Learning Realistic and Safe Pedestrian Behavior by Imitation

Doctoral Consortium

José Aleixo Cruz
Faculty of Engineering of the University of Porto
Porto, Portugal
up201403526@fe.up.pt

ABSTRACT

Navigating amongst pedestrians is a very complex task for an autonomous agent. Not only must the agent understand traffic rules and navigate safely, but it must also act in a way that obeys social norms and does not interfere with other pedestrians. Here, we focus on obtaining a model that portrays pedestrian behavior from real-life demonstrations using imitation learning. We create a reinforcement learning environment that allows an agent to learn to navigate using the obtained behavior model. The work is still in progress, but we illustrate how we generate demonstrations of pedestrian behavior from video captured by smart glasses and we incorporate them into a reinforcement learning environment.

KEYWORDS

Imitation Learning; Reinforcement Learning; Pedestrian Behavior

ACM Reference Format:
José Aleixo Cruz. 2021. Learning Realistic and Safe Pedestrian Behavior by Imitation: Doctoral Consortium. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 3 pages.

1 INTRODUCTION

Deep reinforcement learning (DRL) [1] combines reinforcement learning and deep neural networks (DNNs), allowing computational agents to learn to execute highly complex tasks. Researchers have mainly explored DRL methods in virtual environments; however, DRL applications are emerging in real-life scenarios. Our study focuses on the particular scenario of pedestrian navigation, where an autonomous agent navigates as a pedestrian amongst other pedestrians and vehicles.

Pedestrian navigation is a straightforward task that many people execute in their daily routines. Still, it requires understanding traffic signs, such as pedestrian signals and crosswalks, social norms, such as the proper distancing between people, and traffic rules. Therefore, understanding pedestrian behavior is crucial for teaching an agent to navigate without impeding or interfering with other pedestrians' circulation and blending in with the crowd. Above all, the navigation should be performed in a safe manner, both to the agent executing the task and to others in the vicinity.

Imitation learning techniques [7] rely on experts' task demonstrations to allow a reinforcement learning agent to learn a policy similar to the experts'. We hypothesize that by using pedestrian navigation demonstrations and imitation learning, we can teach an agent to navigate as a pedestrian would.

The SIMUSAFE project [1] aims at collecting road users' behavioral data in naturalistic and realistic scenarios so as to produce more accurate decision-making models. Using video captured from a monocular camera worn by pedestrians, we employ machine learning and computer vision techniques to generate demonstrations that we can utilize in a custom reinforcement learning environment to teach an agent how to navigate in an urban scenario.

2 GENERATING DEMONSTRATIONS

We asked subject pedestrians to record their daily commutes using smart glasses. Collecting data in a natural setting has the advantage of describing behavior more accurately than data collected in a controlled environment. The smart glasses' camera captures images at 30 Hz with 1920×1080 pixel resolution. While the camera neither records sound nor captures the same field of view as the human eye, it portrays the surroundings that the pedestrian is paying more attention to.

We frame the problem of pedestrian navigation as a Markov decision process (MDP), where at a given time t the pedestrian agent observes a state of the environment $s_t \in S$. The pedestrian takes action $a_t inA$ and receives a reward r_{t+1} based on the new state s_{t+1}. We consider the trajectory τ of an episode with T time steps the sequence of states visited, actions taken, and rewards received during the episode: $\tau = \{s_0, a_0, r_1, s_1, a_1, r_2, \ldots, s_{T-1}, a_{T-1}, r_T, s_T\}$. We consider a demonstration as a set of trajectories of an expert agent, i.e., an agent that performs the optimal policy for the MDP. We consider an optimal policy to be a policy that portrays human behavior.

Because it is not trivial to write a reward function for human behavior, we cannot train an agent to perform similar actions to actual pedestrians using only an objective function. Instead, we assume that pedestrians are experts in pedestrian navigation. Thus, we utilize information collected through behavior elicitation to construct demonstrations that enable agents to learn a policy similar to real pedestrian behavior. More specifically, we attempt to extract trajectories for the pedestrian navigation task from the videos collected by the smart glasses using machine learning and computer vision techniques.

2.1 Pedestrian Observation

For each frame of a video recorded by the smart glasses, we detect objects in the frame using a neural network trained with YOLO [9] on the KITTI datase [2]. The neural network can identify three

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.

[1]http://simusafe.eu/

classes of entities: cars, people, and traffic signs. For each frame, it outputs the type and the bounding box of the identified entities.

Because the videos are from a single monocular camera, we cannot determine the depth of pixels in video frames using techniques for stereo settings. Knowing each pixel's depth lets us map the pixel's two-dimensional position in the image plane to a three-dimensional position in the camera reference frame. The camera reference frame, in this case, is also the pedestrian's reference frame, so it is advantageous to interpret surroundings from this point of view. To estimate pixel depth, we trained a neural network using the MonoDepth [5] algorithm on the Karlsruhe dataset [3, 4]. The network estimates the disparity map of an input frame, which we use to calculate the respective pixel depth map. We calculate every pixel's 3D position in an image from the depth map and their 2D position on the image plane, which lets us place detected objects relative to the pedestrian.

We further use the depth map to estimate the position of any obstacles that have not been identified by the detection network or that do not belong to any of the identifiable classes. We perform ground estimation using the technique described by Kircali and Tek [8]. We apply the method assuming fixed camera pitch and roll. Video captured by the smart glasses is not stable, therefore to reduce the error of the ground estimation method, we stabilize the video's roll angle using pointer feature matching. We denote the estimate of the ground plane z_g, and assume that pixels with depth $z < z_g + \theta$ are above the ground and therefore should be considered obstacles. θ is an arbitrary threshold we use to make sure we filter out any noise in the estimation to be an obstacle.

To generate an observation from the obtained data, we describe each pixel (x, y) of the frame with a tuple $p_{x,y} = \langle c, z \rangle$, where c is the cost of the pixel and z is the depth of the pixel. The cost c of the pixel is equal to 1 if it portrays an object detected by the neural network or an obstacle detected using ground estimation, otherwise $c = 0$. We consider an observation to be the two-dimensional matrix where the element (i, j) is the tuple $p_{i,j}$. One possible improvement to this observation structure is reducing the matrix size. As we are dealing frames with full HD resolution frames, the matrix dimension can be too large for training. To reduce the size, we may average the values for c and z in blocks of four elements. The value of c is then rounded to the nearest integer, 0 or 1. If there is a need to reduce the dimension further, we simply apply the same method to the resulting matrix.

2.2 Pedestrian Action

We consider a pedestrian's action to be its movement between two consecutive frames. More specifically, we consider an action at time step t to be the transformation of the pedestrian's pose between time step t and $t + 1$. For these poses, we consider the frame of reference the agent's initial pose, that is, the pose at time step $t = 0$. To calculate the action, we must first estimate the pedestrian's pose for each time step. We achieve this by employing simultaneous localization and mapping (SLAM) techniques. More specifically, we employ visual RGBD SLAM using the OpenVSLAM framework [11]. The SLAM algorithm outputs the pose of the pedestrian for every video frame.

3 REINFORCEMENT LEARNING ENVIRONMENT

To train and test our imitation learning agents, we have built a 3D environment using the Unity game engine.

Using mechanics provided by Unity, the agent can detect other pedestrians, cars, and traffic signs. The agent then calculates the bounding box of the detected objects from the current perspective. In Figure 1, we show how the agent perceives the bounding boxes from its current position. We do not need ground plane estimation nor depth estimation, as we can get these values directly from the environment.

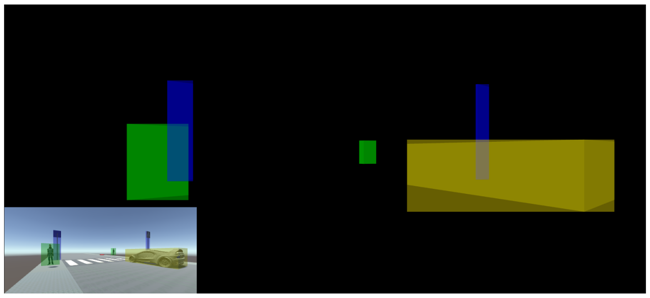

Figure 1: An illustration of the agent observing the bounding boxes of detected objects in the RL environment.

4 FUTURE WORK

We are currently in the phase of applying different techniques to teach an agent to navigate. Our first approach uses the Maximum Entropy Deep Inverse Reinforcement Learning (MaxEnt Deep IRL) [12] algorithm with the demonstrations we generated to get the objective reward function. We then use this reward function in our environment and let the agent learn using a reinforcement learning algorithm such as Proximal Policy Optimization (PPO) [10]. Our second approach uses Generative Adversarial Imitation Learning (GAIL) [6], which intrinsically computes a policy from the demonstrations.

After having a trained an agent, we test and validate its policy by comparing executing it in our Unity environment against scenarios that have been recorded but have not been used to train the agent. We hope to achieve behavior that is as natural as the real pedestrian's behavior.

A significant component for future work also involves making sure that the policy learned by the agent is safe and robust. We wish the agent to act naturally, but above all, it should not execute potentially dangerous maneuvers, such as bumping other pedestrians or crossing a road unsafely. Therefore, we will explore safe reinforcement learning techniques that ensure the agent obeys all safety constraints while demonstrating realistic behavior.

ACKNOWLEDGMENTS

This work is supported by the SIMUSAFE Project, funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 723386. This research is also supported by LIACC (FCT/UID/CEC/0027/2020).

REFERENCES

[1] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. 2017. A Brief Survey of Deep Reinforcement Learning. *CoRR* abs/1708.05866 (2017). arXiv:1708.05866 http://arxiv.org/abs/1708.05866

[2] Andreas Geiger, Philip Lenz, and Raquel Urtasun. 2012. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[3] Andreas Geiger, Martin Roser, and Raquel Urtasun. 2010. Efficient Large-Scale Stereo Matching. In *Asian Conference on Computer Vision (ACCV)*.

[4] Andreas Geiger, Julius Ziegler, and Christoph Stiller. 2011. StereoScan: Dense 3D Reconstruction in Real-time. In *Intelligent Vehicles Symposium (IV)*.

[5] Clement Godard, Oisin Mac Aodha, and Gabriel J. Brostow. 2017. Unsupervised Monocular Depth Estimation With Left-Right Consistency. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[6] Jonathan Ho and Stefano Ermon. 2016. Generative Adversarial Imitation Learning. arXiv:1606.03476 [cs.LG]

[7] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. 2017. Imitation Learning: A Survey of Learning Methods. *ACM Comput. Surv.* 50, 2, Article 21 (April 2017), 35 pages. https://doi.org/10.1145/3054912

[8] Doğan Kircalı and F. Boray Tek. 2014. Ground Plane Detection Using an RGB-D Sensor. In *Information Sciences and Systems 2014*. Springer International Publishing, 69–77. https://doi.org/10.1007/978-3-319-09465-6_8

[9] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 2016-Decem. IEEE, 779–788. https://doi.org/10.1109/CVPR.2016.91 arXiv:1506.02640

[10] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347 [cs.LG]

[11] Shinya Sumikura, Mikiya Shibuya, and Ken Sakurada. 2019. OpenVSLAM: A Versatile Visual SLAM Framework. In *Proceedings of the 27th ACM International Conference on Multimedia* (Nice, France) *(MM '19)*. ACM, New York, NY, USA, 2292–2295. https://doi.org/10.1145/3343031.3350539

[12] Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. 2016. Maximum Entropy Deep Inverse Reinforcement Learning. arXiv:1507.04888 [cs.LG]