Learning to Navigate Like Humans

Mahmoud Hamandi, Mike D'Arcy, and Pooyan Fazli

***Abstract*—We present a novel human-aware navigation approach, where the robot learns to mimic humans to navigate safely in crowds. The presented model referred to as DeepMoTIon, is trained with pedestrian surveillance data to predict human velocity. The robot processes LiDAR scans via the trained network to navigate to the target location. Our experiments show that DeepMoTIon outperforms state-of-the art in terms of human imitation and reaches the target on 100% of the test cases without breaching humans' safe distance.**

I. INTRODUCTION

Robots are gradually moving from factories and labs to streets, homes, offices, and healthcare facilities. These robots are currently assigned tasks that require interaction with humans, such as guiding passengers through busy airport terminals [1] or roaming around university buildings and interacting with nearby humans [2, 3].

As robots are increasingly becoming part of our everyday lives, it is essential for them to be aware of the surrounding humans while performing their tasks. Navigation is a basic skill for autonomous robots. We define human-aware navigation as the ability of the robot to navigate while complying with social norms and ensuring human safety.

While many existing systems allow robots to navigate safely within crowds [4, 5], they still rely on manually-crafted models of human motion. Such models may capture the aspects of human motion as understood by their designers, while they may likely miss subtle trends that characterize their human aspect. In addition, manually-crafted models do not have a way to automatically adapt to different cultures, so it may require significant manual effort to be used in a different environment.

In this context we present DeepMoTIon (Deep Model for Target-driven Imitation), a deep imitation learning algorithm that eliminates the need for an explicit model of human motion and instead learns the human navigation patterns directly by observing pedestrians.

With the absence of a true human simulator, learning the reward governing human motion is expensive with current Inverse Reinforcement Learning algorithms such as the one presented in [6]. As such, we tackle the imitation problem as a classification one, where the network learns a specific command for each observation without simulating the learned policy. This approach reduces the amount of time required for each architecture test and allows us to explore multiple network configurations.

II. PROBLEM DEFINITION

Figure 1 shows the different parts of our human imitation method. Our model is trained with the ETH pedestrian dataset [7] presenting videos of humans navigating in a real-world environment. The dataset contains environment maps and a set χ of humans, and for each human h the trajectory ζ_h that they took through the environment. Each ζ_h is a sequence of locations $l_{h,t}$, representing the position of human h at time t. We use a simulator to estimate LiDAR scans $\mathbf{z}_{h,t}$, velocities $\mathbf{v}_{h,t}$, and target locations $\tau_{h,t}$ for each human, which we then use to train the network to imitate the human trajectories. After training, the robot processes its target location and LiDAR scans via DeepMoTIon to calculate navigational commands that allow it to reach the target safely while moving similarly to the humans in the dataset.

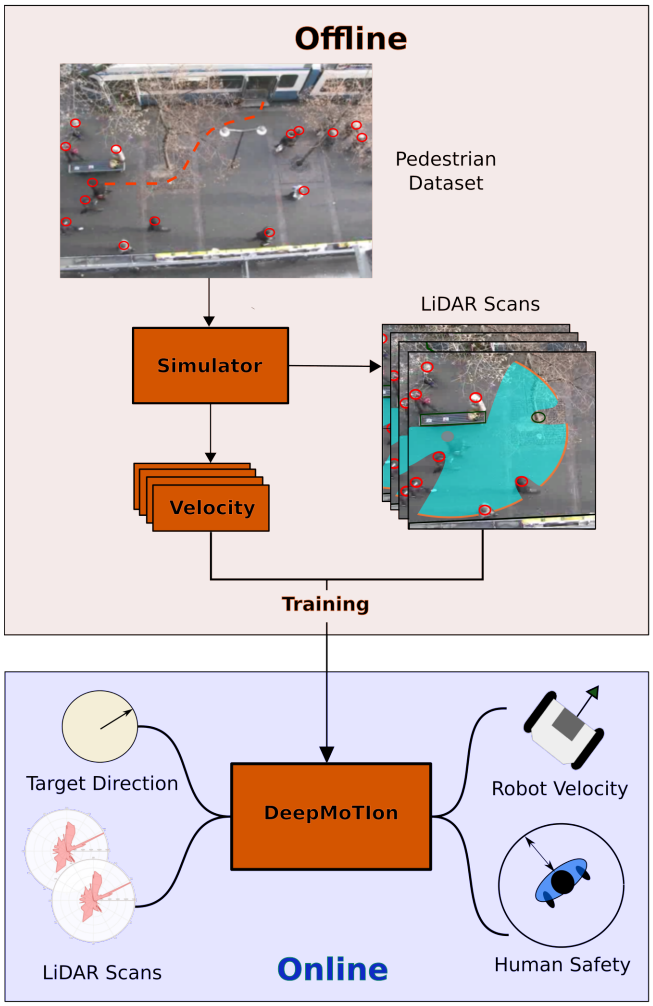

Fig. 1: Algorithm overview: human imitation learning with DeepMoTIon.

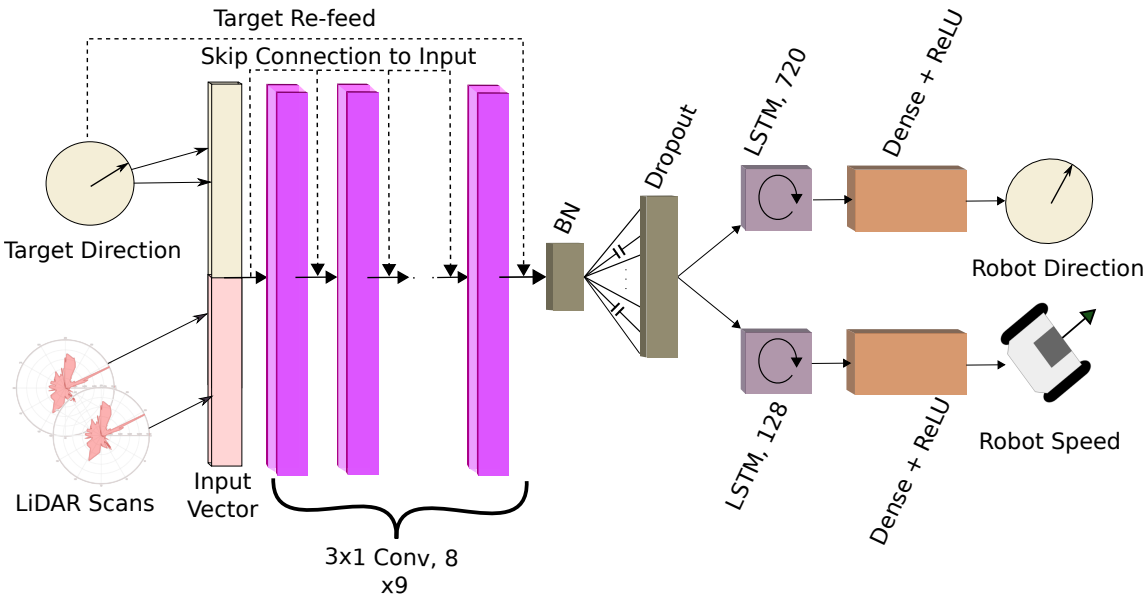

Fig. 2: DeepMoTIon network architecture.

III. DeepMoTIon

DeepMoTIon is a deep neural network $f(s_{h,t})$ defined as:

$$\{d_{h,t}, v_{h,t}\} = f(s_{h,t}),$$
$$s_{h,t} = [\mathbf{z}_{h,t-1}, \mathbf{z}_{h,t}, \tau_{h,t}]^T, \quad (1)$$

where $s_{h,t}$ is the input state vector, and $\{d_{h,t}, v_{h,t}\}$ is the output action set describing the direction and magnitude of the velocity $\mathbf{v}_{h,t}$. As shown in Equation (1), our network is trained to predict velocity commands to reach a target represented by its direction $\tau_{h,t}$, while processing the current and last raw LiDAR scans $\mathbf{z}_{h,(t-1,t)}$. The concatenation of the two LiDAR scans was necessary for the network to deduce static from dynamic obstacles and learn the motion pattern of the latter.

Our DNN architecture is shown in Figure 2. In this architecture the skip connections from the input were inspired by classical planning algorithms [8] such as value iteration and greedy search. However, after the convolutional layers we re-feed only the raw target direction to the network due to its direct correlation to the velocity direction, while the LiDAR scans add minimal value in their raw state. We found through experimentation that only shared convolutional layers were required for the network to correctly deduce the direction and speed from the input state while adding specialized convolutional layers for each of the two outputs reduced its performance.

In addition, for a planning algorithm each state and the corresponding action are tightly related to the previous observations. The LSTM layer was added to the network to allow it to remember the past state of the environment, as these layers have been shown to improve the prediction of future states based on their memory of the past [9]. We later provide an experimental comparison to show the LSTM's necessity. Batch normalization was necessary to assure the boundedness of the input to the LSTM layers.

A. Loss Function

Our loss function is designed to train the network to output the direction and speed as seen in the human dataset by minimizing the squared error of the speed and the cross entropy error of the output direction.

However, human imitation presents a challenge due to their stochasticity. In fact, two humans might behave differently even with the same observations depending on their personality and other hidden factors. This suggests that the correct direction might be one of many directions in a range about the ground-truth. As such it is desirable to penalize the network less for cases where it is close to the ground truth than cases where it is completely wrong. To this end, we model the output direction as a Gaussian distribution about the human-chosen direction with a standard deviation σ.

We also observe that humans closely approach obstacles and other humans. While this is acceptable between humans, a robot is typically less agile, so it must keep a larger distance to all obstacles to ensure the safety of both itself and humans. This motivates us to add a safety factor to the loss function that encourages a larger distance between the robot and the closest object around it.

As such, our complete loss function for a batch of N training examples can be expressed as follows:

$$\overbrace{\frac{1}{N}\sum_{i=1}^{N}(v_i - \hat{v}_i)^2}^{\text{Speed Loss}} + \overbrace{\frac{1}{N}\sum_{i=1}^{N} H(d_i, \hat{d}_i(\sigma))}^{\text{Direction Loss}}$$
$$+ \underbrace{\frac{1}{N}\sum_{i=1}^{N} max(0, 1 - log(min(z_i) + 1 - safeDistance))}_{\text{Safety Loss}}, \quad (1)$$

where $H(p, q)$ is the cross entropy loss function, d_i is the predicted direction distribution, and $\hat{d}_i(\sigma)$ is the Gaussian distribution about the human-chosen direction with standard deviation σ.

IV. Experiments

To evaluate the performance of DeepMoTIon when imitating humans, we conducted experiments on the ETH BIWI walking pedestrians dataset [7]. The dataset provides annotated trajectories of 650 humans recorded over 25 minutes of time on two maps. To avoid overfitting and allow the network to generalize to unseen maps, the training data was augmented by replicating each path while rotating the map at random angles. The dense crowds and sudden changes of pedestrian direction in this dataset make it sufficiently challenging for our experiments.

We compared our results with the Generalized Reactive Planner (GRP) algorithm presented by Groshev *et al.* in [8], the End-to-End Motion Planning (EMP) algorithm presented by Pfeiffer *et al.* in [10] and the Social Force Model (SFM) based robot control scheme presented by Ferrer *et al.* in [4]. In addition, to test the necessity of the LSTM layer we test two variants of the network, where we refer to DeepMoTIon$_{LSTM}$ as the DNN with the architecture as explained before and DeepMoTIon$_{conv}$ as the DNN without any LSTM layers. More convolutional layers were added to DeepMoTIon$_{conv}$ to accommodate for the depth difference.

We assess the trajectories generated by each algorithm with the following metrics:

1) **Squared Path Difference (SPD)**: SPD is the sum of squared error between the correct human trajectory and the one taken by the robot.
2) **Proximity:** Proximity is the closest distance the robot comes to a human on its path.
3) **Number of Collisions:** The number of times the robot collides with a human while navigating.
4) **Target:** The percentage of trials where the robot reached the goal within the 400-step threshold.

A. Comparison with Benchmarks

Table I compares our network against the benchmark algorithms and assess each following the metrics presented earlier.

TABLE I: Performance Metrics Comparison

	SPD	*Proximity*	*Collisions*	*Target*
DeepMoTIon$_{LSTM}$	151	0.31	0.67	100%
DeepMoTIon$_{conv}$	732	0.25	0.89	69%
SFM	3817	0.29	0.26	100%
EMP	15437	0.001	7.69	32%
GRP	334	0.18	0.78	84%

Table I illustrates the ability of our network to imitate humans better than the other benchmarks for all the metrics, with the exception of SFM showing a lower rate of collision. SPD shows that our network has the lowest path difference for all the tested algorithms, with the next best algorithm (GRP) showing more than double the path difference.

Table I also shows that DeepMoTIon$_{LSTM}$ reaches the target on 100% of the trials, while the other networks fail many with EMP reaching the target only 32% and GRP reaching the target 84% of the trials. The proximity parameter shows that our network keeps an average proximity of $0.31m$ to any human, while SFM keeps a 0.29m despite explicitly weighting its repulsive force to humans the most out of the other social forces. *Collisions* shows that SFM has the lowest number of collisions among all algorithms. This can be explained by the ability of the algorithm to stop or reverse its direction in the case of dense crowds, while all the other networks were not trained on any human demonstration that exerted that type of behavior. We expect DeepMoTIon to learn to stop and avoid collisions better when trained on more pedestrian data showing a wider set of possible navigation scenarios.

In addition, the LSTM variant of DeepMoTIon imitates humans much better than DeepMoTIon$_{conv}$, as shown in Table I, where DeepMoTIon$_{LSTM}$ exhibits a better performance for all metrics.

Finally, we note that our network was able to navigate even with a LiDAR range other than the one it was trained on. All the algorithms above were trained and tested on a LiDAR with a $30m$ range. To show the ability of our network to generalize to different ranges, we tested its performance with a LiDAR scanner with a $6m$ range without any retraining. The network was still able to reach the target on 97% of the trials, with a slight increase in path difference, and a decrease in collisions to 0.51. The decrease in collisions was expected, as the network is observing obstacles in locations that were supposed to be free, and thus the robot becomes more careful.

V. Conclusion and Future Work

We introduced a novel deep imitation learning framework and studied its performance when learning to navigate from human traces. We trained the deep network to predict robot command velocities from raw LiDAR scans without the requirement of any preprocessing or classification of the surrounding objects. Our experiments showed DeepMoTIon's ability to generate navigational commands similar to humans, and plan a path to the target on all test sets, outperforming all of the benchmarks on path difference metrics, and all except SFM on other metrics like collisions. In addition, we presented a comparative assessment that showed the necessity of an LSTM layer for a planning algorithm via a DNN, where the robot navigating without the LSTM was led astray on many test cases. Finally, we presented a novel loss function to train the network. The loss function allowed us to accommodate for human motion stochasticity while at the same time forcing the robot to navigate safely.

In the future, we plan to train the network to navigate using raw images instead of LiDAR scans, where we believe the larger bandwidth of data can help the network understand human motion from their point of view.

References

[1] R. Triebel, K. Arras, R. Alami, L. Beyer, S. Breuers, R. Chatila, M. Chetouani, D. Cremers, V. Evers, M. Fiore *et al.*, "Spencer: A socially aware service robot for passenger guidance and help in busy airports," in *Field and Service Robotics*. Springer, 2016, pp. 607–622.

[2] M. Veloso, J. Biswas, B. Coltin, S. Rosenthal, T. Kollar, C. Mericli, M. Samadi, S. Brandao, and R. Ventura, "Cobots: Collaborative robots servicing multi-floor buildings," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, 2012, pp. 5446–5447.

[3] U. Patel, E. Hatay, M. D'Arcy, G. Zand, and P. Fazli, "Beam: A collaborative autonomous mobile service robot," in *Proceedings of the AAAI Fall Symposium on Artificial Intelligence for Human-Robot Interaction, AI-HRI 2017*, 2017.

[4] G. Ferrer, A. G. Zulueta, F. H. Cotarelo, and A. Sanfeliu, "Robot social-aware navigation framework to accompany people walking side-by-side," *Autonomous robots*, vol. 41, no. 4, pp. 775–793, 2017.

[5] E. A. Sisbot, L. F. Marin-Urias, R. Alami, and T. Simeon, "A human aware mobile robot motion planner," *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 874–883, 2007.

[6] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning." in *Proceedings of the AAAI Conference on Artificial Intelligence, AAAI*, 2008, pp. 1433–1438.

[7] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *Proceedings of the 12th IEEE International Conference on Computer Vision, ICCV*, 2009, pp. 261–268.

[8] E. Groshev, A. Tamar, S. Srivastava, and P. Abbeel, "Learning generalized reactive policies using deep neural networks," *arXiv preprint arXiv:1708.07280*, 2017.

[9] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2017.

[10] M. Pfeiffer, M. Schaeuble, J. Nieto, R. Siegwart, and C. Cadena, "From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots," in *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA*, 2017, pp. 1527–1533.