Accelerating Reinforcement Learning for Autonomous Driving using Task-Agnostic and Ego-Centric Motion Skills

Tong Zhou, Letian Wang, Ruobing Chen, Wenshuo Wang, Yu Liu

Abstract—Efficient and effective exploration in continuous space is a central problem in applying reinforcement learning (RL) to autonomous driving. Skills learned from expert demonstrations or designed for specific tasks can benefit the exploration, but they are usually costly-collected, unbalanced/sub-optimal, or failing to transfer to diverse tasks. However, human drivers can adapt to varied driving tasks without demonstrations by taking efficient and structural explorations in the entire skill space rather than a limited space with task-specific skills. Inspired by the above fact, we propose an RL algorithm exploring all feasible motion skills instead of a limited set of task-specific and object-centric skills. Without demonstrations, our method can still perform well in diverse tasks. First, we build a task-agnostic and ego-centric (TaEc) motion skill library in a pure motion perspective, which is diverse enough to be reusable in different complex tasks. The motion skills are then encoded into a low-dimension latent skill space, in which RL can do exploration efficiently. Validations in various challenging driving scenarios demonstrate that our proposed method, TaEc-RL, outperforms its counterparts significantly in learning efficiency and task performance.

I. INTRODUCTION

Reinforcement learning (RL) has achieved great success in various domains [1]–[3] by learning via trial-and-error during constant interaction with the environment. However, the learning efficiency will quickly decay in large-dimension environments [4] [5] such as autonomous driving in urban traffic due to un-informed exploration in continuous action space and the sparse and delayed rewards. As a result, existing plain RL (common RL with single step action) algorithms are often data inefficient to necessitate a large amount of experience, and failing in complex tasks or environments.

Toward such learning efficiency and performance concerns, one great attempt is to accelerate RL algorithms via expert demonstrations. Many existing works contribute this direction, such as pre-training policy via imitation learning [6] [7], regularizing and augmenting the reward during RL training [8]–[10], and injecting demonstrations into the replay buffer [11] [12]. Ideally, these methods can overcome exploration challenges and even further surpass the performance of expert policies. However, such expert demonstrations are usually (i) expensive and labor-intensive to collect and annotate if unavailable, (ii) unbalanced in distribution and hardly optimal-guaranteed, and (iii) struggling to transfer to new tasks as the demonstrations are environment-conditioned or task-specific.

Behavioral science reveals that human behavior is in nature temporally extended [13], whose low-level actions should be regarded as results of skill executions rather than a decision space to explore. As in Fig 1(b), human drivers’ sequence of consistent low-level actions are instinctive muscle responses to decided high-level driving intentions, such as overtaking, following, or braking due to avoidance, etc. The temporally-extended skills enables efficient learning with structured exploration and accelerated reward encountering [4]. In comparison, most existing RL algorithms explore in raw actions and result in inconsistent action sequences, which rarely reflect driving intentions. For instance, autonomous vehicles might move in a wiggling way when purely making explorations in the action space, thereby failing to achieve specific tasks such as overtaking a leading car, as shown in Fig. 1(a). As a result, a number of works explicitly exploited skills to encourage structural exploration and accelerate reward encountering. Some works learn or distinguish skills in offline motion dataset [4] [14]. Nevertheless, it is still hard to guarantee that all essential skills are learned or covered in the dataset. Other works manually design delicate skills such as task-specific decision hierarchies or object-centric parameterized primitives [18]–[20]. These methods succeed in certain settings of well-defined tasks such as overtaking one specific car. However, these task-specific and object-centric skill design limits the flexibility and expressiveness of motions, which is particularly indispensable in complex

Fig. 1. RL agents (e.g. autonomous vehicles) taking exploration (a) over the raw action space will results in inconsistent action sequences, or (b) over the skill space can generate a sequence of consistent low-level actions. (c) In dense traffic, autonomous vehicles need to consider the relationships of surrounding vehicles to generate desired motions, which is too complicated to manually design from a task or object view.
environments. Fig. 1(c) illustrates a multi-agent scenario in which an autonomous car needs to navigate in a dense traffic flow. In such multi-agent settings, the relationships of surrounding vehicles need to get integrated into motion generation, which is usually too complicated to design manually. To tackle the above-mentioned limitations, we proposed TaEc-RL (RL with Task-agnostic and Ego-centric motion skills), an RL algorithm exploring all possible motion skills. Our method can perform well in diverse environments and tasks without demonstrations or delicate task designs. First, we design a task-agnostic and ego-centric (TaEc) motion skill library capable of covering all possible dynamic-feasible ego motions. With sufficient expressiveness and flexibility, the skills only need to be defined once and then can be reused in diverse tasks. Also, note that the design of motion skills is effortless as it has been well investigated in sampling-based motion planning communities [21]–[23]. Then, we distill these motion skills into a low-dimensional latent skill space, in which the autonomous vehicle can be trained with RL efficiently and effectively. By learning over all possible motion skills, our method retains the potential to solve diverse, complex tasks like driving in multi-agent settings.

In summary, our contributions are threefold:

- Designing a task-agnostic and ego-centric motion skill library, which is general-purpose to cover diverse motion skills and can be reused across tasks.
- Distilling motion skills into a latent skill space and modifying the RL algorithm to explore in the latent skill space to encourage efficient and effective learning.
- Demonstrating that our method achieves efficient and effective learning for autonomous driving in three challenging dense-traffic scenarios.

II. RELATED WORK

A. State Space Motion Primitive

Given boundary state constrains (such as final position, orientation and velocity), motion primitive methods [22], [24] can find valid trajectories quickly. [24] used Newton’s method to optimize parameterized control signals to generate dynamic-feasible trajectories for vehicles. Moreover, by building a lookup table storing amount of trajectories, the users can directly pick the most suitable one as initial guess or final output. [21], [22] used path-velocity decomposition method to generate spatial curve and speed profile independently and combine them together.

B. Composite Task Learning in Reinforcement Learning

Performing composite and long-term tasks is a very essential ability in reinforcement learning area. [4] learns a prior of action sequence based on current environment states to help exploration in the reinforcement learning task. But it relies heavily on the quality of expert data and will fail in those driving scenarios lacking efficient expert data. [25] proposed unsupervised skill discovery methods through mutual information. But it didn’t make full use of the existing prior knowledge in the driving domain.

C. Reinforcement Learning in Autonomous Driving

Reinforcement Learning is widely used for autonomous driving in complex scenarios [26], [27]. [26] trained an end to end RL policy to compromise feasible spaces with neighbors when there is no enough space to perform lane-change task. [27] proposed a method to combine RL with an interaction planner, to enable negotiating with other agents in narrow lane navigation task. However, these algorithms are not effective for learning diverse skills in cross-scenario, long-term tasks.

Hierarchical Reinforcement Learning methods are proposed to solve cross-scenario problems in dense traffic scenes. [28] used RL to train the high-level policy to decide the reference speed along a fixed path. However, this method is easy to fall in local optimality. [29] manually collected expert data of different driving styles to train an hierarchical framework in near accident scenarios. However, it is expensive and labor-tense to collect expert data and these task-specific data cannot transfer to other tasks easily.
factor. The pipeline of our proposed TaEc-RL is illustrated in Fig. 2. First, a task-agnostic and ego-centric motion skill library $\mathcal{L}$ is designed to cover diverse motion skills, each motion skill $\tau = \{x_t\}_{t=0}^T$ with a fixed horizon $T$. The motion skill is then distilled as a latent skill $z$ in a latent space $\mathcal{Z}$ by a reconstruction process with a motion encoder $q_{\phi}(z|\tau)$ and a skill decoder $q_{\theta}(\tau|z)$. This allows conducting efficient and effective RL in the temporally-extended skill space with a skill policy $\pi_{\theta}(z|s)$. The proposed method will be detailed in the following sections.

A. Ego-Centric Task-Agnostic Motion Skill Library

We first define a task-agnostic and ego-centric (TaEc) motion skill library from a pure ego-motion view. Specifically, we take a sampling-based motion planning approach to generate flexible and diverse motion skills. The TaEc motion skill library generation procedure consists of four steps: 1) path generation - spatial sampling method to specify the shape of the curve; 2) speed profile generation - temporal sampling method to specify the variation of speed given a time horizon; 3) raw trajectory generation - compose paths and velocity profiles together; 4) post-process the raw trajectory to get standard-form skill trajectory. To conveniently catch up, we briefly revisit the procedures as follows.

1) Path Generation: The path is generated by connecting start nodes and terminal nodes in the road. Each node contains three dimensions of longitude, latitude and heading angle. The start node is the origin of ego-vehicle’s coordinate system, and the end nodes are uniformly sampled along longitude, latitude and heading angle dimensions. For each start-end node pair, we use the state-lattice method [22] to generate a path. The diversity of generated paths reproduces spatial motion such as lane-following, lane-change and turn around. Fig. 3(a) is an example of sampling procedure along longitude and latitude (the heading angle is fixed to 0 in this figure) of terminal nodes.

2) Speed Profile Generation: The speed profile can be represented as a third-order polynomial with respect to time.

$$v(t) = q_0 + q_1 \cdot t + q_2 \cdot t^2 + q_3 \cdot t^3,$$  \hspace{1cm} (1)

The polynomial are calculated by the initial speed, acceleration, time horizon and final speed. The variety of speed changes can cover temporal intentions such as accelerating, decelerating and emergent stop. As in Fig. 3(b), we sample the final speeds under a specific horizon given a case that initial speed is fixed.

3) Raw Trajectory Generation: The generated paths and velocity profiles are composed into raw trajectories as [22] to achieve spatial-temporal planning. Each raw trajectory can be represented by a series of vehicle states, $[x_1, x_2, ..., x_T]$, and each vehicle state is a tuple $x_t = (x_t, y_t, \phi_t, v_t) \forall t \in \{1, ..., T\}$, where $x_t$ and $y_t$ are the longitude and latitude position, $\phi_t$ the heading angle and $v_t$ the speed with respect to the ego vehicle’s current coordinates.

The raw trajectory is still gapped from composing the motion skill. One reason is the horizon $T_h$ is so long that the trajectory may contain multiple skills. The other reason is the unbalance of generated trajectory, for example, straight-line drivings occupy a substantial proportion of the trajectory set. As a result, the raw trajectories need to be processed to get TaEc Motion Skill.

4) TaEc Motion Skill Generation: The Post-processing consists of two operations: Slicing and Filtering.

1) Slicing. We adopt a time-based sliding window mechanism to slice the long-horizon trajectories into skills. The long-horizon trajectory with length $T_h$ is mapped into sliding windows with length $T$ and sliding interval $T/2$. The divided piece with horizon $T$ is denoted as a skill trajectory, which contains different skill information at different time steps.

2) Filtering. We then perform a filtering operation by building a lookup table. Generally, the state of the terminal node and the arc length of the trajectory is considered the most representative feature of a skill trajectory. So we discretize the 5-dimensional keys forming a look-up table. All the skill trajectories with similar features will be hashed in the same key. Lastly, each key in the table only keeps one trajectory that best matches all its contents, and the redundant ones will be filtered out.

Finally, the TaEc skill library is approached and can be represented as $\tau = [x_1, x_2, ..., x_T]$. The grid occupancy map of endpoints $x_T$ in TaEc motion skills is shown in Fig. 3(c).

B. Latent Skill Space Distilling

The TaEc skill library is then distilled into a low-dimensional latent skill space to generalize deployments of RL-based autonomous vehicles in complex and varied tasks. As shown in Fig 2, a generative model consisting of a motion encoder $q_{\phi}(z|\tau)$ and a skill decoder $q_{\theta}(\tau|z)$, is proposed. The former compresses trajectory information into low dimension latent variables, and the latter restores the trajectory from the variables. Here, to make sure the decoded
trajectory is kinematically solvable for a vehicle, we propose to use vehicle kinematic model to convert the output of skill decoder into reasonable and reachable trajectories.

1) Skill Embedding: The TaEc motion skill library is supposed to be embed into latent skill space for the RL policy to use, under the effect of motion encoder \( q_m(z|\tau) \) and skill decoder \( q_d(\tau|z) \). The motion encoder \( q_m(z|\tau) \) takes motion skill \( \tau \) as input and outputs the parameters for Gaussian distribution of the latent skill \( z \). One sample from the latent skill distribution represents one abstract behavior. The skill decoder \( q_d(\tau|z) \) reconstructs the motion skill \( \hat{\tau} \) from the sampling results of the latent skill distribution. The training of this model utilizes the following evidence lower bound (ELBO):

\[
\mathbb{E}_{q_m}
\left[
\log q_d(\tau|z) - \zeta \left( \log q_m(z|\tau) - \log p(z) \right)
\right],
\]

with two types of losses in terms of reconstruction and regularization. The reconstruction loss is designed to rebuild the motion skill \( \tau \) from the latent skill \( z \), and the regularization loss is designed to compact the latent space so as to make exploration efficiently. The prior \( p(z) \) is set to be a unit Gaussian \( \mathcal{N}(0, I) \), and the \( \zeta \) balances the two loss terms.

To make the reconstruction temporarily available, we use an LSTM model to encode and decode the motion skill. The encoder interactively inputs vehicle states yielding an embedding at the end. The decoder outputs a sequence of control actions at every time, which are denoted as \([u_1, u_2, ..., u_T]\), given initial state \( x_0 \). They are then used to generate trajectories satisfying kinematic constrains by the vehicle model.

2) Vehicle Model: In the skill decoder, we use the bicycle model as the vehicle kinematic model, described as follows:

\[
\begin{align*}
\dot{x} &= v \cdot \cos(\phi + \beta) \\
\dot{y} &= v \cdot \sin(\phi + \beta) \\
\dot{\phi} &= \frac{v}{l_r} \cdot \sin(\beta) \\
\dot{v} &= a \\
\beta &= \arctan\left( \frac{l_r}{l_r + l_f \cdot tan(\delta)} \right)
\end{align*}
\]

where \( \beta \) is the velocity angle, \( l_r \) and \( l_f \) are the distances of the rear and front tires from the gravity center of the vehicle. The state of the vehicle can be represented as \( x = [x, y, \phi, v] \), and the vehicle control input \( u \) is composed of the forward acceleration \( a \) and steering angle \( \delta \), \( u = [a, \delta] \). The vehicle model is used to convert the given control inputs into trajectory states in skill decoder. Constraints such as the highest speed, maximum acceleration, and steering rate limits are taken into account in the vehicle model to ensure that the trajectory is dynamically feasible.

For each time step \( t \in \{1, 2, ..., T\} \), the vehicle model will propagate one action \( u_t \) at state \( x_{t-1} \) into vehicle model in equation 3 to generate a new state \( x_t \). The procedure will repeat \( T \) times to generate a whole trajectory \( \hat{\tau} = [\hat{x}_1, \hat{x}_2, ..., \hat{x}_T] \), which is the final output of the skill decoder \( q_d(\tau|z) \). The reconstruction loss consists of three parts, including the position error, the velocity error and angle error between the motion skill trajectory \( \tau \) and generated trajectory \( \hat{\tau} \). at each time step \( t \in \{1, 2, ..., T\} \).

In light of this reconstruction, the latent skill space can represent diverse and flexible task-agnostic and ego-centric motion skills. The skill decoder \( p_d(\tau|z) \) will then be fixed and reused to generate future behaviors from a sample of the latent skill space.

C. RL with Exploration in Skill Space

Within the latent skill space \( Z \), the RL agents can conduct structured and temporally-extended exploration to accelerate reward encountering. Fig. 2 illustrates the general idea of our proposed method. Specifically, instead of directly learning a policy over raw actions \( \pi(a|s) \) (here raw action \( a \) is a single control signal \( u_1 \)), we learn a policy that outputs latent skill variables \( \pi_\theta(z|s) \) which is then decoded to motion skill by the fixed skill decoder \( q_d(\tau|z) \). Each motion skill is tracked for a fixed length of \( T \) steps before next skill is sampled. This process follows a typical semi-MDP process with temporal abstraction and succeeds in learning for long-horizon tasks [30]–[33]. The horizon \( T \) in RL is consistent with the skill learning of Section III-B and the motion skill library generation of Section III-A. The perfect execution length might depend on tasks, environments, and circumstances, and many works attempt to learn policies with a flexible length [34]–[36]. In this paper, we empirically found that policies with a fixed length can reach a satisfying performance; hence, the learning for replanning triggering strategy will be as the future work.

To encourage exploration and enhance robustness to disturbances, we take a maximum-entropy RL [37], [38] to train the skill policy \( \pi_\theta(z|s) \) to maximize the objective:

\[
J = \mathbb{E}_\pi \left[ \sum_{i=1}^{N} \gamma^i r(s_i, z_i) + \alpha \mathcal{H}(\pi_\theta(z|s)) \right],
\]

where \( \sum_{i=1}^{N} \gamma^i r(s_i, z_i) \) is the discounted reward returned from the environment after tracking the motion skill for \( N \) decision steps ahead. This term aims to encourage the autonomous vehicle to reach destination with shortest time while penalizing jerks, collisions, and driving out of road. The entropy term \( \mathcal{H}(\pi_\theta(z|s)) \) is designed to encourage exploration by maximizing the negated KL divergence between the latent skill space distribution (i.e., outputs of the skill policy) and a uniform distribution \( U(z) \):

\[
\mathcal{H}(\pi_\theta(z|s)) = -\mathbb{E} \left[ \log \pi_\theta(z|s) \right] \propto -\text{D}_{\text{KL}}(\pi_\theta(z|s), U(z)).
\]

Specifically, we modified Soft Actor-Critic algorithm [39] [40] to implement our idea. The entire learning procedures of TaEc RL is shown in Algorithm 1.

IV. EXPERIMENT

A. Experiment Setting

1) Environment and task: We apply our proposed TaEcRL algorithm to the tasks of autonomous driving and evaluate
its generalizability across diverse environments with different traffic settings using the MetaDrive simulator [41]. As shown in Fig. 4, we consider three common-yet-challenging traffic scenarios: highway, roundabout and intersection. In these scenarios, the ego vehicle must reach its destination within a specified time without collisions or going out of bounds. Especially in highway and roundabout scenarios, to prevent the agent from simply following the leading vehicle without any attempts at overtaking and lane changing, we set a constraint on the longest passing time. We use a 5-channel bird-eye view image as observation tensor with a size of 200 × 200 × 5, as illustrated in Figure 5.

2) Reward and Step Information: At each simulation step, the ego vehicle choose a control input \( u_t \) as action, and the reward can be represented as:

\[
r_t = c_1 \cdot R_{\text{driving}} + c_2 \cdot R_{\text{speed}} + c_3 \cdot R_{\text{termination}} + c_4 \cdot R_{\text{jerk}}
\]

1) \( R_{\text{driving}} \) denotes the longitudinal coordinates of the ego vehicle in the current lane of two consecutive time steps, and encourages agent to move forward.

2) \( R_{\text{speed}} = v_t / v_{\text{max}} \), encourages the car to move fast.

3) \( R_{\text{termination}} \) contains a set of sparse rewards. The reward scheme is positive if the car succeed to drive to destination, negative if the car run out of the road or crash other objects, and zero if the game is not terminal.

4) \( R_{\text{jerk}} \) measures the stability of the car’s motion. If a vehicle does not run smoothly, it will get a negative reward.

Note that in TaEc RL pipeline, one skill step consists of \( T \) simulation steps. Once a skill is chosen, \( T \) actions are executed before sampling the next skill, and the reward will be the summary of \( T \)-step rewards \( \bar{r} = \sum_{t=1}^{T} r_t \).

3) Baselines: We compare the performance of the proposed TaEc-RL with several baselines:

- **PPO**: Train an agent from scratch by Proximal Policy Optimization (PPO) [42].
- **SAC**: Train an agent from scratch with Soft Actor-Critic (SAC) [39]. Along with PPO, they serve as typical RL baseline methods with single-step output for comparison.
- **Flat TaEcRL**: Train an agent with output of single-step skill. This method tests the effect of temporal

---

**Algorithm 1 TaEc RL**

1: **Input**: Motion skill library \( L \), discount \( \gamma \), target divergence \( \delta \), learning rates \( \lambda_\alpha \), \( \lambda_\beta \), \( \lambda_\mu \), target update rate \( m \).
2: Initialize motion encoder \( q_m(z_t|\tau) \), skill decoder \( q_d(\tau|z) \), skill policy \( \pi_\phi(z_t|s_t) \), critic \( Q_\phi(s_t, z_t) \), target network \( \bar{Q}_{\theta}(s_t, z_t) \), replay buffer \( D \).
3: for each iteration do
4:     Sample a skill trajectory \( \tau \) from \( L \) \# Latent Skill Space Distilling
5:     \( z \sim q_m(z_t|\tau); (\{u_i\}_{i=1}^T, \tau) \sim q_d(\tau|z) \)
6:     Update \( q_m \), \( q_d \) according to Equation (2)
7: end for
8: for each iteration \# RL in Latent Skill Space do
9:     for every \( T \) environment step do
10:        \( z_t \sim \pi_\phi(z_t|s_t) \) sample skill latent variable
11:        \( (\{u_i\}_{i=1}^T, \tau) \sim q_d(\tau|z_t) \) \# generate skill
12:        \( s_{t+1} \sim p(s_{t+1}|s_t, u_t, \{u_i\}_{i=1}^T) \) \# state transition
13:        \( \bar{r}(s_t, z_t) = \sum_{i=1}^T r_{t+i} \) \# reward calculation
14:        \( D \leftarrow D \cup \{s_t, z_t, \bar{r}(s_t, z_t), s_{t+1}\} \) \# replay buffer
15: end for
16: for each gradient step do
17:        \( Q = \bar{r}(s_t, z_t) + \gamma \bar{Q}_{\theta}(s_{t+1}, \pi_\phi(z_{t+1}|s_{t+1})) + \alpha \bar{H}(\pi_\phi(z_{t+1}|s_{t+1})) \) \# compute Q-target \# update policy network parameter
18:        \( \theta \leftarrow \theta - \lambda_\alpha \nabla_{\theta} \bar{Q}_{\theta}(s_t, z_t, \pi_\phi(z_t|s_t)) + \alpha \bar{H}(\pi_\phi(z_t|s_t)) \) \# update critic network parameter
19:        \( \phi \leftarrow \phi - \lambda_\beta \nabla_{\phi} (\frac{1}{2} (Q_{\phi}(s_t, z_t) - Q)^2) \) \# update alpha
20:        \( \alpha \leftarrow \alpha - \lambda_\mu \nabla_{\alpha} [\alpha \cdot (\bar{H}(\pi_\phi(z_t|s_t)) - \delta)] \) \# update target network parameter
21:        \( \phi \leftarrow \tau \phi + (1 - m) \phi \)
22: end for
23: return trained policy \( \pi_\phi(z_t|s_t) \) and skill decoder \( q_d(\tau|z) \)

---

Fig. 4. Three scenarios of our experiments listed from top to down: highway, roundabout and intersection. The ego vehicle needs to reach the assigned destination with less time but no collisions or task failures such as driving out of road.
abstraction in skills.

- **Const SAC**: Train an agent with SAC that output a fixed control signal (once decided) during a period of time $T$, the same as the skill horizon of TaEc-RL. This method verifies the necessity of skill diversity and expressiveness with the same decision frequency as TaEc-RL.

4) **Evaluation**: To evaluate the optimal and stable performance, we used two commonly-used metrics:

- **Success rate**: the rate of successfully arriving at the destination within a specified time without collisions and failures;
- **Road completion ratio**: the ratio of road length completed against the whole road length

The skill horizon $T$ for temporally extended methods is set as 10. The traffic densities of Highway and Roundabout are 0.3 and 0.25, respectively. In intersection, we set the highest density with 0.45. To ensure the training insistancy, we deploy all experiments with 3 seeds on a single V100 GPU with 16 CPUs and 150G of memory.

B. **Comparisons**

All results are shown in Fig.6, where the first row shows the road completion ratio, and the second row represents the success rate in relation to the training iterations.

In the intersection scenario, we achieve almost 100% completion and 90% success with only 15k iterations. Const SAC is relatively close to our method in terms of completion, but its success rate is only around 60%. Meanwhile, SAC, Flat TaEc, and PPO are relatively similar in completion. SAC has a success rate of over 20%, whereas the other two algorithms can hardly complete the task.

In the highway and roundabout scenarios, the strict passing time limit causes a generally lower success rate for all algorithms compared to the intersection scenario. Specifically, in the highway scenario, Our algorithm achieves a 90% completion ratio and more than 60% success rate within 20k iterations. SAC achieves a similar completion ratio after 50k iterations, but its success rate is only 5%. Although Const SAC can achieve a 10% success rate, its convergence speed is slow, and its completion ratio and success rate are far inferior to ours. The PPO algorithm has the lowest completion ratio, only reaching 40%, which is the worst among all algorithms. In the roundabout scenario, our algorithm achieves a 90% completion ratio and a 60% success rate within 20k iterations, reaching a peak success rate of 70% at 40k iterations. The performance of Const SAC is slightly worse, taking 50k iterations to achieve an 80% completion ratio and a 40% success rate. SAC and PPO perform similarly on the completion ratio indicator, around 60%, but SAC has a higher success rate of 20%. Flat TaEc-RL can only reach a 20% completion ratio and an almost zero success rate.

Based on the above analysis, it can be concluded that our proposed TaEc-RL algorithm has the fastest convergence in each scenario and achieves optimal performance, outperforming other state-of-the-art methods.

C. **Visualizations and ablation study**

We deploy visualization of motion skill library by sampling evenly in the latent skill space, and draw the distribution of its correlated trajectory end point. As in Fig. 7, diverse motion skills can be reconstructed from the latent skill. Such visualizations also provide interpretability of the latent skill space. The latent skill space is a complete and continuous representation of various motion skills in hyperspace dimensions.

In order to get the best skill horizon and latent dimension, we deploy two sets of comparative experiments. Details, three sets of horizons with length 1, 10, 20 and three sets of latent dimension 2, 5, 10 are tried and shown in Fig. 8. A short skill horizon (Skill Horizon 1) has low success rate because it loses the temporal coherence of a driving skill. As the horizon becomes longer, the convergence speed of the low skill dimension gets slower, which illustrates that longer horizon trajectories need higher skill dimensions to represent.

Also, long skill horizon fails in small latent skill dimensions, due to the limitation of expressiveness of latent space. Finally, latent skill dimension 5, and skill horizon 10 is chosen to have the fastest convergence with the highest success rate.

V. **CONCLUSIONS**

We present TaEc-RL, an RL method over motion skills to solve diverse and complex driving tasks without demonstration. We design Task-agnostic and Ego-centric motion skill library to cover diverse motion skills. The motion skills are distilled into a latent skill space by a reconstruction process. The RL algorithm is modified to explore in the skill space rather than raw action space. Validations on three challenging
Fig. 6. Comparison of our method with other baselines. All experiments are conducted with 3 seeds. The driving difficulty increases from Intersection, to Highway and Roundabout. SAC and PPO represents the plain RL over raw action spaces. Const SAC temporally extends the same action, and Flat TaEc RL exploits the motion skill of a single step. Our methods exploits both temporal abstraction (Const SAC) or motion skills (TaEc-RL) both show better learning efficiency and task performance as the task become more complex, and TaEc-RL outperformed all other methods. The difference between our TaEc-RL and other methods also becomes more obvious as the task gets more difficult.

dense-traffic driving scenarios demonstrate that our TaEc-RL significantly outperforms its counterpart especially when the driving task become more complex.

Fig. 7. Visualization of latent skill space. Fig (a) shows samples in the latent skills with latent dimension size 2 and horizon length 10, together with the ego vehicle’s speed which is also the input of RL policy. Fig (b) visualizes the corresponding reconstructed motion skill end point denoted with the same color in (a).

REFERENCES

[1] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lancot et al., “Mastering the game of go with deep neural networks and tree search,” nature, vol. 529, no. 7587, pp. 484–489, 2016.

[2] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chang, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev et al., “Grandmaster level in starcraft ii using multi-agent reinforcement learning,” Nature, vol. 575, no. 7782, pp. 350–354, 2019.

[3] S. Liu, G. Lever, Z. Wang, J. Merel, Š. Eslami, D. Hennes, W. M. Czarnecki, Y. Tassa, S. Omidshafiei, A. Abolmaleki et al., “From motor control to team play in simulated humanoid football,” arXiv preprint arXiv:2105.12196, 2021.

[4] K. Pertsch, Y. Lee, and J. Lim, “Accelerating reinforcement learning with learned skill priors,” in Conference on robot learning. PMLR, 2021, pp. 188–204.

[5] R. Jia, Q. Li, W. Huang, J. Zhang, and X. Li, “Consistency regularization for ensemble model based reinforcement learning,” in 18th Pacific Rim International Conference on Artificial Intelligence. Springer, 2021, pp. 3–16.

[6] J. Peters and S. Schaal, “Reinforcement learning of motor skills with policy gradients,” Neural networks, vol. 21, no. 4, pp. 682–697, 2008.

[7] Y. Pan, J. Xue, P. Zhang, W. Ouyang, J. Fang, and X. Chen, “Navigation command matching for vision-based autonomous driving,” in 2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2020, pp. 4343–4349.

[8] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, “Overcoming exploration in reinforcement learning with policy gradients,” Neural networks, vol. 21, no. 4, pp. 682–697, 2008.

[9] Y. Zhu, Z. Wang, J. Merel, A. Rusu, T. Erez, S. Cabi, S. Tunsusunakool, J. Kramár, R. Hadsell, N. de Freitas et al., “Reinforcement and imitation learning for diverse visuomotor skills,” arXiv preprint arXiv:1802.09564, 2018.

[10] Y. Wu, M. Mozifian, and F. Shkurti, “Shaping rewards for reinforcement learning with imperfect demonstrations using generative models,” in 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021, pp. 6628–6634.
**Fig. 8.** Ablation analysis of skill length and hidden dimension in Highway scenario without strict passing time limit. Different horizon scenarios (length 1, 10, 20) are listed from top to down. At each horizon scenario, three sets of latent dimension (2, 5, 10) are compared with respect to road completion ratio and success rate.

[12] M. Vecerik, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. Riedmiller, “Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards,” *arXiv preprint arXiv:1707.08817*, 2017.

[13] J. Jing, E. C. Cropper, I. Hurvitz, and K. R. Weiss, “The construction of movement with behavior-specific and behavior-independent modules,” *Journal of Neuroscience*, vol. 24, no. 28, pp. 6315–6325, 2004.

[14] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet, “Learning latent plans from play,” in *Conference on robot learning*. PMLR, 2020, pp. 1113–1132.

[15] L. Wang, Y. Hu, L. Sun, W. Zhan, M. Tomizuka, and C. Liu, “Transferable and adaptable driving behavior prediction,” *arXiv preprint arXiv:2202.05140*, 2022.

[16] J. Merel, Y. Tassa, D. TB, S. Srinivasan, J. Lemmon, Z. Wang, G. Wayne, and N. Heess, “Learning human behaviors from motion capture by adversarial imitation,” *arXiv preprint arXiv:1707.02201*, 2017.

[17] L. Wang, Y. Hu, L. Sun, W. Zhan, M. Tomizuka, and C. Liu, “Hierarchical adaptable and transferable networks (hatn) for driving behavior prediction,” *arXiv preprint arXiv:2111.00788*, 2021.

[18] N. Deo, A. Rangesh, and M. M. Trivedi, “How would surround vehicles move? a unified framework for maneuver classification and motion prediction,” *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 2, pp. 129–140, 2018.

[19] M. Dalal, D. Pathak, and R. R. Salakhutdinov, “Accelerating robotic reinforcement learning via parameterized action primitives,” *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[20] Z. Wang, C. R. Garrett, L. P. Kaelbling, and T. Lozano-Pérez, “Learning compositional models of robot skills for task and motion planning,” *The International Journal of Robotics Research*, vol. 40, no. 6-7, pp. 866–894, 2021.

[21] Z. Li, W. Zhan, L. Sun, C.-Y. Chan, and M. Tomizuka, “Adaptive sampling-based motion planning with a non-conservatively defensive strategy for autonomous driving,” in *The 21st IFAC World Congress*, 2020.

[22] T. Gu, “Improved trajectory planning for on-road self-driving vehicles via combined graph search, optimization & topology analysis,” Ph.D. dissertation, Carnegie Mellon University, 2017.

[23] L. Wang, L. Sun, M. Tomizuka, and W. Zhan, “Socially-compatible behavior design of autonomous vehicles with verification on real human data,” *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3421–3428, 2021.

[24] T. M. Howard and A. Kelly, “Optimal rough terrain trajectory generation for wheeled mobile robots,” *The International Journal of Robotics Research*, vol. 26, no. 2, pp. 141–166, 2007.

[25] B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine, “Diversity is all you need: Learning skills without a reward function,” *arXiv preprint arXiv:1802.06070*, 2018.

[26] D. M. Saxena, S. Bae, A. Nakhæi, K. Fujimura, and M. Likhachev, “Driving in dense traffic with model-free reinforcement learning,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 5385–5392.

[27] Z. Wang, Y. Zhuang, Q. Gu, D. Chen, H. Zhang, and W. Liu, “Reinforcement learning based negotiation-aware motion planning of autonomous vehicles,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 4532–4537.

[28] B. Brito, A. Agarwal, and J. Alonso-Mora, “Learning interaction-aware guidance policies for motion planning in dense traffic scenarios,” *arXiv preprint arXiv:2107.04538*, 2021.

[29] Z. Cao, E. Biyik, W. Z. Wang, A. Raventos, A. Gaidon, G. Rosman, and D. Sadigh, “Reinforcement learning based control of imitative policies for near-accident driving,” in *Proceedings of Robotics: Science and Systems (RSS)*, July 2020.

[30] J. Merel, L. Hasenclever, A. Galashov, A. Ahuja, V. Pham, G. Wayne, Y. W. Teh, and N. Heess, “Neural probabilistic motor primitives for humanoid control,” *arXiv preprint arXiv:1811.11711*, 2018.

[31] J. Merel, S. Tunyasuvunakool, A. Ahuja, Y. Tassa, L. Hasenclever, V. Pham, T. Erez, G. Wayne, and N. Heess, “Catch & carry: reusable neural controllers for vision-guided whole-body tasks,” *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 39–1, 2020.

[32] R. S. Sutton, D. Precup, and S. Singh, “Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning,” *Artificial intelligence*, vol. 112, no. 1-2, pp. 181–211, 1999.

[33] P.-L. Bacon, J. Harb, and D. Precup, “The option-critic architecture,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[34] K. Pertsch, O. Rybkin, J. Yang, S. Zhou, K. Derpanis, K. Daniilidis, T. M. Howard, and P. Sermanet, “Learning latent plans from play,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 6-7, pp. 866–894, 2021.

[35] Z. Cao, E. Biyik, W. Z. Wang, A. Raventos, A. Gaidon, G. Rosman, and D. Sadigh, “Reinforcement learning based control of imitative policies for near-accident driving,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[36] J. Merel, L. Hasenclever, A. Galashov, A. Ahuja, V. Pham, G. Wayne, Y. W. Teh, and N. Heess, “Neural probabilistic motor primitives for humanoid control,” *arXiv preprint arXiv:1811.11711*, 2018.

[37] J. Merel, S. Tunyasuvunakool, A. Ahuja, Y. Tassa, L. Hasenclever, V. Pham, T. Erez, G. Wayne, and N. Heess, “Catch & carry: reusable neural controllers for vision-guided whole-body tasks,” *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 39–1, 2020.

[38] R. S. Sutton, D. Precup, and S. Singh, “Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning,” *Artificial intelligence*, vol. 112, no. 1-2, pp. 181–211, 1999.

[39] P.-L. Bacon, J. Harb, and D. Precup, “The option-critic architecture,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[40] K. Pertsch, O. Rybkin, J. Yang, S. Zhou, K. Derpanis, K. Daniilidis, T. M. Howard, and P. Sermanet, “Learning latent plans from play,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 6-7, pp. 866–894, 2021.

[41] J. Merel, L. Hasenclever, A. Galashov, A. Ahuja, V. Pham, G. Wayne, Y. W. Teh, and N. Heess, “Neural probabilistic motor primitives for humanoid control,” *arXiv preprint arXiv:1811.11711*, 2018.

[42] J. Merel, L. Hasenclever, A. Galashov, A. Ahuja, V. Pham, G. Wayne, Y. W. Teh, and N. Heess, “Neural probabilistic motor primitives for humanoid control,” *arXiv preprint arXiv:1811.11711*, 2018.