machines

MDPI

Article

Utilizing Human Feedback in Autonomous Driving: Discrete vs. Continuous

Maryam Savari and Yoonsuck Choe *

Department of Computer Science & Engineering, Texas A&M University, College Station, TX 76549, USA; maryamsavari@tamu.edu
* Correspondence: choe@tamu.edu; Tel.: +1-979-845-5466

Abstract: Deep reinforcement learning (Deep RL) algorithms are defined with fully continuous or discrete action spaces. Among DRL algorithms, soft actor–critic (SAC) is a powerful method capable of handling complex and continuous state–action spaces. However, a long training time and data efficiency are the main drawbacks of this algorithm, even though SAC is robust for complex and dynamic environments. One of the proposed solutions to overcome this issue is to utilize human feedback. In this paper, we investigate different forms of human feedback: head direction vs. steering and discrete vs. continuous feedback. To this end, a real-time human demonstration from steer and human head direction with discrete or continuous actions were employed as human feedback in an autonomous driving task in the CARLA simulator. We used alternating actions from a human expert and SAC to have a real-time human demonstration. Furthermore, to test the method without potential individual differences in human performance, we tested the discrete vs. continuous feedback in an inverted pendulum task, with an ideal controller to stand in for the human expert. The results for both the CARLA and the inverted pendulum tasks showed a significant reduction in the training time and a significant increase in gained rewards with discrete feedback, as opposed to continuous feedback, while the action space remained continuous. It was also shown that head direction feedback can be almost as good as steering feedback. We expect our findings to provide a simple yet efficient training method for Deep RL for autonomous driving, utilizing multiple sources of human feedback.

Keywords: deep reinforcement learning; soft actor–critic; continuous actions; discrete action feedback; learning from demonstrations; learning from interventions; autonomous driving; inverted pendulum

Citation: Savari, M.; Choe, Y. Utilizing Human Feedback in Autonomous Driving: Discrete vs. Continuous. *Machines* **2022**, *10*, 609. https://doi.org/10.3390/machines10080609

Academic Editors: Antonios Gasteratos and Ioannis Kostavelis

Received: 30 June 2022
Accepted: 18 July 2022
Published: 26 July 2022

1. Introduction

Deep reinforcement learning (Deep RL) algorithms use either discrete or continuous state–action spaces. Although continuous state–action spaces are suitable for complex and dynamic environments [1,2], they do involve some downsides [3,4], such as a long training time [5–7] a high amount of complexity [8,9], and sample inefficiency [10]. One of the proposed solutions to overcome these issues is to combine deep RL algorithms with learning from demonstration (LfD) [11–15]. The major challenges of LfD are sample complexity and learning efficiency [16,17]. In LfD, human experts should put lots of effort before the training time of the RL algorithm and cover all possible steps in the environment that are impossible in a continuous space [18–20]. Therefore, it needs a large number of data samples, and sometimes the excessive reliance on optimal actions may cause policy divergence [5,21,22].

Learning from intervention (LfI) is a method that requires fewer data samples from the human expert. It is an online method where a human expert would intervene when the agent makes a mistake during task performance [5]. In the LfI method, the human expert's reaction is introduced with a non-negligible delay, and it acts as a blocker for the policy after claiming control over the environment [23]. LfI methods can lead to a safer training time, but due to less exploration, they are slower to converge [24]. In addition, having frequent supervision from the human expert may cause policy confusion, and it can result

in unstable behavior [16]. Therefore, in our proposed work, the actions executed in the environment alternate between those from the human and the policy, just for a specified number of steps. In this way, the human expert can intervene 50% of the time only during a few initial steps, improving sample efficiency and performance.

The soft actor–critic (SAC) is a powerful method capable of handling complex and continuous action spaces [25]. It is more sample efficient and robust to brittleness in convergence than other Deep RL algorithms. Although SAC is an efficient algorithm, it still has the limitation of long training times due to the high complexity of its continuous action space, which makes its application challenging to autonomous driving. On the other hand, discretizing an RL algorithm might provide a shorter training time. However, discretizing a complex environment might cause information loss [26–28]. To address these issues, we investigated whether different human feedback can solve these issues and which type of human feedback can reduce the training time of the SAC algorithm.

First, we considered whether the human head direction, as well as steering, can be regarded as a reliable feedback in autonomous driving environments, inspired by the fact that when a driver faces a curve, they receive rotational and acceleration stimulation and turn the head to the curve direction [29,30]. Therefore, the human head direction may be closely related to the direction of the road curve [30,31]. We then tested the effect of discrete vs. continuous human feedback, while the action space remained continuous. Putting these together, we collected the human head direction feedback as either discrete or continuous signals and combined each with the stable baseline SAC algorithm that operates in a continuous action space. This experiment was performed with the lane-keeping task in the CARLA simulator. Surprisingly, we found that a discrete human head direction gives a better result than its continuous counterpart. We further investigated whether other human feedback, such as steering in continuous and discrete spaces, can cause the same effect in the SAC algorithm for the same task. Here, we observed the same effect, i.e., discrete feedback is more effective than continuous. Furthermore, we found that the human head direction is almost as good as steering feedback. Finally, the approach was tested in the inverted pendulum (OpenAI Gym) environment with an ideal controller to stand in as a human expert, which removes any human individual differences as a factor affecting our results above. We observed the same outcome: discrete feedback is better than continuous feedback.

In this work, we reduced the training time of the SAC algorithm by using discrete actions from the human expert without actually discretizing the action space of the SAC algorithm. The results showed that discrete human feedback in a continuous Deep RL action space such as SAC demonstrates the best impact compared to other investigations and could significantly accelerate the training of this complex algorithm.

2. Related Works

As compared to other works, our contribution is summarized in Table 1. As represented in this table, human demonstration and Deep RL algorithm action space were always matched (that is, discrete vs. continuous). However, in our work, we tested whether a mismatched type of human feedback in a complex continuous action space could be more beneficial than the matching type and whether the human head direction could be potentially used as reliable feedback in autonomous driving applications.

Table 1. Our contribution compared to existing research.

Approaches	Method			Human Feedback		
	Algorithm	Discrete	Continuous	Discrete	Continuous	Type
[32]	SAC	-	✓	-	✓	Steer
[16]	TD3	-	✓	-	✓	Steer
[33]	DNN	-	✓	-	✓	Steer
[34]	DDPG	-	✓	-	✓	Steer
[3]	DQfD	-	✓	-	✓	Steer
[26]	DDPG	-	✓	-	✓	Steer
[35]	NN	-	-	-	✓	Gaze
[36]	NN	-	-	-	-	Eye tracking, heart rate, physiological data
[37]	DQN&DRQN	✓	-	✓	-	Steer
Ours (matching)	SAC	-	✓	-	✓	Steer
Ours (matching)	SAC	-	✓	-	✓	Head
Ours (mismatching)	SAC	-	✓	✓	-	Steer
Ours (mismatching)	SAC	-	✓	✓	-	Head

3. Background

Soft Actor–Critic

A Markov decision process (MDP) [38] is specified as (A, S, p, r, γ). The action space A and state space S were defined to be continuous. The state-transition probability p is the probability of moving from the current state s_t with action a_t to the next state s_{t+1}. The reward function is denoted by r and γ is a discount factor in the range of (0, 1).

The soft actor–critic (SAC) algorithm [25] is an off-policy algorithm with a continuous state–action space that optimizes a stochastic policy. The objective of the policy is to maximize the trade-off between entropy and the sum of accumulated rewards [25]. Utilizing entropy regularization is the main difference or advantage of SAC compared to other RL algorithms.

The sum of the accumulated rewards is calculated by $\sum_t \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r(s_t,a_t)]$ where ρ_π is the state marginals of the trajectory distribution enforced by a policy $\pi(a_t|s_t)$. In addition to accumulated rewards, the SAC policy is seeking to maximize the entropy in each visited state [25]:

$$\pi^* = argmax_\pi \sum_t \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r(s_t,a_t) + \alpha\mathcal{H}(\pi(.|s_t))] \quad (1)$$

where $\mathcal{H}$ is the entropy term and $\alpha > 0$ is the temperature parameter that specified the relative significance of the reward against the entropy to administrate the stochasticity of the optimal policy.

Although, by principle, it is not required to have a distinct Q and V function (since the policy function relates them), there exist three separate functions in SAC because it helps the convergence [25]. These functions can be calculated as follows:
(1) A state-value network V is parameterized by ψ and approximates the soft value function. This network is trained by minimizing the squared residual error:

$$J_v(\psi) = \mathbb{E}_{s_t\sim D}[\frac{1}{2}(V_\psi(s_t) - \mathbb{E}_{a_t\sim\pi_\phi}[Q_\theta(s_t,a_t) - \log\pi_\phi(a_t|s_t)])^2] \quad (2)$$

The aim of the soft value function is to reduce the squared residual difference between the predicted value network and the expected predicted Q-function added to the entropy of the policy π that is computed by a negative log. The parameters of Equation (2) were updated by computing the gradient as below:

$$\hat{\nabla}_\psi J_V(\psi) = \nabla_\psi V_\psi(s_t)(V_\psi(s_t) - Q_\theta(s_t,a_t) + log\pi_\phi(a_t|s_t)) \quad (3)$$

(2) The soft Q-network parameterized by θ and was trained by minimizing the soft Bellman residual error:

$$J_Q(\theta) = \mathbb{E}_{(s_t,a_t)\sim D}\left[\frac{1}{2}(Q_\theta(s_t,a_t) - \hat{Q}(s_t,a_t))^2\right] \tag{4}$$

where

$$\hat{Q}(s_t,a_t) = r(s_t,a_t)\gamma\mathbb{E}_{s_{t+1}\sim\rho}[V_{\bar{\psi}}^{-}(s_{t+1})] \tag{5}$$

and $\bar{\psi}$ is the parameter of the target value function $V_{\bar{\psi}}$ that exponentially moves the mean of value network weights and can make the training more stable [39]. The aim of Equation (4) is to minimize the squared error for all of the state–action pairs from the replay buffer by getting the difference between the predicted Q-function and the current reward added to the discounted expected value of the next step. Again, the optimization of the parameters was computed by stochastic gradients:

$$\hat{\nabla}_\theta J_Q(\theta) = \nabla_\theta Q_\theta(a_t,s_t)(Q_\theta(a_t,s_t) - r(s_t,a_t) - \gamma V_{\bar{\psi}}(s_{t+1})) \tag{6}$$

(3) The last function is policy function π parameterized by ϕ that is trained by minimizing the expected KL divergence [25]:

$$J_\pi(\phi) = \mathbb{E}_{s_t\sim D}\left[D_{KL}\left(\pi_\phi(.|s_t)\middle\|\frac{\exp(Q_\theta(s_t,.))}{Z_\theta(s_t)}\right)\right] \tag{7}$$

The objective here is to make the policy distribution bear a greater resemblance to the Q function exponentiation distribution normalized by function Z. There are several solutions to minimize this objective in which the likelihood ratio gradient estimator was applied. However, this method does not have a backpropagation, and the target density is a neural network (Q-function). Therefore, a differentiable policy sampling process was utilized and called the reparameterization trick by authors [25].

$$a_t = f_\phi(\epsilon_t;s_t) \tag{8}$$

The notation Z is omitted because it does not have any dependency on ϕ, and that epsilon (ϵ_t) is a noise vector polled from a spherical Gaussian distribution. Based on this equation, the objective function can be rewritten as follows:

$$\begin{aligned} J_\pi(\phi) = \mathbb{E}_{s_t\sim D,\epsilon_t\sim N}[log\pi_\phi(f_\phi(\epsilon_t;s_t)|s_t) - \\ Q_\theta(s_t,f_\phi(\epsilon_t;s_t))] \end{aligned} \tag{9}$$

The approximated gradient of this objective is as follows:

$$\begin{aligned} \hat{\nabla}_\phi J_\pi(\phi) = \nabla_\phi log\pi_\phi(a_t|s_t) + (\nabla_{a_t} log\pi_\phi(a_t|s_t) - \\ \nabla_{a_t}Q(s_t,a_t))\nabla_\phi f_\phi(\epsilon_t;s_t)) \end{aligned} \tag{10}$$

4. Method

In our previous work, we proposed an online learning method that is a combination of LFD, LFI, imperfect human demonstration, and SAC [22], where we utilized the human expert feedback to enhance the original algorithm. Note that the "human expert" in our case is simply an average human driver. They are only an expert from the perspective of the learning agent. By this combination, we could significantly reduce the human expert's effort. The action selection in this work was inspired by our earlier method. In this work, the human expert is not collecting data before the RL training time. Instead, during the training time of the RL algorithm, human actions are collected and fed back to the environment and into the replay buffer. We used the SAC reply buffer for both human data and SAC policy actions. Figure 1 shows our procedure for getting real-time human expert demonstration

actions. The human expert and policy perform actions in an alternating fashion in the environment to prevent having frequent supervision by the human expert and eliminate the delay in human reaction. Therefore, actions a_t were generated from the human expert for even states and from the policy for odd states. By performing human action a_t for even state s_t, the related reward r_t was collected, and the next state s_{t+1} observed. Thus, the human expert generated the transition τ(s, a, r, s′) for even states and stored it in the replay buffer of SAC along with transactions of the SAC algorithm. In this method, the human expert is not gathering any data before the training time of the SAC algorithm. We do not have any reward sketching period, the evaluation step is removed, and the human is not blocking the policy by too many intervention actions. Instead, if the policy performs a bad action a_t in state s_t, in the next step (s_{t+1}), the human would generate action a_{t+1} to correct the mistake of the policy as much as possible.

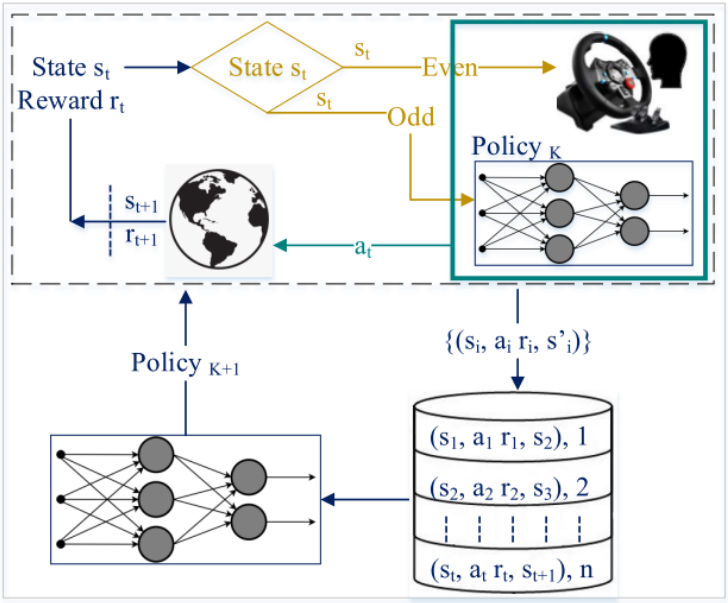

Figure 1. Schematic illustration of our procedure for obtaining human expert demonstration actions.

As mentioned before, one of the aims of this approach is to investigate which type of human feedback (continuous and discrete) can accelerate the training time of a continuous RL algorithm such as SAC. Therefore, we examined the human discrete and continuous feedback with human head direction and steering. For human steer feedback, a Drive Force Steer and a Wii Remote Controller were utilized for collecting the human steering and head direction, respectively. In each of the scenarios, human feedback was merged with SAC actions for approximately 5000 steps (almost 20 episodes); then, the human feedback was cut from the experiment, and the normal process of the SAC algorithm was started.

5. Task 1: Autonomous Driving in CARLA

5.1. Experiment

We tested our approach in the driving task of lane-keeping in the CARLA simulator [40]. The CARLA simulator is a reliable simulator for the development, training, testing, and validation of autonomous driving methods in a safe simulated environment. It is an open source simulator that provides a flexible environment for different driving conditions such as urban layouts, buildings, vehicles, and pedestrians.

The CARLA environment and its top–down view of the road are shown in Figure 2a,b, respectively. In each episode, the car was spawned at a random location on the road. Since defining each RL work's Markov decision process (MDP) is necessary [41], our work's MDP transition is defined below.

Figure 2. (**a**) CARLA simulator environment; and (**b**) a top–down view of the map.

Action space: regarding the SAC algorithm, in our work, a continuous action space from $[-1, 1]$ was defined for steer and from $[0, 1]$ for throttle. We had the same range for continuous and discrete human steer and head direction feedback. We discretized $[-1, 1]$ with 50 equal-sized bins for discrete human steer feedback. Head direction was detected with a Wii remote controller that only generates discrete actions. For continuous human head direction feedback, a noise in the range of bin distance was added to each action. In addition, a speed of 0.2 was added to each generated action from the policy for the acceleration to prevent any extra loading beyond keeping in the lane.

Reward function: The reward function $f(x)$ (Equation (11)) for keeping with the lane for both human feedback and policy actions were calculated by dividing the car speed cs_t over the maximum possible acceleration acc_t minus the minimum possible acceleration [22,42]. This reward function resulted in almost 1.1 in rewards for every single step. In addition, when the car is off the road, a penalty of -100 was given, and the episode was terminated.

$$f(x) = \begin{cases} r_t = \dfrac{cs_t}{\max(acc_t) - \min(acc_t)} \sim 1.1, & \text{on the road} \\ -100, & \text{off the road} \end{cases} \tag{11}$$

State space: The car's front camera took an RGB image for each of the steps in the CARLA simulator. The image was pre-processed within an auto-encoder [42] to eliminate unwanted environmental features, and the encoded image was utilized as the state O_t in this work.

5.2. Results

5.2.1. Results: Continuous Steer Feedback from Human and SAC Algorithm

As discussed previously, one of the experiments combines human continuous steer feedback with the SAC algorithm to discover which type of human feedback can accelerate the training time of a continuous Deep RL algorithm. Figure 3 shows a comparison between the average rewards of episodes for the baseline SAC algorithm (blue line) and continuous human steer feedback used by the SAC algorithm (orange line). The transparent areas in blue and orange color represent the standard deviation of the baseline SAC and our method averaged over ten samples, respectively. The y axis represents the average of rewards over ten randomly selected samples, and the x axis shows the number of episodes. The SAC baseline, on average, could obtain approximately 501.1 in reward over 90 episodes. This number increased to 691.1, when utilizing just 5000 steps (about 20 episodes) of continuous steer feedback from the human expert. The continuous feedback method shows a 37.9% improvement over the baseline SAC algorithm.

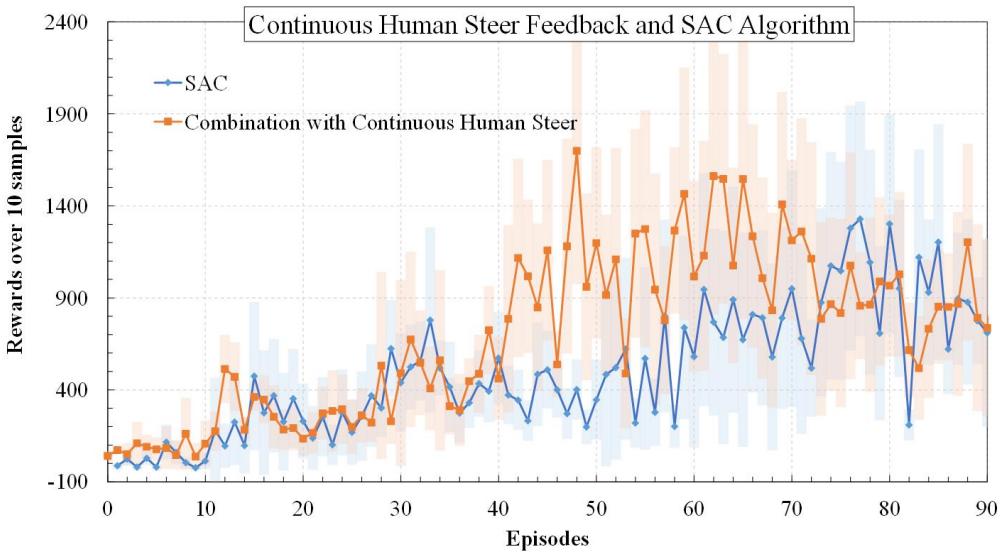

Figure 3. A comparison of baseline SAC with the proposed method engaged with human continuous feedback from steer averaged over 10 samples.

5.2.2. Results: Discrete Steer Feedback from Human and SAC Algorithm

In the second case, discrete human expert feedback by steering was combined with the SAC algorithm. Humans' and SAC actions were employed in the environment in an alternating fashion for approximately 20 episodes (5000 steps). Interestingly, discrete actions from the human expert further reduced the training time of the SAC algorithm and facilitated convergence. Figure 4 shows the average rewards of episodes for ten randomly selected samples for discrete human steer feedback. The blue and orange lines show the average reward gained by SAC and the combination of SAC and discrete steer human feedback, respectively. The average rewards of the baseline SAC algorithm (501.1) were compared with the proposed method (958.1), resulting in a significant (91.1%) improvement in rewards by combining discrete human feedback with SAC. This is a further improvement on the results using continuous human steering feedback, where the improvement in the baseline was only 37.9% (see results above).

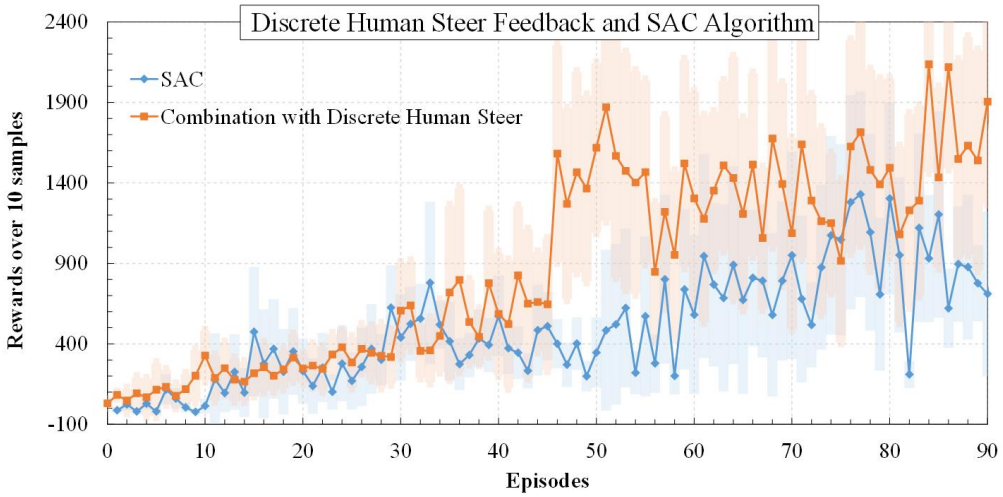

Figure 4. A comparison of baseline SAC with the proposed method engaged with discrete human feedback from steer averaged over 10 samples.

5.2.3. Results: Continuous Human Head Direction Feedback and SAC Algorithm

In order to test whether other forms of human feedback (other than steering) can also improve the baseline SAC, we used human head direction feedback. When the driver decides to turn in a direction in real-world driving, the head turns before the hand. In addition, the human head direction is matched with the curve direction of the road [29,30]. This fact was used to determine whether the human head direction can make the SAC algorithm converge faster. The result of combining human expert head direction (continuous feedback) with SAC is shown in Figure 5. The rewards gained by this method were 553.0, which shows just a 10.3% improvement. Therefore, combining continuous human head direction with SAC only gives a slight edge over the baseline SAC.

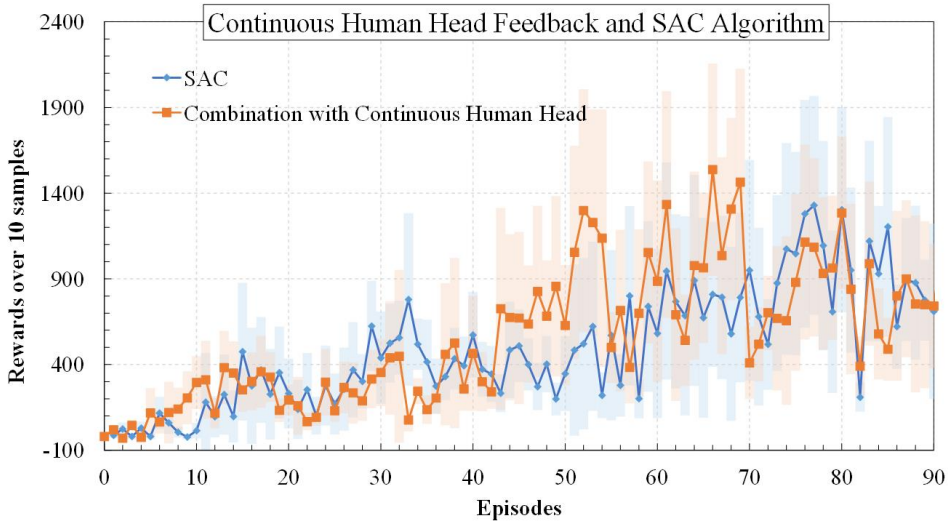

Figure 5. A comparison of baseline SAC with a proposed method engaged with human continuous feedback from the head averaged over 10 samples.

5.2.4. Results: Discrete Human Head Direction Feedback and SAC Algorithm

Figure 6 shows the result of discrete human head direction combined with the SAC algorithm. By employing discrete human feedback from the head direction, a considerable improvement of 62.9% was obtained (816.4 in rewards on average in total) compared to the baseline SAC algorithm.

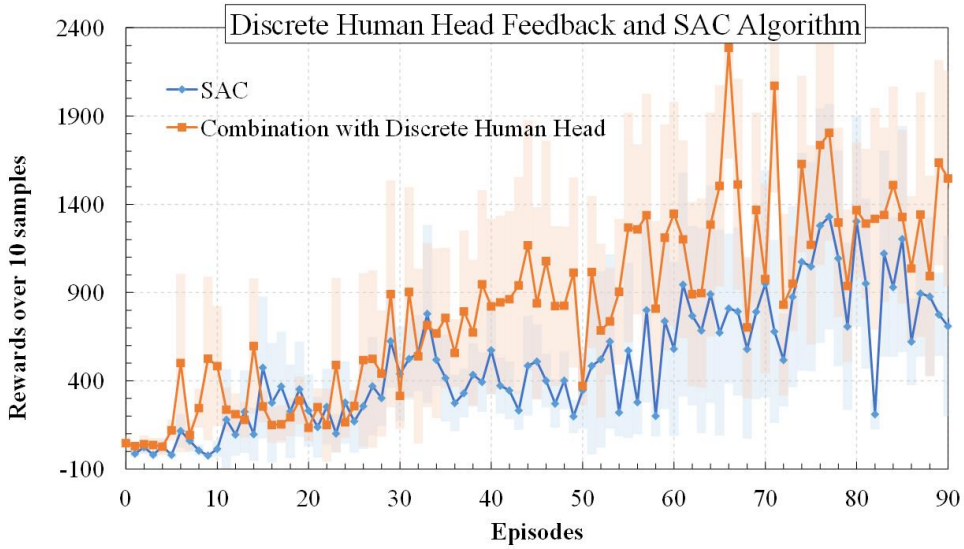

Figure 6. A comparison of the baseline SAC with the proposed method using discrete human feedback from head averaged over 10 samples.

5.3. Results: Behavior

The following presents a sample test case of car behavior on the track during the training time with the baseline SAC and our proposed methods. Figure 7 shows the 50 first episodes of one of the sample studies of the SAC algorithm in the environment. As it is clear from this figure, the car is veering off the road and even could not finish half of the road by the first 50 episodes during the training time of the baseline SAC algorithm. In this sample test case, the first full lap of the track was completed after 70 episodes. All of the episodes were not shown to prevent representing a busy and overcrowded figure with too many repeated actions over each other.

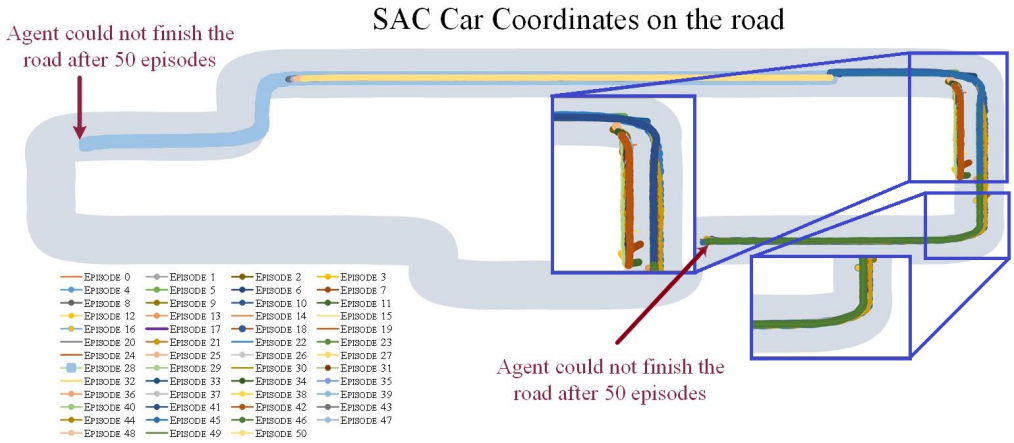

Figure 7. Baseline SAC: the car's trajectory on the roadway for the first 50 episodes of training by the SAC algorithm. Two zoomed-in overlays show details of some busy parts on the road.

By combining the baseline SAC with continuous human steer feedback, the agent could complete a lap (a full trip around the track) by the 48th episode (Figure 8). The human expert trained the agent for the first 19 episodes, and after that, the SAC could complete the lap within 29 episodes, which shows an appropriate improvement over the SAC algorithm.

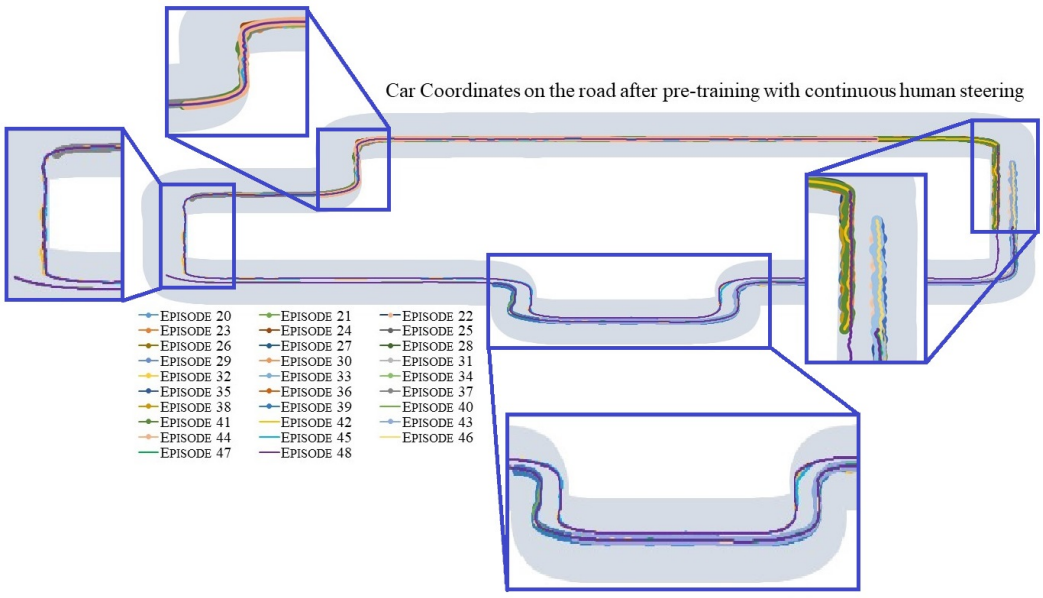

Figure 8. SAC + Continuous Steering Feedback: the car's trajectory on the roadway after pre-training the SAC by 5000 steps of continuous human steer feedback. Four zoomed-in overlays show details of some busy parts of the road.

Figure 9 shows one test case of the training with discrete human steer feedback that shows a significant improvement. The human expert trained the agent for 5000 timesteps (the first 18 episodes). After these 18 episodes, the agent could complete the lap by the 28th episode, which means that the SAC explores the environment just for ten episodes after human training time.

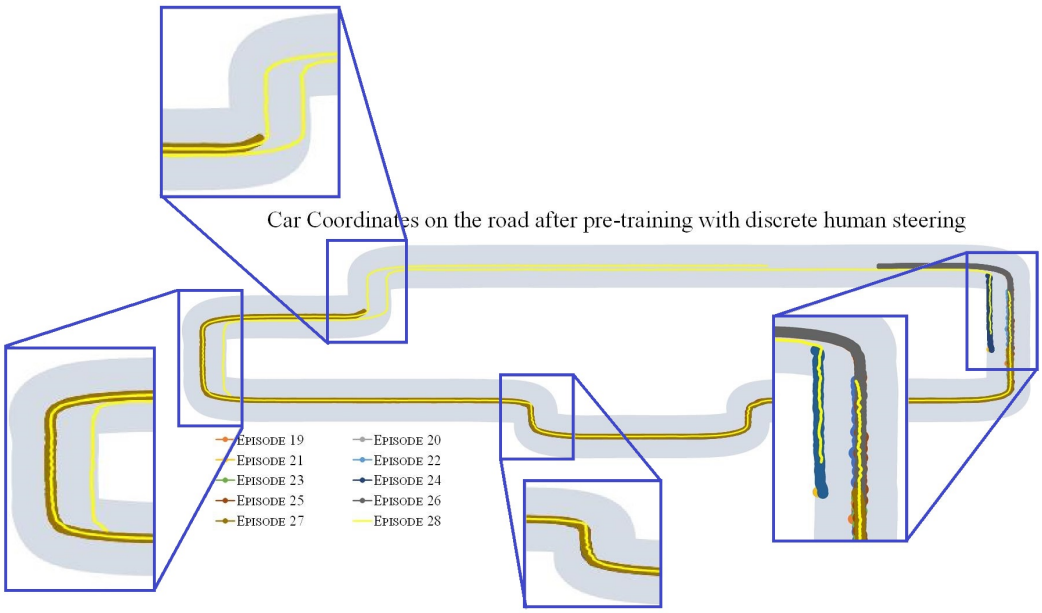

Figure 9. SAC + Discrete Steering Feedback: the car's trajectory on the roadway after pre-training the SAC by 5000 steps of discrete human steer feedback. Four zoomed-in overlays show the details of some busy parts of the road.

Figure 10 shows episodes 19–49 of the car trajectory. During the first 18 episodes, the human expert trained the agent with continuous human head direction. As the figure shows, in this example, the car could complete the lap after episode 49 when trained with continuous human head direction.

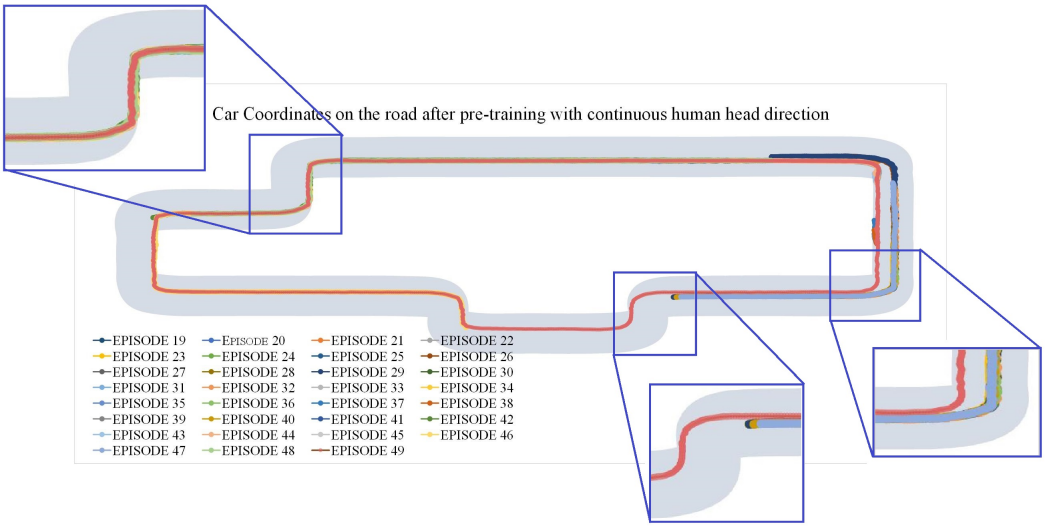

Figure 10. SAC + Continuous Head Direction Feedback: the car's trajectory on the roadway after pre-training the SAC by 5000 steps of continuous human head direction feedback. Three zoomed-in overlays show details of some busy parts of the road.

Figure 11 shows the first 43 episodes of training by discrete human head direction. The human expert trained the agent for 20 episodes and the car could complete the road after 43 episodes.

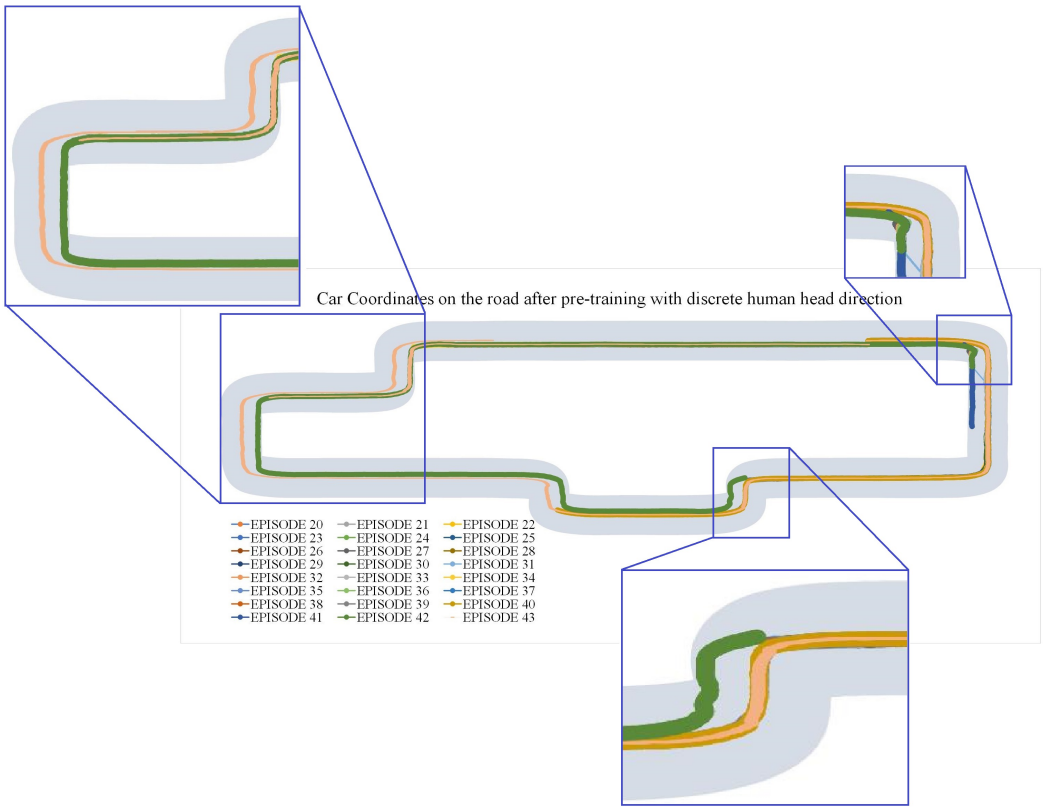

Figure 11. SAC + Discrete Head Direction Feedback: the car's trajectory on the roadway after pre-training the SAC by 5000 steps of discrete human head direction feedback. Three zoomed-in overlays show the details of some busy parts of the road.

6. Task 2: Inverted Pendulum

6.1. Experiment

Humans have variability in their performance and their actions in the environment. In this section, an exact algorithmic expert (an oracle) was used instead of a human expert in the Inverted Pendulum-v2 environment to show the effectiveness of our algorithm under no human variability. Inverted Pendulum-v2 is one of the MuJoCo [43] environments in the OpenAI gym.

6.2. Result

Figure 12 shows the result of combining discrete (green line) and continuous (red line) actions from an algorithmic expert with the SAC algorithm vs. SAC algorithm alone (blue line) in an Inverted Pendulum environment. Continuous actions were converted to discrete by 50 equal-sized bins. The x axis represents the number of episodes, and the y axis represents the number of rewards averaged over ten randomly selected samples. The transparent areas around each line are the standard deviations over the ten samples.

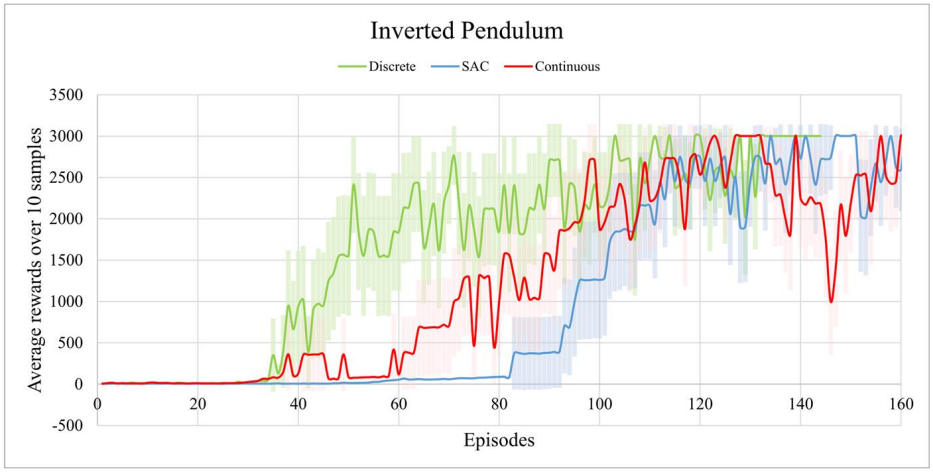

Figure 12. SAC, continuous algorithmic expert feedback, and discrete algorithmic expert feedback examined in openAI gym Inverted Pendulum averaged over 10 samples.

As it is clear from this figure, continuous algorithmic expert feedback improved the performance of the baseline SAC algorithm. The combination of SAC and continuous algorithmic expert earned 1360.1 in rewards which showed a 43.6% improvement compared to the baseline SAC algorithm (earned 946.9 in rewards). On the other hand, the discrete algorithmic expert actions combined with the SAC algorithm could further improve the performance and increase the earned rewards to 1656, which means a 74.9% improvement.

7. Summary of Results

Table 2 summarizes the experimental results for both the autonomous driving task and the Inverted Pendulum task. This table clearly shows that the discrete expert feedback in both the CARLA and the Pendulum environments is better than continuous feedback.

Table 2. Improvement over the baseline SAC by continuous and discrete human expert feedback in the CARLA environment and algorithmic expert feedback in the Pendulum environment.

Type	*CARLA: Steer*	*CARLA: Head*	*Pendulum*
Continuous	37.9%	10.3%	43.6%
Discrete	91.1%	62.9%	74.9%

8. Discussion

Contribution

In this work, we investigated which type of human feedback (discrete vs. continuous) can accelerate the training time of the SAC algorithm and if human head feedback can be used as well as steering for autonomous driving applications. In our experiment, the human expert and policy alternately generate actions in the environment for 5000 steps. We know that frequent supervision by the human expert may cause divergence and overfitting [44]. In addition, to prevent blocking policy for generating actions by the human expert and to eliminate the delay time of human reaction, like the LfI method, the human- and policy-based actions were performed in an alternating fashion in the environment. We compared the six following conditions with this method in the CARLA and the inverted pendulum environment:

- Continuous human steer feedback;
- Discrete human steer feedback;
- Continuous human head direction feedback;
- Discrete human head direction feedback;
- Continuous algorithmic expert feedback;
- Discrete algorithmic expert feedback.

We found that the head direction feedback was almost as effective as the steering feedback, and in general, discrete feedback was better than continuous feedback, regardless of the feedback modality, even when the action space remained continuous.

In summary, the advantages of this work are as follows:

- When a driver faces a curve, they receive the rotational and acceleration stimulation and control the head to the curve direction [29,30]. Therefore, the human head direction is closely related to the direction of the road curve [30,31]. We used human head direction to train the policy without any effort from the human during training.
- This method significantly improved the data efficiency. Therefore, the human expert is not gathering any data samples before the SAC training time, and human effort for training SAC was significantly reduced (5000 steps of human training) compared to other human demonstration methods such as LfD.
- Discrete action–space has a faster training time but is unsuitable for covering all the aspects of a complex environment. In this work, we took advantage of discrete actions to tune the policy faster without changing the action–space of the SAC algorithm to discrete.
- In a LfI method, when the agent performs any mistake in the environment, the human expert intervenes to correct the fault action. Therefore, there exists a non-neglected delay from humans to take control over the policy. This delay was removed in our method since the human and policy alternately generated actions to take control over the policy.
- The training time was significantly reduced, especially when the feedback was discrete.

9. Conclusions

The main contribution of this work is in the investigation of different types of human intervention and feedback effects in combination with the SAC algorithm to make reinforcement learning safer and faster during the training time. The results show that the head direction is almost as reliable as the steering feedback and the discrete feedback performs better than continuous feedback even when action space is continuous. In order, discrete steer human feedback with 91.1% improvement over baseline SAC was the best result, then human discrete head direction feedback with 62.9%, continuous steer human feedback was second with 37.9%, and finally, human continuous head direction feedback with 10.3% had the lowest amount of improvement for the SAC algorithm. This shows that we can take advantage of discrete action feedback to accelerate the training time of the SAC algorithm while keeping the action space itself continuous. Furthermore, we found that human head direction can also serve as a reliable source of human feedback. In future work, we will combine the discrete human head and steer feedback to determine whether the combination of them can further reduce the training time of the SAC algorithm. Another interesting idea in this area is to find the optimum ratio of actions by the human expert vs. the policy in the environment and the optimal discrete bars for human feedback. Finally, we will investigate whether defining the reward function through IRL can accelerate the SAC algorithm's training time compared to the current reward function. In sum, we expect the proposed method in this work to make deep reinforcement learning algorithms more robust in challenging environments such as autonomous driving.

Author Contributions: M.S.: Conceptualization; Formal analysis; Methodology; Coding and data collection; Experiments and Analysis; Validation; Visualization; Writing, review, and editing. Y.C.: Conceptualization; Methodology; Project administration; Resources; Supervision; Validation; Visualization; Review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets generated during and/or analyzed during the current study will be shared upon request, by the first author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Smart, W.D.; Kaelbling, L.P. Practical Reinforcement Learning in Continuous Spaces. 2000. Available online: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.97.9314&rep=rep1&type=pdf (accessed on 17 July 2022).
2. Lange, S.; Riedmiller, M.; Voigtländer, A. Autonomous reinforcement learning on raw visual input data in a real world application. In Proceedings of the 2012 International Joint Conference on Neural Networks (IJCNN), Brisbane, QLD, Australia, 10–15 June 2012; pp. 1–8.
3. Liu, H.; Huang, Z.; Lv, C. Improved deep reinforcement learning with expert demonstrations for urban autonomous driving. *arXiv* **2021**, arXiv:2102.09243.
4. Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; Moritz, P. Trust region policy optimization. In Proceedings of the International Conference on Machine Learning, Lille, France, 7–9 July 2015; pp. 1889–1897.
5. Bi, J.; Dhiman, V.; Xiao, T.; Xu, C. Learning from interventions using hierarchical policies for safe learning. *Proc. AAAI Conf. Artif. Intell.* **2020**, *34*, 10352–10360. [CrossRef]
6. Liu, K.; Wan, Q.; Li, Y.A deep reinforcement learning algorithm with expert demonstrations and supervised loss and its application in autonomous driving. In Proceedings of the 37th Chinese Control Conference (CCC), Wuhan, China, 25–27 July 2018; pp. 2944–2949.
7. Kendall, A.; Hawke, J.; Janz, D.; Mazur, P.; Reda, D.; Allen, J.-M.; Lam, V.-D.; Bewley, A.; Shah, A. Learning to drive in a day. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 8248–8254.
8. Hancock, P.A.; Nourbakhsh, I.; Stewart, J. On the future of transportation in an era of automated and autonomous vehicles. *Proc. Natl. Acad. Sci. USA* **2019**, *116*, 7684–7691. [CrossRef] [PubMed]
9. Wang, J.; Zhang, Q.; Zhao, D.; Chen, Y. Lane change decision-making through deep reinforcement learning with rule-based constraints. In Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 14–19 July 2019; pp. 1–6.
10. Ward, P.N.; Smofsky, A.; Bose, A.J. Improving exploration in soft-actor-critic with normalizing flows policies. *arXiv* **2019**, arXiv:1906.02771
11. Dossa, R.F.J.; Lian, X.; Nomoto, H.; Matsubara, T.; Uehara, K. A human-like agent based on a hybrid of reinforcement and imitation learning. In Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 14–19 July 2019; pp. 1–8.
12. Shin, M.; Kim, J. Adversarial imitation learning via random search. In Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 14–19 July 2019; pp. 1–8.
13. Mnih, V.; Badia, A.P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; pp. 1928–1937.
14. Sallab, A.E.; Abdou, M.; Perot, E.; Yogamani, S. Deep reinforcement learning framework for autonomous driving. *Electron. Imaging* **2017**, *19*, 70–76. [CrossRef]
15. Zhang, X.; Ma, H. Pretraining deep actor-critic reinforcement learning algorithms with expert demonstrations. *arXiv* **2018**, arXiv:1801.10459.
16. Wu, J.; Huang, Z.; Huang, C.; Hu, Z.; Hang, P.; Xing, Y.; Lv, C. Human-in-the-loop deep reinforcement learning with application to autonomous driving. *arXiv* **2021**, arXiv:2104.07246.
17. Gao, Y.; Xu, H.; Lin, J.; Yu, F.; Levine, S.; Darrell, T. Reinforcement learning from imperfect demonstrations. *arXiv* 2018, arXiv:1802.05313.
18. Hussein, A.; Gaber, M.M.; Elyan, E.; Jayne, C. Imitation learning: A survey of learning methods. *ACM Comput. Surv. (CSUR)* **2017**, *50*, 1–35. [CrossRef]
19. Konda, V.R.; Tsitsiklis, J.N. Actor-critic algorithms. In *Advances in Neural Information Processing Systems*; NeurIPS: Lake Tahoe, NV, USA, 2000; pp. 1008–1014. Available online: https://papers.nips.cc/paper/1999/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html (accessed on 17 July 2022).
20. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2015**, arXiv:1509.02971.
21. Codevilla, F.; Santana, E.; López, A.M.; Gaidon, A. Exploring the limitations of behavior cloning for autonomous driving. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 9329–9338.
22. Savari, M.; Choe, Y. Online virtual training in soft actor-critic for autonomous driving. In Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 18–22 July 2021.
23. Saunders, W.; Sastry, G.; Stuhlmueller, A.; Evans, O. Trial without error: Towards safe reinforcement learning via human intervention. *arXiv* **2017**, arXiv:1707.05173.

24. Goecks, V.G.; Gremillion, G.M.; Lawhern, V.J.; Valasek, J.; Waytowich, N.R. Efficiently combining human demonstrations and interventions for safe training of autonomous systems in real-time. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolului, HI, USA, 27 January–1 February 2019; Volume 33, pp. 2462–2470.
25. Haarnoja, T.; Zhou, A.; Abbeel, P.; Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 1861–1870.
26. Chen, S.; Wang, M.; Song, W.; Yang, Y.; Li, Y.; Fu, M. Stabilization approaches for reinforcement learning-based end-to-end autonomous driving. *IEEE Trans. Veh. Technol.* **2020**, *69*, 4740–4750. [CrossRef]
27. Millán, C.; Fernandes, B.J.; Cruz, F. Human feedback in continuous actor-critic reinforcement learning. In Proceedings of the 27th European Symposium on Artificial Neural Networks, Bruges, Belgium, 24–26 April 2019.
28. Hasselt, H.V.; Wiering, M.A. Reinforcement learning in continuous action spaces. In Proceedings of the IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning, Honolulu, HI, USA, 1–5 April 2007.
29. Fujisawa, S.; Wada, T.; Kamiji, N.; Doi, S. Analysis of head tilt strategy of car drivers. In Proceedings of the ICROS-SICE International Joint Conference, Fukuoka, Japan, 18–21 August 2009; pp. 4161–4165.
30. Land, M.F.; Tatler, B.W.Steering with the head: The visual strategy of a racing driver. *Curr. Biol.* **2001**, *11*, 1215–1220. [CrossRef]
31. Braunagel, C.; Kasneci, E.; Stolzmann, W.; Rosenstiel, W. Driver-activity recognition in the context of conditionally autonomous driving. In Proceedings of the 2015 IEEE 18th International Conference on Intelligent Transportation Systems, Gran Canaria, Spain, 15–18 September 2015; pp. 1652–1657.
32. Huang, Z.; Wu, J.; Lv, C. Efficient deep reinforcement learning with imitative expert priors for autonomous driving. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, 1–13. [CrossRef] [PubMed]
33. Pan, Y.; Cheng, C.-A.; Saigol, K.; Lee, K.; Yan, X.; Theodorou, E.; Boots, B. Agile autonomous driving using end-to-end deep imitation learning. *arXiv* **2017**, arXiv:1709.07174.
34. Zuo, S.; Wang, Z.; Zhu, X.; Ou, Y. Continuous reinforcement learning from human demonstrations with integrated experience replay for autonomous driving. In Proceedings of the 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), Macau, China, 5–8 December 2017; pp. 2450–2455.
35. Pal, A.; Mondal, S.; Christensen, H.I. Looking at the right stuff-guided semantic-gaze for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11883–11892.
36. Huang, G.; Liang, N.; Wu, C.; Pitts, B.J. The impact of mind wandering on signal detection, semi-autonomous driving performance, and physiological responses. In Proceedings of the Human Factors and Ergonomics Society Annual Meeting, Seattle, WA, USA, 28 October–2 November 2019; SAGE Publications Sage: Los Angeles, CA, USA, 2019; Volume 63, pp. 2051–2055.
37. Du, Z.; Miao, Q.; Zong, C. Trajectory planning for automated parking systems using deep reinforcement learning. *Int. J. Automot. Technol.* **2020**, *21*, 881–887. [CrossRef]
38. Sutton, R.S. On the significance of markov decision processes. In Proceedings of the International Conference on Artificial Neural Networks, Lausanne, Switzerland, 8–10 October 1997; Springer: Berlin/Heidelberg, Germany, 1997; pp. 273–282.
39. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef] [PubMed]
40. Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A.; Koltun, V. CARLA: An open urban driving simulator. In Proceedings of the 1st Annual Conference on Robot Learning, Mountain View, CA, USA, 13–15 November 2017; pp. 1–16.
41. Palanisamy, P. Multi-agent connected autonomous driving using deep reinforcement learning. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–7.
42. Raffin, A.; Sokolkov, R. Learning to Drive Smoothly in Minutes. 27 January 2019. Available online: https://github.com/araffin/learning-to-drive-in-5-minutes/ (accessed on 17 July 2022).
43. Todorov, E.; Erez, T.; Tassa, Y. Mujoco: A physics engine for model-based control. In Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Algarve, Portugal, 7–12 October 2012; pp. 5026–5033.
44. Wang, D.; Devin, C.; Cai, Q.-Z.; Yu, F.; Darrell, T. Deep object-centric policies for autonomous driving. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 22–24 May 2019; pp. 8853–8859.