Learning to Drive using Waypoints

Tanmay Agarwal*, Hitesh Arora*, Tanvir Parhar*, Shubhankar Deshpande, Jeff Schneider
Robotics Institute, CMU

Motivation

❖ **Traditional Autonomous Driving Pipeline:** Highly modularized with different subsystems for localization, perception, actor prediction, planning & control.

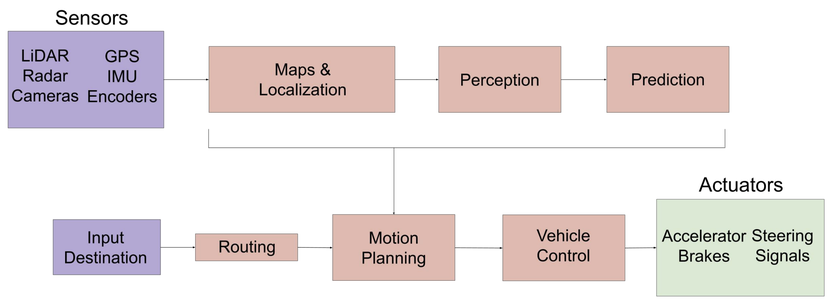

Fig 1: Traditional Autonomous Driving Pipeline

❖ **Challenges:**
- ➢ Generalizability to newer environments.
- ➢ Hand-engineering of numerous parameters.

❖ **Proposed Solution:**
- ➢ Deep Reinforcement Learning for autonomous driving.
- ➢ Potential generalizability to unseen scenarios enabling scalability with reduced engineering efforts.

CARLA Simulator

❖ **CARLA: Open source urban driving simulator for autonomous driving research**
- ➢ Diverse sensor suite, various environmental conditions, configurable static/dynamic actors with maps generation.
- ➢ Sensor suite comprises of LIDAR, RGB camera, semantic camera, depth sensors and GPS.

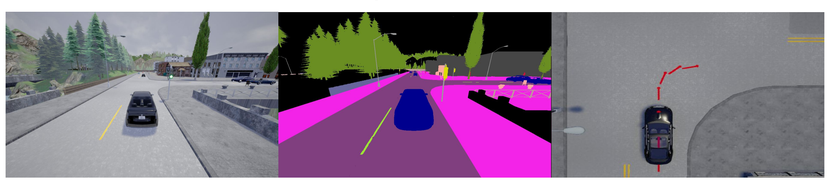

*Fig 2:**Left:** RGB Image, **Middle:** Semantic Image, **Right:** Waypoints*

RL Formulation

Model-free on-policy RL formulation using Proximal Policy Optimization (PPO) [3] algorithm

❖ Input observations:
- ➢ Latent Representation of top-down semantically segmented (SS) Image:

$$\tilde{\mathbf{h}} = g(\mathbf{SS}_{\text{image}})$$

- ➢ Waypoint Features computed using agent's current pose and next n waypoints:

$$\tilde{\mathbf{w}} = f(\mathbf{p}, \mathbf{w_1}, \mathbf{w_2}, ..., \mathbf{w_n})$$

❖ Reward function components:

$$R = R_s + R_d + \mathbf{I}(c) * R_c$$

- ➢ Speed reward

$$R_s = \alpha * \mathrm{u}$$

- ➢ Trajectory distance penalty

$$R_d = -\beta * \mathrm{d}$$

- ➢ Collision penalty

$$R_c = -\gamma * \mathrm{u} - \delta$$

❖ Output Actions
- ➢ *Steer*
- ➢ *Target Speed*
- ➢ For better stability, we use PID controller that outputs *throttle* & *brake* given current speed & target speed.

Experimental Setup

❖ 4 increasingly difficult driving tasks: (a) Straight (b) One-Turn (c) Navigation (d) Navigation with dynamic obstacles
❖ 25 goal-directed scenarios for each of the tasks.
❖ Training is performed in Town 01 & testing in Town02.
❖ Pre-train convolutional auto-encoder (AE) for learning latent representation of SS image.
❖ Finetune AE to learn better input representation that aids in learning a better policy.
❖ Train policy network and AE simultaneously.

Model Architecture

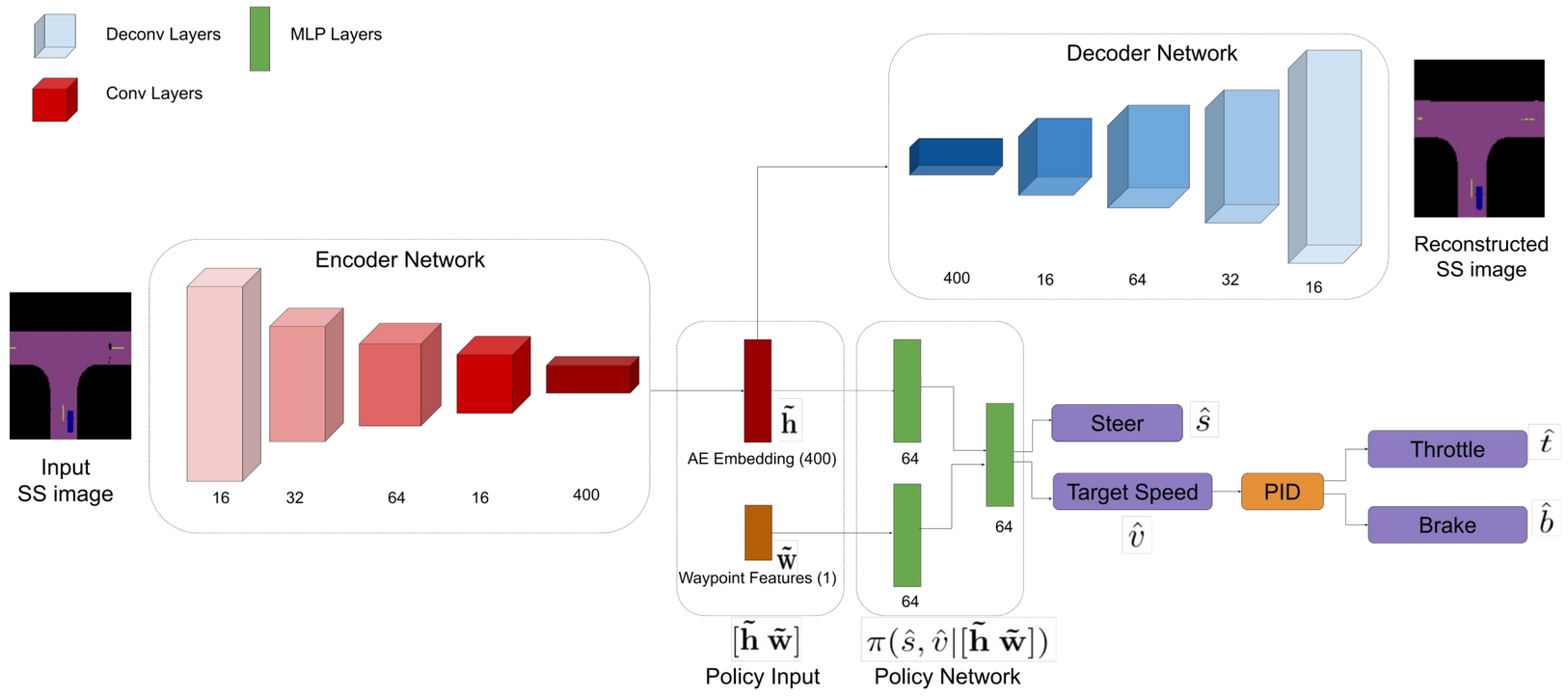

Fig 3: Our Proposed architecture block diagram

Our proposed architecture & its two variants:

- ➢ **WRL:** PID controller outputs only throttle to control speed; Pretrain task (d) with task (c); Collision penalty set to zero.
- ➢ **WRL+:** PID controller outputs both throttle & brake; Pretrain on a simple scenario to learn how to brake; Frameskip of 10; Scale the network output.

Results & Discussion

❖ **Baselines**: CARLA RL [1], Controllable Imitative Reinforcement Learning (CIRL) [2]
❖ **Input Differences**: RGB images v/s semantic segmented (SS) images, high level navigation features v/s low level waypoint features.
❖ **Benchmark**: CARLA benchmark with more stringent and realistic evaluation to terminate episode on collision.
❖ **Results:**
- ➢ Significant improvement in performance on all tasks compared to CARLA RL [1].
- ➢ Even though the CIRL baseline [2] has an advantage of pre-training using imitation learning on expert data, our approach achieves similar performance on training from scratch.

Task	Training Conditions (Town 01)				New Town (Town 02)				New Weather (Town 01)				New Town/New Weather (Town 02)			
	CARLA	CIRL	WRL	WRL+	CARLA	CIRL	WRL	WRL+	CARLA	CIRL	WRL	WRL+	CARLA	CIRL	WRL	WRL+
Straight	89	98	**100**	**100**	74	100	**100**	**100**	86	100	**100**	**100**	68	98	**100**	**100**
One Turn	34	97	**100**	99	12	71	**100**	99	16	94	**100**	99	20	82	**100**	99
Navigation	14	93	**99**	**99**	3	53	**97**	94	2	86	**99**	**99**	6	68	**97**	94
Navigation Dynamic	7	**82**	65	79	2	41	46	**60**	2	**80**	65	79	4	**62**	46	60

Table 1: Quantitative comparison with state-of-the-art approaches on CARLA benchmark. The table reports percentage (%) of successfully completed episodes in each task. The reported approaches are CARLA RL baseline (CARLA) [1], CIRL [2], and our waypoint based DRL variants WRL & WRL+.

Task (c): Navigation (WRL)

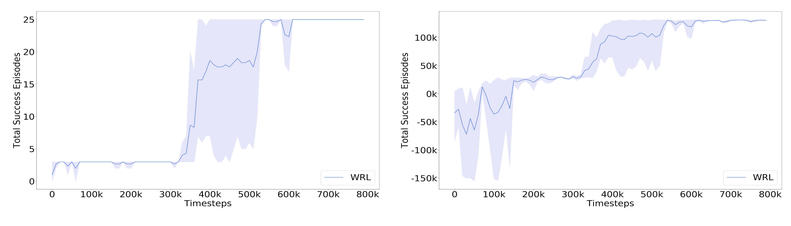

Fig 4: Total Success Episodes and Total Rewards v/s Timesteps for WRL in Task (c) (Navigation).

Task (d): Navigation with Dynamic Obstacles

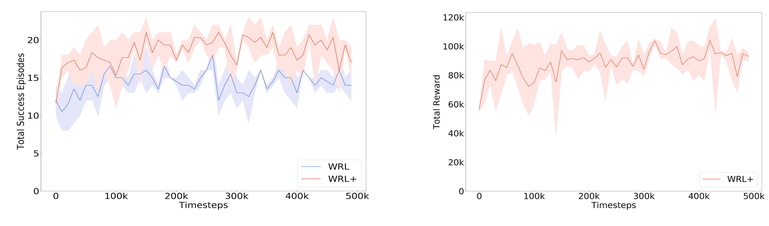

Fig 5: Total Success Episodes v/s Timesteps for WRL & WRL+ in Task (d) (Navigation with dynamic obstacles).

Fig 6: Total Rewards v/s Timesteps for WRL+ in Task (d) (Navigation with dynamic obstacles).

Note: The shaded region corresponds to the minimum and maximum values showing variation across 3 runs.

Future Work

❖ Comprehensive ablation study to analyze the effect of each component change in WRL+ that improved its performance compared to WRL.
❖ Learn better state representations to encode other dynamic actors intent to perform better with dynamic actors.
❖ Compare across other model-free RL algorithms like SAC, DDPG, TD3.
❖ Develop approaches to improve the sample efficiency of the current model-free RL algorithms.
❖ Explore meta-reinforcement learning algorithms to further improve sample efficiency.

References

[1] Alexey Dosovitskiy, Germán Ros, Felipe Codevilla, Antonio López, and Vladlen Koltun. Carla: An open urban driving simulator. In *CoRL,* 2017.

[2] Xiaodan Liang, Tairui Wang, Luona Yang, and Eric P. Xing. Cirl: Controllable imitative reinforcement learning for vision-based self-driving. In *ECCV,* 2018.

[3] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.