hungry AND thirsty; setting that to -1" — Participant 25

Is the word "reward" harming reward design?

Reasoning about reward accumulation (return) is done poorly.

Example reused

"It's best to not be hungry and thirsty, so I'll set that to the max, 1. Being not thirsty is better than being not hungry [so 0.3 for only hungry /not thirsty and -0.35 for only thirsty / not hungry]. Worst is at hungry AND thirsty; setting that to -1" — Participant 25



Trial-and-error reward design can overfit to the training context (RL algorithm, hyperparameters, and task).

And RL experts appear to do so in practice. Impact on algorithmic comparison and ablation studies?

Misalignment by Discounting

REWARD AND RETURN

<u>Field</u>

reinf. learning motion planning control theory evolutionary algs. utility theory optimization

$G(\tau) = \sum_{t=0}^{T} f$	$\gamma^t R(s_t, a_t, s_{t+1})$
return	Tewara
-1 × cost	-1 × cost
-1 × cost	-1 × cost
fitness	-
utility	-
objective function*	-
performance metric	-
score	-

* "objective function" more precisely refers to the expectation of $G(\tau)$

REWARD AND RETURN

T $G(\tau) = \sum \gamma^t R(s_t, a_t, s_{t+1})$ t=0

REWARD AND RETURN

T $G(\tau) = \sum \gamma^t R(s_t, a_t, s_{t+1})$ t=0discount factor



Time steps until a reward

Contemporary RL tends to have <u>2 discount factors</u>: problem-side and algorithmic

Problem-side, γ_{MDP} - part of the MDP definition

- determines how return should be calculated when evaluating a policy's performance (e.g., for comparing algorithms or reporting results in a publication)
- with a start state distribution, determines the ranking of policies and therefore the set of optimal policies

Algorithmic, γ_{ala} - a hyperparameter of the RL algorithm

- $\bullet \quad \gamma_{alg} \leq \gamma_{MDP}$
- $\gamma_{alg} \le 0.999$ in deep RL papers I have seen, usually $\gamma_{alg} \le 0.99$
- in practice, γ_{alg} trades stability during learning at the cost of greater distance between the RL algorithm's loss function and the task objective

Do not confuse the two! We focus on γ_{MDP} unless otherwise stated.
Contemporary RL tends to have <u>2 discount factors</u>: problem-side and algorithmic

Problem-side, γ_{MDP} - part of the RL problem definition

• creates the true return

Algorithmic, γ_{alg} - a hyperparameter of the RL algorithm

- $\bullet \quad \gamma_{\text{alg}} \leq \gamma_{\text{MDP}}$
- $\gamma_{alg} \le 0.999$ in deep RL papers I have seen, usually $\gamma_{alg} \le 0.99$
- in practice, γ_{alg} trades stability during learning at the cost of greater distance between the RL algorithm's loss function and the task objective

Do not confuse the two! We focus on γ_{MDP} unless otherwise stated.

Estimating return during RL at absorbing state vs. when stopping an episode for other reasons

If stopping at absorbing state—i.e., satisfying termination conditions—the absorbing state value is 0 except under highly unusual circumstances.

When function approximation is used, there is danger that value inference will return a nonzero value. You can use $\gamma=0$ to get the equivalent effect as having a value of 0.

If stopping at non-absorbing state—i.e., without satisfying termination conditions—include the value of the final state discounted by γ_{ala} (or γ_{MDP}).



The set of optimal policies can change as the discount factor changes.



In this continuing domain,

- if $\gamma < 0.5$, then choosing the left loop from s is optimal
- if $\gamma > 0.5$, then choosing the fight loop from s is optimal

Separate intuitive argument: if changing γ didn't change the set of optimal policies, then we would just set $\gamma=0$ and forget about the credit assignment problem.

To develop intuition about your discounting, calculate time-to-10% value (and 1% and 0.1%) via $\log_{\rm v}$

Example: Autonomous driving often has 100ms time steps.

lf **γ=0.9**,

the rewards are discounted to X% of their full value this far in the future:

- $10\% 2.19 s = \log_{10} 0.1 * 0.1s$
- 1% 4.37 s = log 0.01 * 0.1s
- $0.1\% 6.56 s = \log_{0.001} * 0.1s$

It takes a constant amount of time for each reduction by a factor 0.1.

To develop intuition about your discounting, calculate time-to-10% value (and 1% and 0.1%) via $\log_{\rm v}$

Example: Autonomous driving often has 100ms time steps.

lf **γ=0.99**,

the rewards are discounted to X% of their full value this far in the future:

- $10\% 22.9s = \log_{10} 0.1 * 0.1s$
- 1% 45.8s = log_0.01 * 0.1s
- $0.1\% 68.7s = \log_{10} 0.001 \times 0.1s$

Even with a relatively high γ =0.99, events one minute into the future likely have negligible effect on the value function!

To develop intuition about your discounting, calculate time-to-10% value (and 1% and 0.1%) via $\log_{\rm v}$

Example: Autonomous driving often has 100ms time steps.

lf **γ=0.999**,

the rewards are discounted to X% of their full value this far in the future:

- 10% 230 s / 4 min
- 1% 460 s / 8 min
- 0.1% 690 s / 12 min

Each 10x decrease in $(1 - \gamma)$ results in a ~10x increase in horizon.

While a precise horizon does not exist, there is an order of magnitude in which discounting goes from being significant to being negligible.

Make all episodic tasks undiscounted.

Exponential discounting is a seemingly necessary evil in continuing tasks. It ensures finite returns and encourages getting reward sooner.

But it has **drawbacks**:

1) It appears to **decrease alignment** with humans, who do not evaluate outcomes with exponential discounting.

2) It makes return less legible / human-readable.

It's not necessary though in episodic tasks, so to avoid these drawbacks it should not be used.

Whether you use discounting for your *algorithm* is a different matter.

A continuing exponentially discounted task may not have an optimal policy under function approximation.



Argument comes from Discounted Reinforcement Learning Is Not an Optimization Problem by Naik et al. (2019).

A continuing exponentially discounted task may not have an optimal policy under function approximation.



Argument comes from Discounted Reinforcement Learning Is Not an Optimization Problem by Naik et al. (2019).

A continuing exponentially discounted task may not have an optimal policy under function approximation.



Under the optimality criterion $v^{\pi}(s) \ge v^{\pi'}(s)$ for all states *s* and all **policies** π' , there may be no optimal *representable* policy.

Can we still specify an optimal *representable* policy by setting the start state distribution? I.e., set $J(\pi) = \mathbb{E}_{s_0 \sim p(s_0)}[v_{\pi}(s_0)]$

Not if we want an aligned learning objective.

- Over the infinite time of a continuing task, the state visitation distribution may have no support for the states that are visited from start states within the discount factor's "horizon" of non-negligible impact.
- Generally violates the idea that we care about performance over an infinite task, not just at its start.

Designing Aligned Reward

There are no best practices! (Well, not yet.)

But our methods for catching misalignment might help.

Sketch of possible best practices

- 1. Consider the simplest set of **outcome variables** that differentiate varying levels of success vs. failure.
 - Find a per-time step version of each outcome variable that adds up to its full-trajectory value.
 - Example: time to goal
 - Example: soccer
- 2. Create a **parametrized reward function representation** with these variables.
 - Recommendation: try a linear representation and stubbornly try to make it work
- 3. **Tune the parameters** so that its preference ordering over outcome distributions matches yours.
- 4. Evaluate.

At any point, you may learn something that causes you to return to an earlier step.

Sketch of possible best practices

- 1. Consider the simplest set of outcome variables that differentiate varying levels of success vs. failure.
 - Find a per-time step version of each outcome variable that adds up to its full-trajectory value.
 - Example: time to goal
 - Example: soccer
- 2. Create a parametrized reward function representation with these variables.
 - Recommendation: try a linear representation and stubbornly try to make it work
- 3. Tune the parameters so that its preference ordering over outcome distributions matches yours.
- 4. Evaluate.

At any point, you may learn something that causes you to return to an earlier step.

Sketch of possible best practices

1. Consider the simplest set of outcome variables that differentiate varying levels of success vs. failure.

Methods for finding misalignment become methods for optimizing

- Example: time to the reward function via RLHF.
- Example: soccer
- 2. Create a parametrized reward function representation with these variables.
 - Recommendation: try a linear representation and stubbornly try to make it work
- 3. Tune the parameters so that its preference ordering over outcome distributions matches yours.
- 4. Evaluate.

At any point, you may learn something that causes you to return to an earlier step.



Tools and insights

- Catching misalignment via preference mismatch (R vs. human)
 - Preferences over trajectorie
 - Preferences over trajectory lotteries
- Shaping
 - Keep shaping in a separate function.
 - Plot true return vs. shaped return to detect overfitting.
 - Consider a shaping method that doesn't change the preference ordering over policies / outcome distributions.
- Discounting
 - $\circ \quad \text{Keep a separate problem-side } \gamma.$
 - Calculate time to 10% / 1% / 0.1% value.
- Others
 - Consider how the RL alg modifies R e.g., clipping
 - Bias towards designing an R with legible return and expected return

What Should We Work Towards?

Promising projects

- Validated best practices for aligned reward function design
- When reward cannot be practically aligned
 - Some valued outcomes aren't measurable by the learning system
- Debug methods --> debug tools

Al safety agenda: Expose where reward design can cause dangerous misalignment. Fix it if possible. Otherwise, identify where it should not be used.

Reward design consultation

Free for academic or non-profit projects

Designing a reward function?

Want to talk about it with someone?



Email bradknox@cs.utexas.edu and ask for a 30 minute consultation.

Disclaimer: I research the design of reward functions. I want to help you while developing my methodology for doing these consultations, which may eventually be published as best practices for reward function design. This is not a formal investigation, but I do hope to learn from you what was helpful and what was not.

Our papers on reward design

W. Bradley Knox, Alessandro Allievi, Holger Banzhaf, Felix Schmitt, and <u>Peter Stone</u>. **Reward (Mis)design for Autonomous Driving**. AIJ 2023.

Serena Booth, W. Bradley Knox, Julie Shah, Scott Niekum, Peter Stone, Alessandro Allievi. **The perils of trial-and-error reward design: misdesign through overfitting and invalid task specifications.** AAAI 2023.

Brad Knox <bradknox@cs.utexas.edu>