Abstraction Augmentation for Deep Reinforcement Learning

Abstract: Human brain tends to neglect most details in visual signals and focus only on salient parts at one time[1], this bottom up approach abstracted the high dimensional continuous 3D world into local topological representation[2], which grants realtime performance and powerful domain adaption for high level decision makers. Prior approaches directly feed the semantic representations into learning process and often suffers from learning inefficiencies due to the local ambiguity (e.g. in 3D navigation, large portion of image are overlapped in consecutive frames). In this work, we present an approach to overcome this problem by compress the segmentation representation into attention representation and combine it with inverse depth map to build up the attention map to actively guide agent's geometrical attention weights during training process. We trained our model with domain randomization, and tested the learning efficiency in five different weather conditions, and further evaluated the design choices in a series of ablation studies.

Keywords: Reinforcement Learning, Autonomous Driving, Image Segmentation

1 Introduction

human mind has developed all kinds of domestication to process different stage of data[3], and all those specialization processors will filter out the unrelated information and offer the higher decision maker or policy learner concise abstractions. To that end, A modular and abstraction[4] based hierarchical learning[5] methods tends to be more biologically plausible than primarily enforce reinforcement learning with deep networks. Primate visual system which enforces more than four layers of visual cortex to solve scale, translation, rotation and even dynamic motion problems. With the help of several edge amplifier, and pattern receptive fields, brain can achieve real-time object recognition and visual control loop. All those structured organizations work as a whole pipeline to power a stereo vision system with super high resolution. As consensus of Rasmussen [6], Frankel and Bedworth [7]: High-level decision making deserves it's highly abstracted state-spaces.

Well trained deep covlutional neural network demonstrated similar structure to receptive fields that can hierarchically percept edges and shapes[8], thus can be analogy to bottom-up layers in visual system, thus combined with "top-down" reinforcement learning modules will theoretically achieve autonomy. However, the large parameter scale requires huge amount of training data, even worse when coupled with reinforcement learning. A few methods has achieved very high data efficiency by: a) reduce data complexity. [9]; b) increase learning ability (find exploration strategies like multi-agent learning, evolution methods, and introduce fast-slow memories to capture long-short term rewards. [10]); c) Find better optimization methods (Schulman et al. [11] and Henderson et al. [12]). Yet we choose to modularize the perception parts and learning parts and decouple the learning from perception to overcome domain shifts (by just add another well-trained perception module, the learnt controller can still work off-the-shelf). Auxiliary LSTM was employed in our learning agent to capture the geometrical dynamic for action predictions, since the recurrent structure are suitable for series data to capture their dynamics through a sliding window.

In our experiments, these components above are still not enough for an agent efficiently learn to drive in dynamic environments. And we introduced attention layer to further help the learner find where to look at. The attention model are well discussed by Xu et al. [13], they introduced a recurrent structure to inhibit the CNN feature extractions where only part of the features are used in the classification task. However, we seek another computational efficient way by building it from the segmentation inferences (Fig 3). Since human attentions lies on the boundaries of each segmentation

3rd Conference on Robot Learning (CoRL 2019), Osaka, Japan.

objects[14], we extract the attention map from segmentation boundries and assign them with high weight when they are geometrically closer to camera. Similarly as Navalpakkam and Itti [1], by combing bottom-up (pre-processing offer the learning agent highly abstracted weighted map) with top-down (learning agent control which interest region to be focus) methods, our agent can notice the small objects yet still keep the global course from shifted (see ablation results in Fig 6).

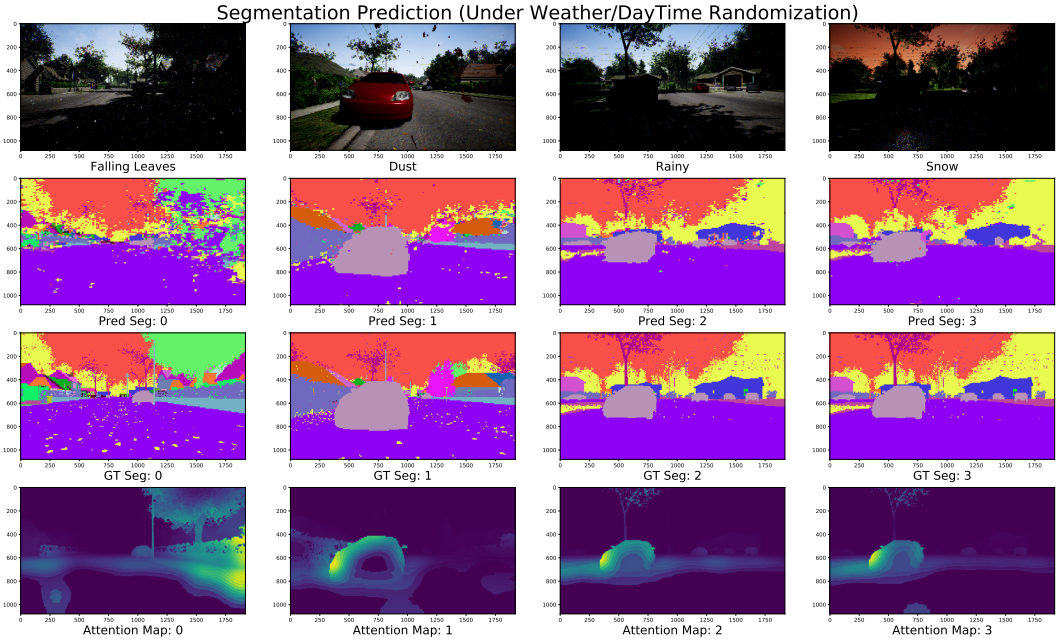

Figure 1: Segmentation module decoupled the biases from shadow and brightness. Last row is attention map for each color image above.

By taking advantages of learning from simulation (we perform our learning tasks in Airsim[15]), we augmented training dataset with permutation of weather and lighting condition, Figure 2 shows that our prediction of segmentation layer remains same with some trivial discrepancies. By applying the attention map with depth map, we can filter out most unnecessary details very close or very far from current state, otherwise the recurrent neural network will take much information in sky which is less useful than the information around vanishing points. And this abstraction augmentation pipeline is also biological plausible: four cortex layers [16] in brain visual system offers pre-processings over edge, orientation, scale, depth and even attention based low level object tracking[17]. Despite deep architectures are able to learn the filter of simple structures, locations, their learnt filters are always constrained by training data, which again, limited it's flexibility and converging speed due to non-trivial couplings that originate either from the network structure itself or from the task[4] [18].

Recurrent Neural Network for series prediction with advantage of memorizing sequence patterns in a sliding window is extremely useful for autonomous driving for getting geometry awareness[19] and handling partial observable scenarios[20]. Hausknecht and Stone [21] applied their DRQN into POMDP games like Pong, where they show the advantage of adding recurrence. Consequently, Apply the recurrent structures inside autonomous driving domain is reasonable since the driving problem itself is unable to scale into MDP horizon. However, for testing of algorithms like DQN which runns in MDP, when trained in simulator, we can still discretely approximate the environment with the help of semantic abstraction into finite horizon.

Our contributions are: a) Proved that abstraction for reinforcement learning out-performed traditional end to end learning methods. b) Proposed a novel attention model and applied into semantic map to achieve higher learning efficiency and robustness.

2 Related Works

Despite deep networks with very large scale of feature maps can memorize many features (e.g. locations, color patterns, and even light effects), when all of the conditions permuted, the combination explosion[22] will soon exhaust the learning machine and introduce the catastrophical forgetting[23] as another pain. People have proposed two popular ways to solve this problem: one is learn by forgetting[24], another is learn without forgetting, by not forgetting, Kirkpatrick et al. [23] proposed to add new learning filters in the joint training. Yet with the learning goes on, tasks and their combinations will exponentially scale up the network complexity that will run out of space before find global optimals. Thrun and Pratt [25] thus took a step ahead and proposed life-long learning. And reinforcement learning is now the golden key.

Reinforcement learning methods has been extensively applied in low-dimensional primitive control policy learning tasks. Deep learning extends the horizon towards high dimensional control problems and achieved remarkable success. Yet three major challenges are still puzzling the community: cursing of the objective specification, hunger of data and lack of domain adaption. methods like policy search and policy gradient are appealing towards robustness and scalability in objective specification, the lack of data efficiency makes them unsuitable to train on robots directly; maximum entropy inverse reinforcement learning is efficient for real robot by decomposition of task demonstration into hierarchies, but the assumptions on dynamic priors increased the difficulty to adapt and transfer; despite model-agnostic meta-learning allow agents to handle domain shift, prohibitively large training data is potentially required by observation of optmizee's data set and it's corresponding weights. In visual domain adaptation domain, Yu et al. [26] achieved one-shot learning with large meta-learning prior libraries learnt from videos and transferred into real-robot with diverse environment. Zhu et al. [27] bridge the reality gap by domain randomization in simulator.

Learn in simulator and scale to real-world thus became a default option for robot learning. Both indoor[28] [27] and outdoor [29] [30] learning are enforced by the synthetic dataset generated in simulators. Simulator also facilitates the learning and transfer process: Müller et al. [4] enforced the segmentation to train a policy network on PID controller, Zhang et al. [31] perform segmentation adaptation by aligning label distributions from simulator absolute ground truth and applied both globally and super-pixel-wisely in target image. The mighty ground truth of segmentation, depth and even absolute location annihilated learning errors from real-world dataset [32]. Sadeghi and Levine [28] augmented their learning policy by domain randomization, and performed validation over indoor corridor with similar geometrical structure from training dataset. Trying to learn from handful dataset yet still capable of complete complex tasks requires both data-efficient and well-defined learning model. Instead of trying off policy correction by sacrificing speed[5] and train the learning end-to-end, one can try simplify the learning model by isolate the perception end and augment it's abstracted observations off the learning phase. So, our method then leverages ground truth of semantic segmentation and depth map to decouple the visual effects from learning agent's encoder, thus achieved high data efficiency while kept the ability for domain adaptation.

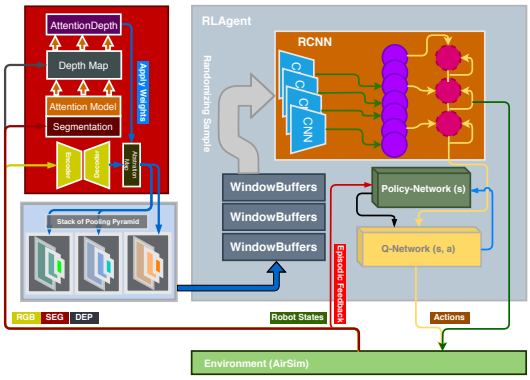

Figure 2: Model overview. The key pipeline is the extraction of attention and apply back to segmentation results before it feed into learning agent.

Reinforcement Learning also boosted Autonomous Driving with its unsupervised fashion. Since deep reinforcement learning with convolutional network and recurrent structures has advanced the robot learning into tackling high-dimensional space and continuous control problems. Kendall et al. [33] has introduced a synthetic network structure with RL backend that takes only images and control signals and trained in a fairly short time. Chen et al. [34] applied RL to run mobile robots in crowded spaces, Wang et al. [35] trained DDPG for autonomous driving policy in TORCS (an open-source 3D car racing simulator). by customizing the action and reward function to suit the

simulator. Shalev-Shwartz et al. [36] modeled the autonomous driving problem within non-MDP settings and reduced the variance of gradient estimation by option graph which can be regard as an temporal abstraction. Other applications[37] [38] [39] including: predict control policies with Recurrent Neural Network, use enforcement learning to solve crossing section in autonomous driving rather than whole pipeline, and the use state-action representations.

3 Methods

Our goal is to learn driving action policies from sequences of rgb images in diverse and dynamic environments with randomness. Colored images are digested by our segmentation networks and weighted by depth and attention map driven on-frame from our attention model. With the witness of advance on segmentation and depth inference with deep structures in a monocular image basis, we can expand our horizon toward enslave those precise lower level perception tools and combined with hierarchical higher level reinforcement learning agents to solve many real-world problems. Inspired by the layered abstraction of human visual perception system, we propose the following biological plausible and computational efficient learning pipeline.

3.1 Abstraction Model

We built our model upon pre-trained VGG16 and implement U-Net segmentation network by minimizing the energy function (the energy function is computed by a pixel-wise soft-max over the final feature map combined with the cross-entropy loss function[40]), and output pixel by pixel class annotations. The reason we choose U-Net is it's special engineered loss function are capable to offer consistent segmentation in discontinuous fashion. With the help of AirSim interfaces, we managed to access each 3D mesh object and map into the ground truth annotations, and by constantly collecting different time of the day combined with weather conditions combinations, we managed to collect more than 2TB of training data in different environments (check the experiment and result section for further details) and validated over random validation batches from normal weather and day-time combinations. The pixel to pixel segmentation with high precision is ideal for navigation because of it filtered out the noisy details yet still retains the geometry information and topological landmark information both of which are crucial for visual navigation. Upon this level of abstraction, learning agent can transfer to new environments easily with few learning epochs, even the new domain looks totally different.

Rather than directly feed the segmentation data into learning agent, we propose to continue the abstraction one more step: computing the attention map. We first extract the boundaries from segmentation map by finding the intensity gradient of the segmented annotations.

$$G_{xy} = \eta I_x \bigsqcup I_y, \forall x, y \in \phi$$

Here I is a mapping function:

$$I(x, x') = \begin{cases} 0, & \text{if } x == x' \\ 1, & \text{otherwise} \end{cases}$$

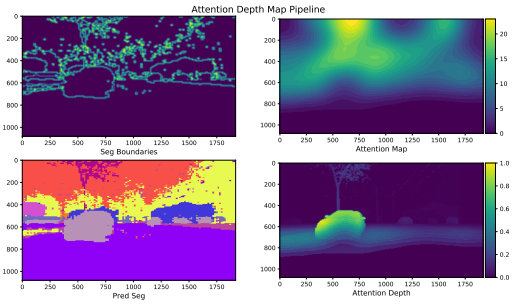

Figure 3: Attention Model pipeline.

Since we don't need to care about the edge directions, we can skip the direction computation part for efficiency. Equations above are equivalent to apply convolutions with two orthogonal $n \times n$ matrix Note that η is a normalization term proportional to class types in unit square (here we use 6 by 6 pixels). We then apply the diffusion function with ADI-method (alternating direction implicit[41]) along the boundaries. Define two direction of diffusion as:

$$\alpha = \frac{D\Delta t}{2\Delta x_i^2}, \ \beta = \frac{D\Delta t}{2\Delta y_j^2}$$

By fix y then fix x direction we get:

$$-\alpha u_{i+1,j}^{n+\frac{1}{2}} + (1+2\alpha)u^{n+\frac{1}{2}} - \alpha u_{i-1,j}^{n+\frac{1}{2}} = \beta u_{i,j+1}^{n} + (1-2\beta)u_{i,j}^{n} + \beta u_{i,j-1}^{n}$$

$$-\beta u_{i,j+1}^{n+1} + (1+2\beta)u_{i,j}^{n+1} - \beta u_{i,j-1}^{n+1} = \alpha u_{i+1,j}^{n+\frac{1}{2}} + (1-2\alpha)u_{i,j}^{n+\frac{1}{2}} + \alpha u_{i+1,j}^{n+\frac{1}{2}}$$

By sweeping of 50 iteration in both directions, we get the diffused boundary maps (As shown in Fig 2).

However, there are still too much heat after the diffusion, we then squeeze the attention map and depth map together using sigmoid regularization function:.

$$u_{x,y} = \frac{1}{1+e^{-\sqrt{u_{x,y}}}},\ d_{x,y} = \frac{1}{1+e^{-\sqrt{d_{x,y}}}}$$

$$d_{x,y} = 1 - \frac{d_{x,y} - \min d_{x,y}}{\max d_{x,y}},\ g = u * d$$

Here u is diffusion map, d is logarithmic inverse depth map, g is the final attention map by element-wise product from depth map d and diffusion map u. Notice that the attention map in Fig 3 filtered out both the road parts that close to the camera and the remote sky that offers little or no use to navigation. This well-defined weight mapping technique offers the agents better guidance on close by objects and can successfully catch their attentions, thus was named as attention map. The simple pipeline makes the abstraction inference fast enough (10fps on Nvidia Jetson TX2) to apply in real-robots. The only drawback now is that this system requires depth sensor and their careful alignments, future extension will expand the inference from depth sensor to deep networks or visual odometries with depth completion.

Another key component worth to mention is that we generate the pooling scale space (Fig 4), named as pooling pyramid to each weighted segmentation map, and we also employed three pretrained VGG16 network to predict along three different scales, yet keep the same dimension on the output end. By doing this, we force the LSTM cells to take into the consideration of global changes and still pay attention to the details. However, with the complexity of model structure, the learning cells requires a lot more data sample than regular architectures, thus slow down the learning from early stage, yet through the course of training, the advantage of memory for time-series prediction gradually started to take effect and contribute to final convergence.

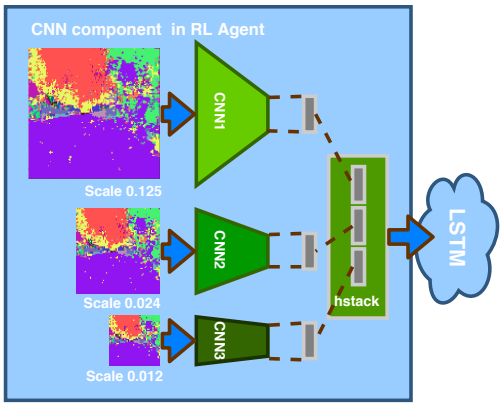

Figure 4: Different Scale of Image was filtered by CNN networks and stack their outputs as input for LSTM Cells.

3.2 Reinforcement Learning Model

The main reason this navigation system works with sparse reward is that by simplify state-space with abstracted representations, learning agent requires less memories to fully parameterise the training sample thus converge faster, just like the other experiments playing Atari games[42], on the contrary, adding recurrent memories made the system propagate sparse rewards backwards to the batched samples in the sliding window and further speed up the converge time.

Within DQN training sessions, we scale our model to the Markov Decision Process framework, which is a tuple of $(U, A, \theta_{sa}, \gamma, R)$ namely: Abstraction Map (U), action (A), transition probabilities (θ_{sa}), discount factor γ and reward(R). And followed by the accumulated rewards under policy π:

$$J(\pi) = \max_{\pi} E[R(s_0) + \gamma R(s_1) + ...|s\ \pi(s)] \tag{1}$$

$$\pi^*(s) = arg\max_{\pi} J(\pi) \tag{2}$$

$$R(s) = \lambda(R_{gdis}(s) + R_{dis}(s)) + (1-\lambda)R_{speed}(s) - \eta\delta_a,\ \lambda \in [0,1] \tag{3}$$

Here the reward function has three components: $R_{gdis}(s)$ is distance reward with respect to the traveling distance in each state starts from s_0, $R_{dis}(s)$ inversely proportional to the road center to reward the learning agent from keep inside road, $R_{speed}(s)$ encourage the agent with higher speed and the final term δ_a penalize the agent from frequently change actions. η is a small constants. Define u_t as the weighted segmentation annotation, and a_t as the action at time t. The goal of the model is to predict the Q-function $Q_\pi(s_t, a_t|w_q)$:

$$\zeta(w) = E[r + \gamma arg \max_{a_{t+1}} \nabla Q_\pi(s_{t+1}, s_t, a_{t+1}, a_t)]^2 \tag{4}$$

$$J(w) = \max_w \zeta(w_q) \tag{5}$$

And perform differential w.r.t w_q, the parameters in Q function network. and optimized by gradient. As for policy function $\pi(s|w_\pi)$ in DDPG experiments we do:

$$\frac{\partial J}{\partial w_\pi} = \frac{\partial Q}{\partial a}|_{a \propto \pi(s|w_\pi)} \frac{\partial \pi(s|w_\pi)}{\partial w_\pi} \tag{6}$$

to update the policy network weights w_π with the partial derivative from accumulated reward expectations.

The reason why DQN and it's extension achieved high scores in Atari games is partially because most of the game environments can be described on MDP, or at least are able to be scale into one. However, with the dynamic real-world, robots need to maintain their own model based on their observations. So, Partial Observable MDP extends MDP with observation o and the model θ based on observation. We found in the experiments that DQN and DDPG under MDP are able to easily learn to control speed and steer for one turn, yet tends to diverge when the way points changed through each episodic randomization process. They had a very hard time under environment changes even after applying abstraction augmentation. To deal with this situation, we apply the LSTM on Q functions[43] which estimate $Q(o_t, u_{t-1}, a_t)$ rather than directly from abstracted mapping state and actions, here u_t is the return from previous cell or step. Our experiments show that adding recurrency to the Q-network allows the Q function to better estimate the underlying geometry.

4 Experimental Results

We use AirSim as our test bed and mainly did the experiments in AirSim-Neighborhood Environment. and completed 7 training iterations. We dumped all the crashing locations in the map and averaged over 7 runs to see how far the robot can explore. In the experiments, we constrained the robot travel distance to prevent the accumulating of rewards (For better plotting results). Additionally, we test the experiments on a PC with GTX1080Ti graphic card, and the average abstraction map inference frame rate is 20^+_-3, yet when tested on Nvidia Jetson TX2, we only got 9^+_-4 fps, which is still low for real-time robots, future investigation on efficiency optimization is needed.

Quantitative Evaluation: From Fig5 we can see that our model converges faster and with higher bottom-bound. We compared it with four baselines, the two traditional pure RL methods with end to end fashion, and the RL backend with Abstraction. The scaled reward reported is the accumulated return function with respect to training episode. The baseline DQN does not converge since the high dimensional data representation are hard to capture, besides, high portion of repeated parts leads to ambiguity to the leaning network, thus partially degenerated the Q learning into random explorations. We see that DDPG with segmentation abstraction leads to very fast learning process in the beginning yet bears with very high variance than our model. Recall that gradients are update randomly from sampled trajectories $(\tau) : \nabla_w J(w) \simeq \frac{1}{N} \sum_{i=1}^{T} \nabla_w log \pi_w(\tau) r(\tau)$. Since each trajectory are composition of sample actions and their corresponding states from the policy, thus can be wildly different as shown above. On the contrary, recurrent network tends to have more stable predictions over the action references thus narrowed down the trajectory variance in our model.

Another observation noteworthy is that with the depth attention map, the learning agent always turn to choose minimum action until the collision object are fairly close, similar evidences can be found in human mind[44] that we do not care about the objective progress in precise details until it caught our attention, and then concentrate the computational resources to reach the upcoming attention-worthy objectives.

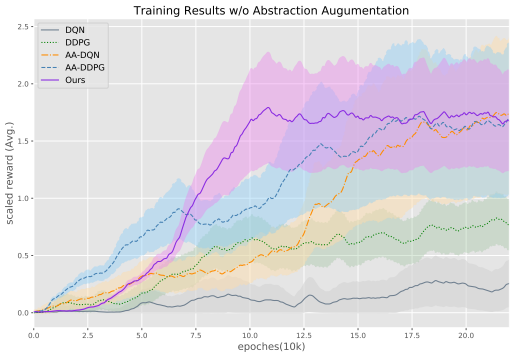

Figure 5: Learning efficiency (smooth scaled by 0.92) of our abstraction model against baselines. The plots are averaged over 7 runs with different random seeds. Hyperparameters are the same accross each runs. Here AA means Abstraction Augumentation.

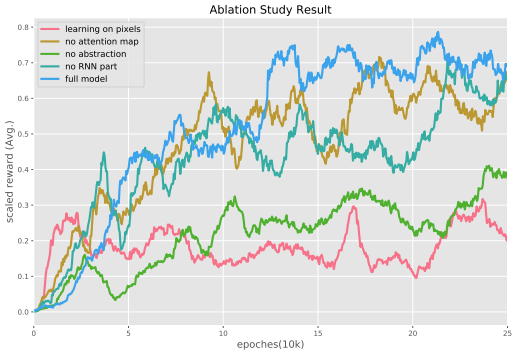

Figure 6: Ablation test by removing each individual component from the full model. Notice that by removing the abstraction component the learning method delegate to DQN. We can see that without the attention map, the agent oscillates drastically, partially because the inference by segmentation parts from sky or tree leaves.

Models	Config	Earliest Converge Time(10k)				
		Leaves	Fog	Rain	Dust	Snow
DQN	Abstraction	0.77	0.93	0.89	0.73	0.54
	E2E	0.41	0.08	0.61	0.38	0.24
DDPG	Abstraction	1.57	1.32	0.90	**1.02**	1.39
	E2E	0.65	0.22	1.33	0.89	0.56
Ours	AA+RNN	**1.59**	**1.79**	**1.52**	0.93	**1.46**

Table 1: Scaled total reward in different test environments. Note we average the rewards from 20 runs in different time-of-day with equal episode iteration times.

We further compared our model with four baselines in the transfer learning problems as shown in the table (note that all our end to end methods are trained under sunny summer afternoon environmental condition). From Table 4 we can see that our methods performs better in most environments. Since End to End will be mostly nullified in vision unfriendly days, depth information that encoded in the abstraction map then contributes in the learning phase and lead to a reasonable converge.

5 Conclusion

In this paper, we proved that abstraction layer can help reinforcement learning algorithms cut the correlation with pixels. Semantic abstraction represent images in a latent coding yet still retains rich geometrical representations. Additionally, our proposal of attention map geometrically kept the key information for agent to the objects that should be paid attention to, while filtered out those redundant and unrelated representations that further reduced the learning burden and ambiguity.

limitations: Our segmentation based pipeline still suffers from noise and uncertainty of deep structures. Deep learning methods' black-box nature makes debug even harder. thus, future extension should rely more on finding robust and efficient representational abstractions. Further extensions can be: 1. extend the segmentation methods from pixel to pixel into patch to patch, combined with depth map to scale into real-time robots with low-computation power. 2. learning methods can be upgraded with imitation learning or inverse reinforcement learning methods to expedite learning speed. 3. Depth estimation is essential to this system, we can employ the depth estimation architectures to realize camera only navigation.

Acknowledgments

References

[1] V. Navalpakkam and L. Itti. An integrated model of top-down and bottom-up attention for optimizing detection speed. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2049–2056. ieee, 2006.

[2] G. Konidaris. On the necessity of abstraction, Dec 2018. URL https://www.sciencedirect.com/science/article/pii/S2352154618302080.

[3] J. Goody and J. R. Goody. *The domestication of the savage mind*. Cambridge University Press, 1977.

[4] M. Müller, A. Dosovitskiy, B. Ghanem, and V. Koltun. Driving policy transfer via modularity and abstraction. *arXiv preprint arXiv:1804.09364*, 2018.

[5] O. Nachum, S. S. Gu, H. Lee, and S. Levine. Data-efficient hierarchical reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 3303–3313, 2018.

[6] J. Rasmussen. The role of hierarchical knowledge representation in decisionmaking and system management. *IEEE Transactions on systems, man, and cybernetics*, (2):234–243, 1985.

[7] C. B. Frankel and M. D. Bedworth. Control, estimation and abstraction in fusion architectures: Lessons from human information processing. In *Proceedings of the Third International Conference on Information Fusion*, volume 1, pages MOC5–3. IEEE, 2000.

[8] A. Mahendran and A. Vedaldi. Visualizing deep convolutional neural networks using natural pre-images. *International Journal of Computer Vision*, 120(3):233–255, 2016.

[9] M. Garnelo, K. Arulkumaran, and M. Shanahan. Towards deep symbolic reinforcement learning. *arXiv preprint arXiv:1609.05518*, 2016.

[10] M. Jaderberg, W. M. Czarnecki, I. Dunning, L. Marris, G. Lever, A. G. Castañeda, C. Beattie, N. C. Rabinowitz, A. S. Morcos, A. Ruderman, et al. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, 2019.

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

[12] P. Henderson, R. Islam, P. Bachman, J. 22 Pineau, D. Precup, and D. Meger. Deep reinforcement learning that matters. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[13] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.

[14] D. C. Burr, M. C. Morrone, and D. Spinelli. Evidence for edge and bar detectors in human vision. *Vision research*, 29(4):419–431, 1989.

[15] S. Shah, D. Dey, C. Lovett, and A. Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, 2017. URL https://arxiv.org/abs/1705.05065.

[16] S. Zeki, J. Watson, C. Lueck, K. J. Friston, C. Kennard, and R. Frackowiak. A direct demonstration of functional specialization in human visual cortex. *Journal of neuroscience*, 11(3): 641–649, 1991.

[17] E. Vul, G. Alvarez, J. B. Tenenbaum, and M. J. Black. Explaining human multiple object tracking as resource-constrained approximate inference in a dynamic probabilistic model. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1955–1963. Curran Associates, Inc., 2009.

[18] T. Glasmachers. Limits of end-to-end learning. *arXiv preprint arXiv:1704.08305*, 2017.

[19] S. Wang, R. Clark, H. Wen, and N. Trigoni. Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2043–2050. IEEE, 2017.

[20] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging*, 2017(19):70–76, 2017.

[21] M. Hausknecht and P. Stone. Deep recurrent q-learning for partially observable mdps. In *2015 AAAI Fall Symposium Series*, 2015.

[22] A. Valmari. The state explosion problem. In *Advanced Course on Petri Nets*, pages 429–528. Springer, 1996.

[23] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[24] M. Ishikawa. Structural learning with forgetting. *Neural networks*, 9(3):509–521, 1996.

[25] S. Thrun and L. Pratt. *Learning to learn*. Springer Science & Business Media, 2012.

[26] T. Yu, C. Finn, A. Xie, S. Dasari, T. Zhang, P. Abbeel, and S. Levine. One-shot imitation from observing humans via domain-adaptive meta-learning. *arXiv preprint arXiv:1802.01557*, 2018.

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

[28] F. Sadeghi and S. Levine. Cad2rl: Real single-image flight without a single real image. *arXiv preprint arXiv:1611.04201*, 2016.

[29] T. Manderson, R. Cheng, D. Meger, and G. Dudek. Navigation in the service of enhanced pose estimation. In *International Symposium on Experimental Robotics (ISER)*, 2018.

[30] R. Cheng, T. Manderson, D. Meger, and G. Dudek. Robust visual pose tracking on a vehicle for off-road driving. In *International Conference on Robotics and Automation (ICRA)*, 2019.

[31] Y. Zhang, P. David, and B. Gong. Curriculum domain adaptation for semantic segmentation of urban scenes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2020–2030, 2017.

[32] S. Shah, D. Dey, C. Lovett, and A. Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and service robotics*, pages 621–635. Springer, 2018.

[33] A. Kendall, J. Hawke, D. Janz, P. Mazur, D. Reda, J.-M. Allen, V.-D. Lam, A. Bewley, and A. Shah. Learning to drive in a day. *arXiv preprint arXiv:1807.00412*, 2018.

[34] Y. F. Chen, M. Everett, M. Liu, and J. P. How. Socially aware motion planning with deep reinforcement learning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1343–1350. IEEE, 2017.

[35] S. Wang, D. Jia, and X. Weng. Deep reinforcement learning for autonomous driving. *arXiv preprint arXiv:1811.11329*, 2018.

[36] S. Shalev-Shwartz, S. Shammah, and A. Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*, 2016.

[37] J. Yuan, H. Wang, C. Lin, D. Liu, and D. Yu. A novel gru-rnn network model for dynamic path planning of mobile robot. *IEEE Access*, 7:15140–15151, 2019.

[38] V. Talpaert, I. M. Sobh, B. R. Kiran, P. Mannion, S. Yogamani, A. E. Sallab, and P. Y. Pérez. Exploring applications of deep reinforcement learning for real-world autonomous driving systems. *ArXiv*, abs/1901.01536, 2019.

[39] A. Jansson. Autonomous driving in crossings using reinforcement learning. 2017.

[40] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[41] V. Simoncini. Computational methods for linear matrix equations. *SIAM Review*, 58(3):377–441, 2016.

[42] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[43] G. Lample and D. S. Chaplot. Playing fps games with deep reinforcement learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[44] H. Moravec. *Mind children: The future of robot and human intelligence*. Harvard University Press, 1988.