Imitating human joystick control ability using style and content disentanglement

Mingyuan Lei
RRIS, NTU, Singapore
mingyuan.lei@ntu.edu.sg

Neha Priyadarshini Garg
RRIS, NTU, Singapore
np.garg@ntu.edu.sg

Meenakshi Gupta
RRIS, NTU, Singapore
meenakshi.gupta@ntu.edu.sg

Tat-Jen Cham
SCSE, NTU, Singapore
astjcham@ntu.edu.sg

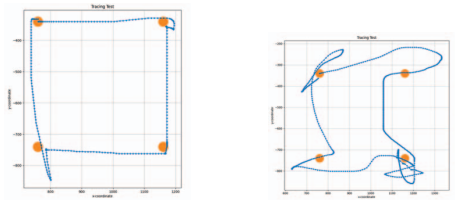
***Abstract*—Ability to extract user's joystick input style from a few demonstrations and applying it to various navigation tasks can be useful for automated testing of shared control algorithms in simulation. This can reduce the need for user studies, leading to time and cost benefits, especially when human subjects suffer from disabilities. State-of-the-art imitation learning methods require demonstration for every task making them unsuitable for this use case. Methods like adversarial motion prior (AMP) and learning from play (Play-LMP) provide an alternative. However, data used to train AMP's discriminator is generated by task-specific policies which can limit its ability to apply a style to other tasks. Also, it tries to optimize both style and task reward simultaneously which can lead to style not being imitated in favor of task completion. Instead of learning style through a discriminator, Play-LMP tries to extract style by learning latent representations. However, this is done in an unsupervised manner making it hard to enforce the correlation between the learned latent representation and the user's input style. In this work, we investigate how the supervised latent optimization based disentanglement approach (used to disentangle images to its style and content latent), can be used to extract style from a few human demonstrations and applied to different tasks with simulator in the loop. Our initial results on synthetic data show that this can be a promising approach.**

***Index Terms*—style transfer, learning from demonstration, joystick digital twin**

I. Introduction

Shared control [1], where the final control command is computed based on the user's input and the robot's perception of the environment, can help people with a lack of fine motor control, drive a robotic wheelchair independently using a simple joystick [2]. For testing shared control methods, human joystick input is needed at every time step. Thus testing is often done through user studies which are time-consuming, costly, and hard to do on a large scale, especially for human subjects with disability. The need for such user studies can be minimized by creating a model of the human joystick input policy from a few human demonstrations, which can then be used to generate the human joystick input for testing in different scenarios. Since the human joystick input will vary depending on a user's ability (See Fig. 1), the model should be able to generate joystick inputs conditioned on the ability of the person.

One way to create such a model is to use imitation learning approaches [3] that aim to learn a policy for performing a given task from human demonstrations. However, these approaches require human demonstrations for every new task, which is not feasible for our use case. We aim to extract the joystick input style of the user from a few demonstrations of some tasks and then use the extracted style to generate the joystick inputs for different tasks.



(a) (b)

Fig. 1. Human subjects were asked to trace a square between orange dots on the screen using a joystick. Trace shown as a blue dotted line of (a) a normal person, (b) a Cerebral Palsy (CP) subject. CP subjects are not able to trace a square properly due to a lack of fine motor control.

Peng et al. [4] proposed a method to extract the style from a few demonstrations and apply it to different tasks. They use a framework similar to generative adversarial imitation learning (GAIL [5]), where a generator and a discriminator are trained together, such that the generator learns to generate a policy that cannot be distinguished from demonstrated trajectories by the discriminator. To capture the style in demonstration and utilize it for different tasks, Peng et al. [4] modify the discriminator to learn whether the two consecutive states come from the demonstration, while the generator learns a policy that maximizes a task reward along with minimizing the discriminator's reward. This approach has been demonstrated to work on different skeletons to capture different gaits. While the discriminator does not use task information, the data used to train it is generated by task-specific policies. This may introduce some task dependencies into it and hinder its ability to apply style to other tasks. Also, having a separate task objective can lead to a policy that puts more emphasis on completing the task successfully instead of imitating the style. In general, approaches based on Generative Adversarial Network (GAN) are known to be difficult to train and require careful tuning of network parameters.

Instead of enforcing style through a discriminator, Lynch et al. [6] learn a latent representation for doing a task in different styles in an unsupervised manner with conditional variational autoencoders (CVAEs), and use this latent representation for performing different goal-conditioned tasks. Learning latent

979-8-3503-5409-6/24/$31.00 ©2024 IEEE
DOI 10.1109/CAI59869.2024.00135

representations in an unsupervised manner is not suited for our case as we need to extract the style of a person and condition the task on it. With unsupervised learning, it is hard to make sure that the learned latent space correlates with the style of different humans.

Thus there is still a need for an approach that can extract human input style from a few human demonstrations and apply it to generate valid trajectories of completing different wheelchair operation tasks. Many works in image processing [7]–[9] show how latent embeddings, which disentangle style and content, can be extracted in a supervised manner from images. We are in particular interested in the latent optimization method [7] as it can learn the embeddings in a supervised manner without the need to train a generative adversarial network.

In this work, we demonstrate how disentanglement using latent optimization can be applied for learning embeddings for user's style from user demonstrations which can then be used to generate the control inputs conditioned on different user styles with a simulator in the loop.

Through our preliminary experiments on synthetic data for robotic wheelchair simulated in Gazebo and controlled by Robot Operating System (ROS), we show that our approach can successfully disentangle the user input style from a few demonstrations and use it to generate trajectories for different tasks.

II. Our Approach

We first define our problem in the context of a robotic wheelchair that can be operated via a joystick.

A. Problem Statement

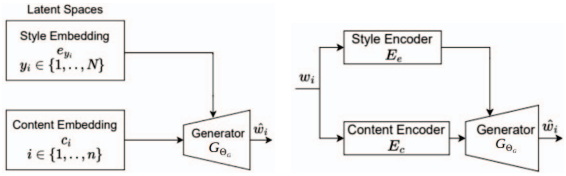
Given a set of n demonstrations, consisting of wheelchair trajectories $\xi_{1..n}$ in a few simulation scenes, generated by N people having different fine motor control abilities using a joystick for wheelchair control, we aim to extract the style of joystick input and apply it to different tasks i.e. traversing a path in a different scene with simulator in the loop, such that the generated trajectory is similar to the trajectory that would have been generated by the person with that style. Each trajectory is labeled with the person who created it. We assume that the variance in the demonstration for a person with a given style is far less than the difference in the trajectories of persons with different styles.

B. LORD

We aim to solve this problem by disentangling the style and task from the few demonstrations given by the person using the LORD (latent optimization representation disentanglement) approach [7] that is used for disentangling style and content of images. We give a brief overview of the LORD next.

Given a set of n images, with each image labeled with a style that can have upto N values, LORD learns to disentangle this set into N style embeddings and n task embeddings using a two-stage process.

In Stage 1 (see Figure 2(a)), each image w_i such that $i \in 1, .., n$ is assigned a style embedding e_{y_i}. This embedding



(a) (b)

Fig. 2. (a) Stage 1: Latent optimization. Parameters of the generator and all the style and content embeddings are optimized using SGD optimizer. All inputs of the same style share a single style embedding. The network is trained using Equation 1. Once the network is trained, the latent spaces of the training set are disentangled. (b) Stage 2: Both the encoders are trained to generate the optimized embeddings of stage 1 for a given input. The model is trained using Equation 3.

is selected from a pool of N randomly initialized embeddings based on the style label y_i , which is recorded in the dataset and ranges from 1 to N. Additionally, each image receives a distinct randomly initialized content embedding. A generator G, with parameters θ_G, transforms these embeddings into the output image $\hat{w}_i$. The latent embedddings $e = (e_1, \ldots, e_N)$ and $c_i = (c_1, \ldots, c_n)$ and generator parameters θ_G are then learned in an end-to-end manner using stochastic gradient descent such that the images' reconstruction loss $\mathcal{L}_1$ is minimized i.e.

$$\mathbf{e}^*, \mathbf{c}^*, \theta_G^* = \arg\min_{\mathbf{e},\mathbf{c},\theta_G} \mathcal{L}_1$$

$$\mathcal{L}_1 = \sum_{i=1}^{n} \|G_{\theta_G}(e_{y_i}, c_i) - w_i\|_2 \tag{1}$$

Since images with the same style are assigned the same style embedding, style embedding captures all the common style features resulting in style and content disentanglement.

In Stage 2 (see Figure 2(b)) style and content encoders, E_e and E_c are trained to estimate the optimized embeddings obtained from Stage 1 for new input images. The loss function aims to minimize the error between the embeddings estimated by the encoders and the original embeddings learned in Stage 1. A reconstruction loss term is added to the loss function to ensure that the learned embeddings can reconstruct the original input sequence. Thus, the overall loss function for Stage 2 becomes:

$$\begin{aligned}\mathcal{L}_2 = \sum_{i=1}^{n} &\|G_{\theta_G}(E_e(w_i), E_c(w_i) - w_i\|_2 \\ &+ \alpha_1.\|(E_e(w_i) - e_{y_i})\|_2 \\ &+ \alpha_2.\|(E_c(w_i) - c_i)\|_2 \end{aligned} \tag{2}$$

The parameters of both the encoders θ_e, and θ_c, and generator network θ_G (initialized with weights from Stage 1),are learned end-to-end using stochastic gradient descent:

$$\theta_e^*, \theta_c^*, \theta_G^* = \arg\min_{\theta_e,\theta_c,\theta_G} \mathcal{L}_2 \tag{3}$$

At the inference time, the style embedding of the input image can be computed using the style encoder, combined with the content embedding computed from some other image, and then used to generate an image with a given style but different content.

C. LORD for human demonstrations

In this section, we will describe how we can use the LORD framework to capture style from human demonstrations. First, we describe how we represent the set of n demonstrations.

1) Input representation: A demonstration takes the form of the wheelchair trajectory ξ_i when controlled by a human. It is represented as a sequence of tuples $(x_i^t, y_i^t, \theta_i^t)$, where x_i^t, y_i^t is the 2D position and θ_i^t is the rotation around z-axis of the wheelchair at timestep t for trajectory ξ_i.

We assume that when given the same task, the difference in style comes from the different physical abilities of the people. Therefore we capture the style relative to a baseline policy for doing the task. This baseline policy can be represented by the global path plan which also implicitly defines the task. This baseline policy is dependent on the current position of the wheelchair. Thus, we record the global path plan generated by A^* planner at each time step for all the trajectories. In a Robot Operating System (ROS), global planner output is recorded as a sequence of waypoints spaced 2cm apart from each other, which is different from the frame or timestep used in the rest of the data. As a result, to provide valid information for the model, we represent global plan P_i corresponding to ξ_i as a sequence of 10 waypoints $(p_{i,1}^t, ..., p_{i,10}^t)$ where $p_{i,k}^t$ represents k^{th} waypoint of the global plan P_i at time step t. Each waypoint is a 2D position. In our experiments, 10 waypoints (20cm) are sufficient to cover the distance of at least 1 timestep, even under the maximum speed.

2) LORD: Each of our recorded data r_i is a sequence of $((x_i^t, y_i^t, \theta_i^t), (p_{i,1}^t, p_{i,2}^t ... p_{i,10}^t))$ i.e. wheelchair pose and 10 way points at every time step t. The length of each sequence varies depending on the time taken to reach the goal which determines the number of time steps in the sequence. Each sequence is labeled with the human/style who gave the demonstration that can have up to N values.

One naive way of using LORD is to use each recorded data r_i as equivalent to an image input and try to reconstruct it in Stages 1 and 2. However, by doing this, we will only be able to reconstruct the whole trajectory at the inference time. This trajectory will not have a simulator in the loop. For our use case, we need to compute the position of the wheelchair $((x_i^t, y_i^t, \theta_i^t)$ at next timestep t, given the past trajectory till time $t-1$ and global plan at time t. This computed position is then sent to the simulator which will provide the actual position and the new global plan from that position. Then we need to take this output from the simulator to again compute the wheelchair position at the next timestep. See Figure 3.

In order to do this, we define our LORD input w_i using a sliding window approach. With a window of size X and stride 1, we can generate $|r_i| - X$ input sequences of size X from recorded data of size $|r_i|$.

In stage 1, each input sequence generated from recorded data r_i is assigned a same style embedding based on the demonstrator of r_i but a different content embedding. The content embedding represents the task which varies for every input sequence even from the same recorded data because it is dependent on the last position of the wheelchair in the input sequence. Stage 1 learns optimized style and content/task embeddings along with generator parameters by trying to minimize the reconstruction loss of the input sequence.

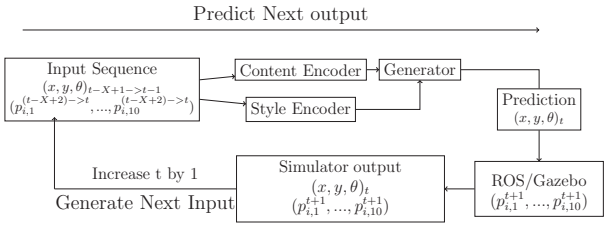


Fig. 3. Auto-regressive prediction using LORD with simulator in the loop. The new input sequence is generated by combining the past sequence and the latest output prediction.

Since at inference time, we do not have all X frames of wheelchair trajectory, as the wheelchair position in the last frame has to be computed, we slightly modify the input and output in Stage 2. We provide only $X-1$ frames of wheelchair trajectory and X frames of the global plan as input during stage 2 training. The generator, however, reconstructs all X frames of trajectory and the global plan. Using this stage 2 model, we can compute the pose of the wheelchair at the next timestep during inference.

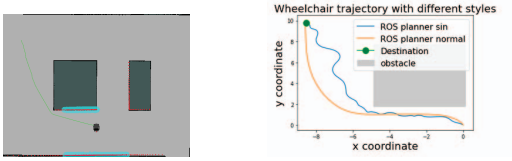
III. Experiments

Since data collected from patients can be noisy and is hard to use, we use synthetic data first to test and prove the concept.

A. Synthetic data generation

We generate wheelchair trajectories using 2 different styles in simulation. One is simply using the local planner to traverse a path from a start position to a goal position. We call it no-style trajectory. For the second style, we add the periodic sinusoidal noise in the angular velocity ω using the following equation :

$$\omega_z = \omega_z + 0.5 * \sin(t_{now} - t_{start}) \tag{4}$$



(a) (b)

Fig. 4. (a) Trajectory generated by Global Planner when the goal is set. (b) Real trajectory with/without sinusoid style simulated by ROS.

Figure 4(b) shows the trajectory generated by 2 different styles. We generate 35 trajectories for each style and 70 trajectories in total. By implementing a sliding window of size 64 and stride length 1, 6994 inputs are generated in total. Out of these, we keep 6792 inputs for training and 202 for testing.

B. Experiment Results

1) Pre-recorded dataset: To predict the future trajectory with style, our framework needs both current position and global planner predictions. However, the global planner predictions recorded in our test data will be valid only if the wheelchair's states match exactly with those in the recorded trajectory, which is not true in most of the cases. To cope with this issue, in our test with the pre-recorded dataset, only the first 64 frames are given as inputs and the model will try to predict both the next positions and next global planner predictions. Since our model is not trained to be a global planner by itself, the robot will never be able to reach the same goal as in the test trajectory, but the desired style should be visible. Figure 5 shows the prediction of both styles on train/test dataset. Predictions based on unseen trajectories from test dataset may become unreliable due to the error accumulation as shown in Figure 5(b), but the sinusoidal style is well maintained throughout the prediction.

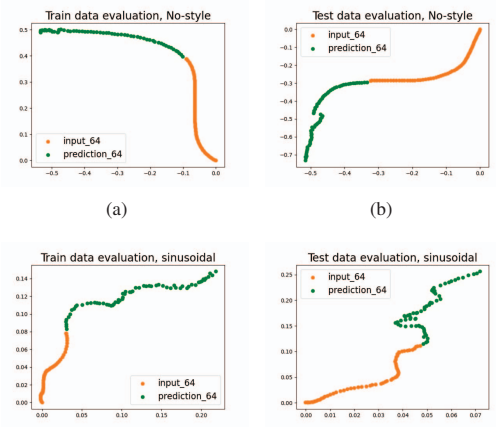


Fig. 5. (a)(b) No-style trajectory prediction based on 64 frames input from train/test dataset. (c)(d) Sinusoidal trajectory prediction based on 64 frames input from train/test dataset.

2) Simulator in the loop: To see if this approach can be deployed online, we test with a simulator in the loop where the positions of the wheelchair generated using our trained model are fed to a simulator to generate the positions for the next timesteps. since the trajectories generated will always differ slightly from each other due to planner and simulator noise, we do not have the ground truth for these experiments. Therefore, we evaluate the resulting trajectories by comparing the difference between the trajectory generated by our model and the synthetic data generator using the 2 styles. As shown in Figure 6, our results show that the trajectories generated by our model are closer to the style for which they were generated.

IV. Conclusion

We have investigated how style and content disentanglement, generally used for disentangling images, can be used to disentangle style from demonstrations for applying it to different wheelchair navigation tasks. Our preliminary results

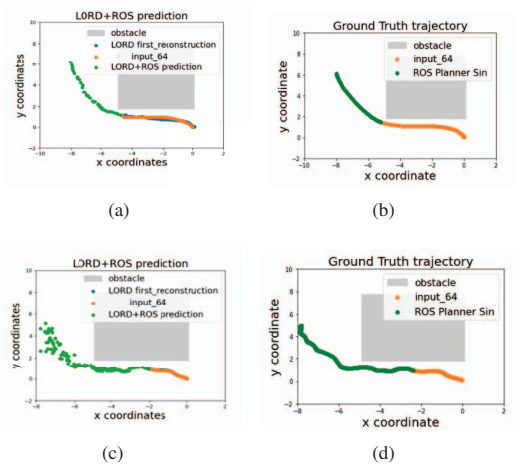


Fig. 6. (a) No-style trajectory predicted by LORD+ROS planner given the first 64 inputs. (b) Ground Truth No-style trajectory generated by data generator. (c) Sinusoidal trajectory predicted by LORD+ROS planner given the first 64 inputs. (d) Ground Truth Sinusoidal trajectory generated by data generator.

on synthetic data show that the learned model can be used to generate the trajectories with a given style online. However, more thorough quantitative testing is still needed. In the future, we will test this model further with more challenging scenarios containing different obstacles and also with real human data from subjects having cerebral palsy, Parkinson's disease etc.

Acknowledgment

This research is supported by A*STAR under its National Robotics Programme (NRP) BAU grant, project titled - Assistive Robotics Programme (Award No: M22NBK0074).

References

[1] D. A. Abbink, T. Carlson, M. Mulder, J. C. F. de Winter, F. Aminravan, T. L. Gibo, and E. R. Boer, "A topology of shared control systems—finding common ground in diversity," *IEEE Transactions on Human-Machine Systems*, vol. 48, no. 5, pp. 509–525, 2018.

[2] Z. Lei, B. Y. Tan, N. P. Garg, L. Li, A. Sidarta, and W. T. Ang, "An intention prediction based shared control system for point-to-point navigation of a robotic wheelchair," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 8893–8900, 2022.

[3] M. Zare, P. M. Kebria, A. Khosravi, and S. Nahavandi, "A survey of imitation learning: Algorithms, recent developments, and challenges," 2023.

[4] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "Amp: Adversarial motion priors for stylized physics-based character control," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–20, 2021.

[5] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, ser. NIPS'16. Red Hook, NY, USA: Curran Associates Inc., 2016, p. 4572–4580.

[6] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet, "Learning latent plans from play," *Conference on Robot Learning (CoRL)*, 2019. [Online]. Available: https://arxiv.org/abs/1903.01973

[7] A. Gabbay and Y. Hoshen, "Demystifying inter-class disentanglement," in *ICLR*, 2020.

[8] H.-Y. Lee, H.-Y. Tseng, J.-B. Huang, M. Singh, and M.-H. Yang, "Diverse image-to-image translation via disentangled representations," in *ECCV*, September 2018.

[9] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *ECCV*, September 2018.