# Controlling Blood Glucose For Patients With Type 1 Diabetes Using Deep Reinforcement Learning – The Influence Of Changing The Reward Function


## Abstract

Reinforcement learning (RL) is a promising direction in adaptive and personalized type 1 diabetes (T1D) treatment. However, the reward function – a most critical component in RL – is a component that is in most cases hand designed and often overlooked. In this paper we show that different reward functions can dramatically influence the final result when using RL to treat in-silico T1D patients.


## 1 Introduction

Reinforcement learning (RL) is a separate direction in machine learning where the aim is to understand and automate goal-directed learning and decision-making [13]. In combination with recent advances in deep learning, deep reinforcement learning has emerged as a very powerful tool for difficult control tasks [11, 6].

The artificial pancreas (AP) is a system involving an insulin pump, a continuous glucose monitor and a control algorithm to release insulin in response to changing blood glucose (BG) levels mimicking a human pancreas. Several works have shown promising results using RL for the AP [2, 7, 8, 12], but the main focus of these algorithms have been on fitting the RL framework to the case of type 1 diabetes (T1D). In this work we focus on the *reward function*, an often overlooked component of empirical reinforcement learning. It is well known that the success of a RL application strongly depends on how well the reward signal frames the goal of the application's designer and how well the signal assesses progress in reaching that goal [18]. In the diabetes case it is particularly the contrasting problems of hyper- and hypoglycemia – too high or too low BG levels –that is problematic for RL applications. We propose several new reward functions suited for (T1D), and perform in-silico experiments testing different reward functions on the trust-region policy optimization (TRPO) algorithm [9] using the Hovorka model [4].

Our experiments demonstrate that focusing on reward functions that contain more domain knowledge, such as stronger penalties for reaching low BG levels, is crucial.

## 2 Deep reinforcement learning: Policy optimization and TRPO

Policy gradient algorithms consider *parametric policies* which are optimized using gradient ascent on a given *performance measure*. The most common choice for the performance measure is the expected return of the start state $s_0$, given as $J(\theta) = v_\pi(s_0) = \mathbb{E}_\pi\left[R_0 + \gamma R_1 + \gamma^2 R_2 + \cdots\right]$.

Using policy gradient algorithms yield several benefits: the *policy gradient theorem*, application of RL to continuous action spaces and a naive extension to deep learning using neural network to parameterize the policies.

Furthermore, a key point of using policy gradient algorithms is the policy gradient theorem [13]:

$$\nabla J(\theta) \propto \sum_s \mu(s) \sum_a q_\pi(s,a) \nabla \pi(a|s,\theta).$$

This states that the gradient of the performance measure is proportional to the gradient of the policy itself. This allows the use of any differentiable policy parameterization. Furthermore, the policy gradient theorem is constructive, so it directly yields a simple sample-based algorithm, REINFORCE [16], omitted here for brevity. This al-



https://doi.org/10.7557/11.5200

gorithm has been well studied and a number of improvements and suggestions have been proposed, see e.g. [9, 10, 5]. The current state-of-the-art in *model free* policy gradient algorithms is *Trust Region Policy Optimization* (TRPO) by Schulman et al. [9] and a simplified version, *Proximal Policy Optimization* [10]. In this work we restrict our attention to TRPO.

Trust region policy optimization is a policy gradient algorithm where each update of the policy is guaranteed to improve the performance. This guarantee is achieved, by enforcing the Kullback-Leibler divergence between the old and the updated policy to be small:

$$\begin{aligned} \underset{\theta}{\text{maximize}} \quad & \mathbb{E}_{s,a\sim\pi_{\theta_{old}}}\left[\frac{\pi_{\theta(a|s)}}{\pi_{\theta_{old}(a|s)}}Q_{\theta_{old}(s,a)}\right] \\ \text{subject to} \quad & \mathbb{E}_{s,a\sim\pi_{\theta_{old}}}\left[D_{KL}(\pi_{\theta_{old}},\pi_\theta)\right] \leq \delta. \end{aligned} \tag{1}$$

We refer the reader to Schulman et al. [9] for further details. The policy $\pi_\theta(a|s)$ is a Gaussian policy:

$$\pi_\theta(a|s) = \frac{1}{\sigma(s,\theta)\sqrt{2\pi}}\exp\left(-\frac{(a-\mu(s,\theta))^2}{2\sigma(s,\theta)^2}\right),$$

where $\sigma(s,\theta)$ and $\mu(s,\theta)$ are feature extractors. We use neural network feature extractors in this work.

# 3 Reward functions

We consider hyperglycemia as values above $bg_{hyper} = 180$ mg/dL, hypoglycemia as values below $bg_{hypo} = 72$ mg/dL and severe hypoglycemia as values below $bg_{hypo^-} = 54$ mg/dL. Thus, normoglycemic range are values between $[bg_{hypo}, bg_{hyper}]$ mg/dL, with a target value $bg_{ref} = 108$ mg/dL. The nine different proposed and tested reward functions can be further divided into two categories:
(1) Symmetric reward functions – hyper- and hypoglycemia are equally penalized by the rewards.
**Absolute reward** [17]: $|bg - bg_{ref}|$
**Binary**:

$$\begin{cases} 1 & : bg \in [bg_{hypo}, bg_{hyper}] \\ 0 & : otherwise \end{cases}$$

**Binary tight**:

$$\begin{cases} 1 & : bg \in [bg_{ref} - 10, bg_{ref} + 10] \\ 0 & : otherwise \end{cases}$$

**Gaussian reward** [2]: $\exp\left(-\frac{1}{\sigma^2}\left(bg - bg_{ref}\right)^2\right)$
**Squared reward**: $-(bg - bg_{ref})^2$

(2) The second category is asymmetric reward functions – hand-designed reward functions including external knowledge from the diabetes disease to give more penalty to hypoglycemic events.
**T1D reward**: Linear function with positive reward for normoglycemic range. Exponential function with negative reward for hypoglycemia, while 0 reward for hyperglycemia. Really negative reward for severe hypoglycemia.

$$\begin{cases} -100 & : bg < bg_{hypo^-} \\ \exp(\frac{\log(140.9)}{bg_{hypo}}bg) - 140.9 & : bg \in [bg_{hypo^-}, bg_{hypo}] \\ \frac{1}{36}bg - 2 & : bg \in [bg_{hypo}, bg_{ref}] \\ -\frac{1}{72}bg + \frac{5}{2} & : bg \in [bg_{ref}, bg_{hyper}] \\ 0 & : bg > bg_{hyper} \end{cases}$$

**Tight T1D reward**: Hypoglycemia considered as values below $bg_{hypo_t} = 90$ mg/dL in order to be even more aggressive against hypoglycemic events.

$$\begin{cases} -100 & : bg < bg_{hypo^-} \\ \exp(\frac{\log(117.5)}{bg_{hypo_t}}bg) - 117.5 & : bg \in [bg_{hypo^-}, bg_{hypo_t}] \\ \frac{1}{18}bg - 5 & : bg \in [bg_{hypo_t}, bg_{ref}] \\ -\frac{1}{72}bg + \frac{5}{2} & : bg \in [bg_{ref}, bg_{hyper}] \\ 0 & : bg > bg_{hyper} \end{cases}$$

**Hovorka reward**: Based on the nonlinear model predictive control from [4].
$-(bg - y(t))^2$
$y(t)$ is the desired glucose profile. When BG levels are above the desired level $y(t)$ linearly decrease, while for BG values below target value $y(t)$ exponentially increases [4].
**Risk reward** [1]: $-10(1.509(\log(bg)^{1.084} - 5.381))^2$

# 4 Experimental setup

**Simulation environment** We use the Hovorka simulator as described in Wilinska et al. [15] and Hovorka et al. [4]. The simulator is implemented in Python and the TRPO agent is trained using the open source reinforcement learning toolbox *garage*[1] [3].

**Experiment protocol and scenarios** Each episode of the simulations consists of a single day

[1] https://github.com/rlworkgroup/garage.

plus 12 hours into the next day. Four meals are given at [08:00, 12:00, 18:00, 22:00] with a uniform chance of moving the meal back or forward 30 minutes. Each meal is fixed at 40, 80, 60 and 30 grams of carbohydrates with a uniform ±20 gram disturbance. Finally, we have a ±30% carbohydrate counting error, meaning that the carbohydrate amount used to calculate the bolus insulin dose might be 30% higher or lower than the true carbohydrate amount.

**Performance measures and testing** We test the algorithm on a fixed scenario consisting of 100 random meal-days generated with a fixed random seed. To measure the performance of our simulations, we use time-in-range and time-in-hypoglycemia as the performance measures, where we want to maximize the former and minimize the latter. We also include low blood glucose risk index (LBGI), high blood glucose risk index (HBGI), risk index (RI) and the coefficient of variation (CoV), all described in Clarke and Kovatchev [1].

# 5 Results and discussion

In this work we test and compare different reward functions using TRPO on the original Hovorka in-silico patient, [4], in order to show the importance of the reward function design.

In the experiments we consider two cases, with different insulin-to-carbohydrate ratio (ICR) used to calculate pre-meal bolus insulin doses. This ratio specifies the number of grams of carbohydrate covered by each unit of insulin, see e.g. [14].

Given the fact that we are in this work considering a single-hormone AP, the only available action for the algorithm when the BG is too low or approaching low levels is to turn off the insulin pump. Due to this the actual ICR used during meals will have a strong influence on the overall result. Especially the severity of carbohydrate counting errors, which we include in our simulations, will be affected by different ICRs.

## 5.1 Case 1: 30g/U ICR

We start with a 30g/U ICR. This translates to the in-silico Hovorka patient taking 1 unit of insulin for each 30 grams of carbohydrate intake. We run the TRPO algorithm for 100 iterations using all the reward functions described in Section 3. Figure 1 shows mean BG level values for the different reward functions used within TRPO and the basal-bolus regimen. The mean BG values show good performance for all the different reward functions and basal-bolus regimen when using 30 g/U ICR as shown in figure 1, spending most of the time within range. However, most of the symmetric rewards show lower values than the asymmetric rewards, resulting in a higher hypoglycemia risk. Only the *tight binary reward function* shows comparable results to the asymmetric reward functions, keeping mean BG values closer to the target value. Results from these experiments are summarized in Table 1.

TRPO outperforms the basal-bolus regimen in terms of time-in-range for all the reward functions tested. However, that is not the case in terms of hypoglycemic events, where the symmetric rewards struggle to avoid hypoglycemia. Only the symmetric *binary tight reward* function presents competitive results avoiding hypoglycemic excursions in similar terms to asymmetric rewards. The *risk reward* function actually increases the time spent in hypoglycemia, showing worse results than the rest of the asymmetric rewards. The opposite happens with hyperglycemic excursions, where the symmetric reward functions show better performance avoiding hyperglycemia. This is because the symmetric reward functions deal equally with hypo- and hyperglycemia events, while asymmetric reward functions are designed taking into account external knowledge from the diabetes problem. In this work, this external information consists of higher penalty to hypoglycemia than to hyperglycemia, which is translated into safer behaviour reducing the time spent in hypoglycemic events. This is also reflected in the risk factors, where the asymmetric reward functions are more robust against risk of hypoglycemia than the symmetric reward functions, while both kind of functions show similar performance in terms of hyperglycemic risk. Therefore, the overall risk factor is lower for the asymmetric rewards. Finally, the asymmetric reward functions where hypoglycemia is penalized more than hyperglycemia also present lower CoV, and only the asymmetric risk reward function show similar results to the symmetric functions.

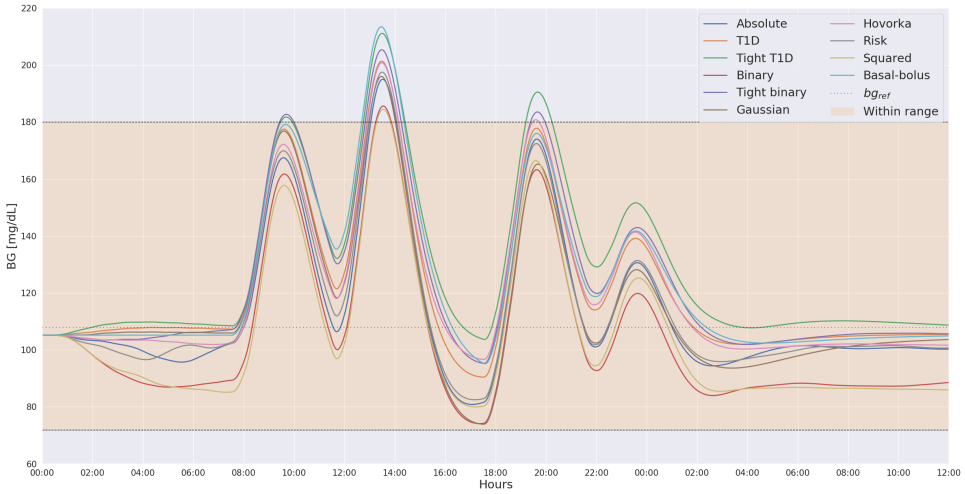


Figure 1: Mean blood glucose levels using TRPO with different reward functions, averaged over 100 episodes. Each test episode runs for one and a half day, a total of 36 hours, to include the effects of the algorithm after the last meal. The Insulin-to-carbohydrate ratio is fixed at 30 g/U.

| Treatment | Time-in-range | -hypo | -hyper | LBGI | HBGI | RI | CoV |
|---|---|---|---|---|---|---|---|
| Basal-bolus | 83.45±7.38 | 2.42±4.9 | 14.13±7.07 | 0.87±0.83 | 4.62±1.45 | 5.5±1.63 | 27.61 |
| Absolute | 86.45±6.04 | 4.50±5.74 | 9.06±4.31 | 1.30±0.82 | 4.23±0.96 | 5.53±1.09 | 27.31 |
| Asymmetric | 88.10±4.78 | 0.54±1.8 | 11.36±4.58 | 0.47±0.31 | 4.34±0.98 | 4.81±1.02 | 25.27 |
| Tight asymmetric | 83.57±5.31 | **0.0±0.0** | 16.43±5.31 | **0.12±0.09** | 4.70±1.17 | 4.82±1.17 | **25.13** |
| Binary | 86.92±6.47 | 6.68±6.39 | 6.40±3.96 | 2.38±0.6 | 3.83±0.94 | 6.20±1.0 | 29.56 |
| Tight binary | 85.03±5.51 | 0.55±2.09 | 14.41±5.31 | 0.41±0.3 | 4.79±1.11 | 5.19±1.16 | 26.43 |
| Gaussian | 84.39±7.16 | 6.15±6.47 | 9.46±4.13 | 1.52±0.98 | 4.24±1.06 | 5.76±1.44 | 27.64 |
| Hovorka | 88.95±3.94 | **0.0±0.0** | 11.05±3.94 | 0.41±0.16 | 4.20±0.81 | **4.61±0.82** | 25.34 |
| Risk | 86.60±5.79 | 3.75±4.81 | 9.65±4.31 | 1.13±0.63 | 4.35±1.0 | 5.48±1.06 | 27.28 |
| Squared | **89.81±5.22** | 4.13±5.12 | **6.06±3.79** | 2.30±0.55 | **3.62±0.89** | 5.91±0.89 | 29.01 |

Table 1: Summary of results for 30g/U insulin-to-carbohydrates ratio. Mean values ± standard deviation of 100 runs with each episode running for one and a half day, a total of 36 hours.

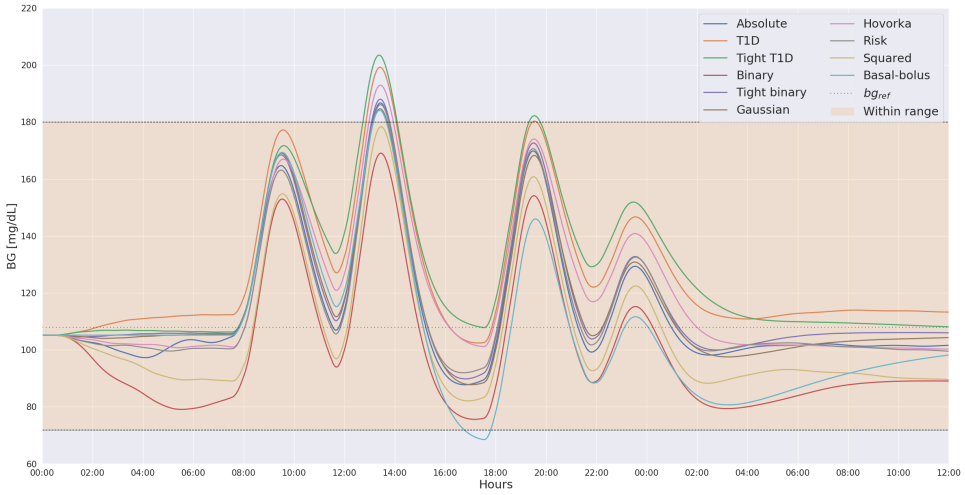


Figure 2: Mean blood glucose levels using TRPO with different reward functions, averaged over 100 runs. Each test episode runs for one and a half day, a total of 36 hours. Insulin-to-carbohydrate ratio fixed to 25g/U.

### 5.2 Case 2: 25g/U ICR

We select a 25g/U ICR for the second set of experiments. That means the in-silico Hovorka patient uses 1 unit of insulin for each 25 grams of carbohydrates intakes. Therefore, in this set of experiments the patient uses more units of insulin to deal with the same amount of carbohydrates. The mean BG level values for the the basal-bolus regimen and the different reward functions used within TRPO during these experiments are shown in figure 2.

TRPO shows good performance with mean BG values within range most of the time. However, symmetric reward functions lead to lower BG values and then higher risk of hypoglycemia, while asymmetric reward functions stay in safer glucose levels.

Results summarized in table 2 show TRPO clearly improving time spent in target range while reducing hypoglycemic events in comparison with the basal-bolus regimen, which in this case is not able to maintain safe BG values.

Furthermore, the asymmetric reward functions taking into account the importance of avoiding hypoglycemia perform better than symmetric reward functions, dramatically reducing hypoglycemic events. This is also reflected in the reduced overall risk index. The symmetric reward functions deals better with high BG values, reducing the time spent in hyperglycemia. However, in spite of this reduction in time spent in hyperglycemia, the risk of hyperglycemia is similar for symmetric and asymmetric reward functions, with the *asymmetric T1D reward* function showing the lowest risk. Therefore, asymmetric reward functions results in lower total risk factor. Regarding the coefficient of variation, the asymmetric T1D reward function shows better performance decreasing variance, while symmetric binary reward function presents a CoV value closer to the basal-bolus strategy. The rest of the reward functions present similar results, reducing the CoV with respect to the basal-bolus regimen.

## 6 Conclusions

Recent research in diabetes and deep learning have facilitated the adoption of new methods and techniques in T1D. At this point, deep RL emerges as a smart, personalized and optimal solution for the AP. However, the success of a RL application strongly depends on the design of the reward function, which is a critical part of any application of RL. In this work we show the influence of including domain knowledge in the reward function design and how this design affects the performance and desired behaviour of the control task.

| Treatment | Time-in-range | -hypo | -hyper | LBGI | HBGI | RI | CoV |
|---|---|---|---|---|---|---|---|
| Basal-bolus | 79.22±12.14 | 14.65±12.73 | 6.14±4.32 | 2.99±1.98 | 3.63±1.15 | 6.62±2.32 | 29.1 |
| Absolute | 91.41±4.69 | 1.62±3.41 | 6.98±3.74 | 0.79±0.42 | 3.73±0.83 | 4.52±0.87 | 24.28 |
| Asymmetric | 87.81±4.92 | **0.08±0.62** | 12.11±4.85 | 0.25±0.31 | **2.87±0.74** | **3.12±0.79** | **22.39** |
| Tight asymmetric | 86.90±5.4 | 0.15±0.89 | 12.95±5.42 | **0.19±0.21** | 3.91±1.0 | 4.11±1.04 | 23.57 |
| Binary | 91.99±5.74 | 4.86±5.72 | **3.15±2.70** | 2.72±0.48 | 2.93±0.78 | 5.66±0.85 | 27.29 |
| Tight binary | 90.48±4.82 | 1.78±3.71 | 7.74±3.7 | 0.59±0.45 | 3.73±0.80 | 4.33±0.88 | 23.47 |
| Gaussian | 91.44±4.95 | 1.44±3.27 | 7.12±3.75 | 0.70±0.42 | 3.59±0.8 | 4.29±0.88 | 23.71 |
| Hovorka | 90.97±4.23 | 0.09±0.55 | 8.94±4.23 | 0.46±0.22 | 3.66±0.79 | 4.11±0.80 | 24.02 |
| Risk | 91.52±4.49 | 1.57±3.08 | 6.91±3.69 | 0.74±0.41 | 3.53±0.76 | 4.26±0.80 | 23.85 |
| Squared | **92.82±4.73** | 2.54±4.4 | 4.64±3.2 | 1.67±0.45 | 3.28±0.79 | 4.95±0.81 | 25.79 |

Table 2: Summary of results for 25g/U insulin-to-carbohydrates ratio. Mean values ± standard deviation of 100 runs with each episode running for one and a half day, a total of 36 hours.

# References


[1] W. Clarke and B. Kovatchev. Statistical tools to analyze continuous glucose monitor data. *Diabetes technology & therapeutics*, 11(S1):S–45, 2009.

[2] M. De Paula, L. O. Avila, and E. C. Martinez. Controlling blood glucose variability under uncertainty using reinforcement learning and gaussian processes. *Applied Soft Computing*, 35:310–332, 2015.

[3] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. Benchmarking deep reinforcement learning for continuous control. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48, pages 1329–1338, 2016.

[4] R. Hovorka, V. Canonico, L. J. Chassin, U. Haueter, M. Massi-Benedetti, M. O. Federici, T. R. Pieber, H. C. Schaller, L. Schaupp, T. Vering, et al. Nonlinear model predictive control of glucose concentration in subjects with type 1 diabetes. *Physiological measurement*, 25(4):905, 2004.

[5] S. M. Kakade. A natural policy gradient. In *Advances in neural information processing systems*, pages 1531–1538, 2002.

[6] S. S. Mousavi, M. Schukat, and E. Howley. Deep reinforcement learning: an overview. In *Proceedings of SAI Intelligent Systems Conference*, pages 426–440. Springer, 2016.

[7] J. N. Myhre, I. K. Launonen, S. Wei, and F. Godtliebsen. Controlling blood glucose levels in patients with type 1 diabetes using fitted q-iterations and functional features. In *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2018.

[8] P. D. Ngo, S. Wei, A. Holubová, J. Muzik, and F. Godtliebsen. Reinforcement-learning optimal control for type-1 diabetes. In *2018 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*, pages 333–336. IEEE, 2018.

[9] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

[10] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[11] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[12] Q. Sun, M. V. Jankovic, and S. G. Mougiakakou. Reinforcement learning-based adaptive insulin advisor for individuals with type 1 diabetes patients under multiple daily injections therapy. *arXiv preprint arXiv:1906.08586*, 2019.

[13] R. S. Sutton and A. G. Barto. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

[14] J. Walsh and R. Roberts. *Pumping insulin: everything you need for success on a smart insulin pump*. Torrey Pines Press, 2006.

[15] M. E. Wilinska, L. J. Chassin, C. L. Acerini, J. M. Allen, D. B. Dunger, and R. Hovorka. Simulation environment to evaluate closed-loop insulin delivery systems in type 1 diabetes. *Journal of diabetes science and technology*, 4(1):132–144, 2010.

[16] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[17] S. Yasini, M. Naghibi-Sistani, and A. Karimpour. Agent-based simulation for blood glucose control in diabetic patients. *International Journal of Applied Science, Engineering and Technology*, 5(1):40–49, 2009.

[18] H. Zou, T. Ren, D. Yan, H. Su, and J. Zhu. Reward shaping via meta-learning. *arXiv preprint arXiv:1901.09330*, 2019.