Gesture2Path: Imitation Learning for Gesture-aware Navigation

Catie Cuan[1], Tsang-Wei Edward Lee[2], Emre Fisher[3], Anthony Francis[4], Leila Takayama[5], Tingnan Zhang[2], Alexander Toshev[6], and Sören Pirk[7]

[1]Stanford University [2]Robotics at Google [3]ASML [4]Logical Robotics [5]Hoku Labs [6]Apple ML Research [7]Kiel University

Abstract. As robots increasingly enter human-centered environments, they must not only be able to navigate safely around humans, but also adhere to complex social norms. Humans often rely on non-verbal communication through gestures and facial expressions, and respect body motion and language, when navigating around other people, especially in densely occupied spaces. Consequently, robots also need to be able to interpret body motion as part of solving social navigation tasks. To this end, we present *Gesture2Path*, a novel social navigation approach that combines image-based imitation learning with model-predictive control. We observe the human and their environment with a neural network operating on streams of images, generating point-to-point navigation tasks solved with state-of-the-art model predictive control. We deploy our human-aware policy on real robots and showcase the effectiveness of our approach for the four gestures-navigation scenarios: left/right, follow me, and make a circle. We validated our method based on in-situ ratings of participants interacting with the robots. Our experiments show our method can interpret complex human gestures and use them to generate socially compliant trajectories for navigation tasks.

Keywords: Social navigation, gesture-based interaction, human-robot interaction, imitation learning, model predictive control

1 INTRODUCTION

Situated agents should not only navigate safely around people, but should also abide by social norms and respond to the full gamut of human behavior – including nonverbal communication such as body language, gestures, and facial expressions. A robot solving a navigation task must be able to interpret human behavior and to carefully adjust its actions to be socially compliant. We refer to this form of navigation task as *Nonverbal Social Navigation*. Social robotics research has recently expanded to study respecting personal space [1] and social dynamics [41, 23], socially-acceptable behavior for approaching humans [20], navigation among groups of people [24] and curating large datasets [22].

To solve navigation tasks, many existing approaches rely on point cloud data obtained from LiDAR scanners (or semantic maps [10]) that provide real-time information about the environment, including dynamic objects such as humans. The captured point clouds are projected to 2-D occupancy maps. With these maps existing policy algorithms – such as model predictive control (MPC) – can efficiently solve complex navigation tasks with remarkable success. However,

Fig. 1. *Gesture2Path* is a novel social navigation policy that combines image-based imitation learning with model-predictive control to enable gesture-aware navigation. Here we show our *right* gesture policy: A robot at its start location begins navigating toward its goal (a). It encounters a person that indicates to the robot with a *right* gesture to pass on their right (b). The robot interprets this gesture and drives around the person in the intended manner (c) and then continues towards its goal (d).

while point cloud data is a powerful sensor modality for generating navigation trajectories in the environment, it rarely provides the accuracy needed to interpret intricate human behavior. Conversely, while RGB sensors do not provide the depth information needed to solve navigation tasks, they produce high-resolution images that allow the capture of nuanced gestures and facial expressions.

In this paper, we propose a novel gesture-aware navigation policy based on imitation learning and MPC. We train a sequential neural network that enables us to generate waypoints of navigation trajectories from a sequence of consecutive images. Our goal is to train this network so that it predicts waypoints that are socially compliant and adherent to the gestures of humans interacting with the robot. Once the network is trained, the predicted waypoints are sent to an MPC algorithm to control the robot. The sequential neural network serves as a high-level planner, while the MPC algorithm provides low-level control. This setup provides the benefits of both approaches: the sequential neural network allows us to obtain nuanced information of human gestures from sequences of images, while the MPC algorithm allows us to safely navigate the robot based on laser-scanned point clouds.

To explore the effectiveness of our gesture-aware social navigation policy, we define the four gesture scenarios: Left/Right, Follow Me, and Make a Circle. For each scenario we define the gesture (e.g. pointing with the hand to the left) and the robot behavior (e.g. drive left) so each gesture maps to a specific robot behavior – an example is shown in Fig. 1. We then collected a dataset of expert examples for each scenario by driving the robot with a human operator. A collected trajectory is defined by a sequence of images and a sequence of robot waypoints. We then train the sequential neural network on subsequences of images to predict subsequences of future waypoints. The MPC algorithm then computes the linear and angular velocities to drive from robots' current position to the predicted position of the sequential model.

We validate our gesture-aware social navigation policy based on in-situ experience ratings of humans interacting with the robots. To obtain these ratings we follow a protocol for social navigation policies [34]. Each of our scenarios is defined by a gesture as well as the start and end positions of the trajectory that the robot is supposed to drive along, mirroring the training setup. Additionally, we define a questionnaire based on a five-level Likert scale for each scenario. Once a scenario is completed a participant is asked to provide their ratings for

a specific scenario. We use this setup to compare our *Gesture2Path* policy with a MPC policy as the baseline.

In summary, our contributions are: (1) we introduce *Gesture2Path*, a novel gesture-aware social navigation policy that combines a sequential neural network with MPC; (2) we show that our policy is able to interpret gestures to generate socially-compliant behavior in gesture-navigation scenarios; (3) we define a canonical set of social navigation scenarios with gestures that can be replicated and tested; (4) we use a novel validation protocol for social navigation scenarios to compare our gesture-aware policy with a standard MPC policy.

2 PRIOR WORK

Due to its importance to robotics, research on social navigation has recently garnered considerable attention, spanning a breadth of methods which we cannot comprehensively discuss. For a general overview, interested readers are referred to recent survey papers by Gao and Huang [17], Charalampous et al. [7], Mavrogiannis et al. [28], Kruse et al. [27], Rios-Martinez [35], and Mirsky et al. [29]. The following discussion focuses on approaches to social navigation and human robot interaction with methods closely related to our work.

A variety of researchers have explored gestures for robotic control. For teleoperation, researchers used hand tracking with a Microsoft Kinect device [43] [13], a Leap motion [42], or a camera to direct a robot's end effector. For robot navigation, [37], [38] extracted gestures from a Kinect sensor to generate navigation commands for a teleoperated robot in a non-social setting. [21] used hidden Markov models to detect six gestures with a Kinect sensor and mapped them to hand-crafted robot navigation behaviors. [6] used a wearable gesture detector to control a mobile robot. Finally, [4] and [2] used imitation learning to replicate human gestures on a robot platform.

Researchers have also examined gestures for social communication, such as following [30], pointing [3] or drone-specific motions [32]. Prior work has also aimed to recognize actions based on human-skeletons [8]. [19] modeled musical improvisation as a series of gestures so a robot marimba player could improvise with a human. In [31], researchers designed a gesture framework for humans to collaborate with a co-worker robot; [14] approached a collaborative task between humans and robots using a gesture framework. Other gesture-based human-robot communication systems have been developed for settings where verbal communication may be challenging, such as emergency settings [11] or underwater [9]. Unlike the existing approaches (e.g., for gesture-aware UAV control) that mostly generate unary control commands from the detected gestures, our work predicts trajectories based on a representation that is learned from multi-modal sensor data (RGB images + occupancy maps). Combined with MPC our method is able to generate safe, efficient, and socially-compliant trajectories for solving point-to-point navigation tasks.

3 METHOD

Our main goal is to develop a gesture-aware navigation policy which can guide robots around humans in a socially compliant manner while responding to provided gestures. We define our *Gesture2Path* policy by combining a sequential

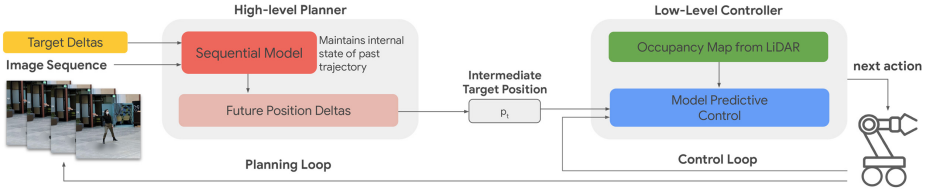

Fig. 2. Overview of *Gesture2Path*: A sequential neural network trained via imitation learning acts as a high-level planner to predict waypoint deltas from image sequences. The predicted waypoint deltas are used to compute a world space waypoint that serves as intermediate target position. MPC uses these intermediate target positions to compute linear and angular velocities to drive the robot.

neural network as a high-level planner with MPC for low level control (Fig. 2). The high-level planner identifies humans and the gestures they perform from images, while the low-level controller navigates the robot from one waypoint to another. The advantage of this setup is that the high-level planner and the low-level controller complement each other. MPC performs well for navigating from one waypoint to another, even around dynamic obstacles, based on point cloud data obtained from the environment. However, MPC fails to detect humans and their gestures as it does not use RGB data. Conversely, the high-level planning neural network is able to detect humans and their gestures from sequences of images, but fails to reliably predict waypoints for the navigation task.

3.1 Gesture-aware Social Navigation

To control a robot in response to a human and their gestures, we employ a sequential neural network [39] that, given a history of observations, outputs a sequence of future positions. We learn this model from demonstrations of successful navigation trajectories of human-robot interactions and show that the network is able to learn to interpret human gestures.

In more detail, we denote by $s = (I, O, p)$ the robot state consisting of an RGB image I from a headmounted camera, an occupancy grid O from a LiDAR sensor, and a robot position p in the world coordinate frame. A demonstration trajectory of length n can be described as a sequence of states: $(s_1, \ldots, s_n)$. An occupancy map is defined as the 2D projection of a point cloud obtained from a LiDAR scanner. Given a history of states, we aim to compute controls that would result in the future robot positions as encoded by the demonstrations. In particular, at step n we utilize a history of k states from the demonstration to predict l future positions:

$$(p_{n+1}, ..., p_{n+l}) = \text{Gesture2Path}(s_n, \ldots, s_{n-k}, g), \tag{1}$$

where g is the goal position of the robot.

In theory, we could use a neural network to directly learn robot control commands. Several approaches do learn direct mappings from pixels to motor controls [25, 26, 33]. However, learning an end-to-end robot control policy remains challenging, as methods tend to require large training datasets and easily overfit to the underlying robot dynamics. Therefore, we employ MPC as a low

level controller that, given an intermediate goal position, drives the robot to this position while avoiding collisions. The MPC algorithm takes care of converting target positions to low level torque controls.

To navigate a robot a MPC controller requires the intermediate next position encoded in the robot frame. Thus, we need to convert the demonstrated robot positions from world frame to a sequence of egocentric positions, each in the frame of the previous position. This can be done by calculating the deltas between subsequent positions: $\delta_i = p_i - p_{i-1}$ for $i = 1, \ldots, l$, which leads to the final model formulation:

$$(\delta_{n+1}, \ldots, \delta_{n+l}) = \text{Gesture2Path}(s_n, \ldots, s_{n-k}, g). \tag{2}$$

Sequential Model Architecture: We train the sequential model by simultaneously embedding subsequences of RGB images I, the occupancy maps O, and the target deltas $\tau = g - p_n$ into a joined latent space. Specifically, our neural network architecture uses 3D convolutional layers to obtain two 64-dimensional embeddings from RGB images and occupancy maps of the input subsequence with a resolution of 256x256 pixels. To combine the target deltas (τ) with the image embeddings we also project them into a 64-dimensional latent space. We then add the embeddings and pass them to three dense layers to obtain a sequence of l position deltas (δ). For the last dense layer we use a linear activation for regression. Fig. 3 shows this network architecture. Our goal for defining a lightweight multi-modal architecture for detecting gestures is to ensure small inference times (< 30ms).

While our policy is agnostic to the choice of the sequential model, we found a common 3D-convolutional neural network similar to [5] provided the best performance for our experiments. We train the network via imitation learning using expert sequences of human-robot interactions captured by manually driving the robot with a human operator (Section 4.2). Instead of using a sequential model to predict sequences of waypoint deltas from sequences of multi-modal input frames (many-to-many), we also experimented with other architectures (many-to-one, one-to-one) to only predict a single waypoint delta. However, we found that a many-to-many architecture provides the best results for our experiments.

We train our sequential neural network on subsequences of the collected expert trajectories (full sequence). The training objective is to predict a sequence of position deltas (with l steps into the future) based on a history of k previous states. The goal is to use the predicted position deltas to compute an intermediate target position (p_t) from the current position (p_n). To train the model we sample all possible subsequences from our dataset of expert trajectories.

Action Parameterization: We use MPC for low-level robot control. Specifically, we formulate the goal following as an optimization problem and use an iterative linear quadratic regulator (iLQR) solver, Trajax [16], that minimizes a hand-engineered cost function, similar to [40, 18, 12]. The optimization problem was solved at each control step and the predicted action at the first step (i.e. velocity command) is sent to the robot. The inputs to MPC include the occupancy grid and relative goal position; its cost function combines time-weighted

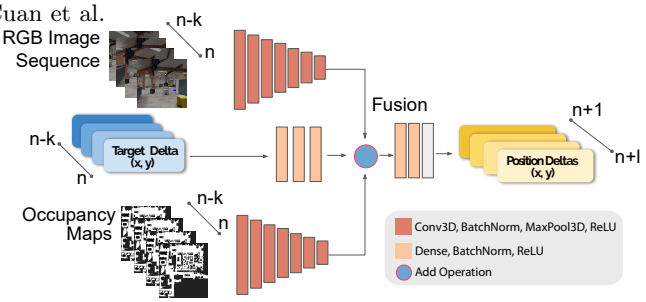

Fig. 3. We use a 3D CovNet neural network architecture that jointly embeds sequences of RGB images I, occupancy maps O, and target deltas τ into a fused embedding space. The network is trained so as to predict sequences of position deltas that we use to generate an intermediate goal position for an MPC algorithm.

goal penalties, margin-offset collision penalties, and a weighted control penalty; and its action space is linear and angular velocities. This MPC policy provides smooth navigation and fast reaction times for differential drive robots and is superior in our tests to a re-implementation of the well-performing reinforcement learning policy in [15].

We use MPC in two ways: first we establish a baseline, where we use the algorithm to navigate from a start to a goal position without additional visual inputs. Second, we use MPC for our gesture-aware policy by tracking waypoints that are generated by the sequential neural network. In both cases, the action space are linear and angular velocities.

For gesture-aware navigation, we use the sequential neural network described in Section 3.1 to predict position deltas from a sequence of previous states. From the position deltas (δ) we compute an intermediate target position $p_t = p_n + \sum_{i=n+1}^{i=n+l} \delta_i$ for MPC based on the current position p_n. The MPC algorithm then computes an optimal path from the current position p_n to the intermediate target position p_t. At every time step we collect the robot state s_n. To generate gesture-aware trajectories we use the neural network to make predictions every k time steps (Fig. 2, Planning Loop), while the MPC algorithm runs at a frequency of 10 Hz (Fig. 2, Control Loop). With $k = 10$, this means that the neural network is used to predict a new intermediate target position every 10 time steps, while the MPC algorithm continuously generates control commands to drive toward the intermediate target position. The sequential network runs at 1 Hz to generate waypoints for MPC, which runs at 10 Hz. This design choice ensures MPC can plan a path from the robot's current position to the predicted future target. If the sequential network took longer than 10 Hz, the robot would stall, so we aim for small inference times. Note that the number of previous states (defined by k) used to make predictions with the neural network need not be the same as the number of predicted position deltas (defined by l). The neural network may be triggered to predict a new intermediate target before the robot reaches the previous one. This ensures that the gesture-aware policy is susceptible to changes in the observed gestures.

Moreover, as illustrated in Fig. 4, the trajectory represented by the predicted position deltas of the sequential neural network and the MPC-computed

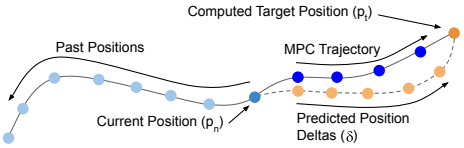

Fig. 4. Once the intermediate target position p_t has been computed from the predicted position deltas, we use MPC to compute an optimal trajectory from the current position p_n to the target position. Note that the trajectory represented by the predicted position deltas and the computed MPC trajectory may diverge – we do not want to over-constrain the MPC algorithm to find an optimal trajectory.

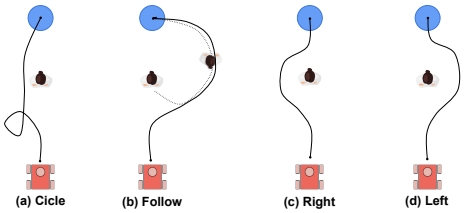

Fig. 5. We test *Gesture2Path* with three policies to respond to four gestures: circle (a), follow (b), right (c), and left (d). For each scenario we only define the gesture (e.g. pointing with the hand to the left), the corresponding expected robot behavior (e.g. drive left), and the start and end positions of the robot trajectory. The robot is tasked to navigate from its start position to a goal area (blue circle). A person is standing in the center of the room and interacts with the robot by performing different gestures to initiate the robots' response.

trajectory may diverge. While we want the robot to closely follow the predicted position deltas to implement gesture-aware behavior, we do not want to over-constrain MPC to find an optimal trajectory from the current position to the intermediate target position. The planning loop based on the sequential neural network provides general direction while considering observed gestures, and the control loop based on MPC ensures safe navigation. Similar to other imitation learning approaches, our method for detecting body motions and gestures requires a dataset of samples that faithfully represent the conditions the policy is exected to operate in. However, in situations where the high-level planner fails – e.g., when the current sequence of observation does not match the training data distribution – the MPC algorithm still generates safe and efficient trajectories.

4 EXPERIMENT SETUP

The robots we use are 7 DOF mobile manipulators with a single arm and a rectangular base. Each robot is equipped with a head-mounted camera and a LiDAR sensor. Robots also have access to an existing static map of their environment to assist with localization and point-to-point navigation.

4.1 Gesture Scenarios

To test the capabilities of *Gesture2Path*, we use the protocol proposed in [34] to define the four gesture scenarios (illustrated in Fig. 5): Circle, Follow, and Left/Right:

Circle: In the make a circle scenario, the demonstrator would stand in the center of the room and raise their right hand straight up to the ceiling. The robot operator then drives the robot to make a counterclockwise circle before passing the human on their left (Fig. 5 a).

Follow: In the follow me scenario, the demonstrator would lift both hands out to the side while keeping their feet together, making a "T" shape. After holding this shape for 2 seconds, they would turn over their left shoulder and walk along a wide arc prior to passing through the end position region. Once the gesture has been observed the robot operator would then drive the robot so as to follow the demonstrator (Fig. 5 b).

Left/Right: In the left/right scenario, the demonstrator would stand in the center of the room and put their arm out to the left or right while leaning in that direction. The robot operator would then drive the robot to pass along the side of the human's outstretched arm (Fig. 5 c, d).

We selected these gestures based on the following criteria: (1) the robot solved a navigation task beginning at one side of the room and ending at the other; (2) the robot moved continuously from beginning to end without pausing or stopping; (3) the gesture instructions would be simple and easy to teach the experiment participants; (4) the gestures would correspond to clear changes in robot behavior; (5) the gestures required the robot to react to a single participant. While the gestures used for the Circle and Follow scenarios were distinct, the Left and Right gestures represented more subtle variations for the sequential model to disentangle.

4.2 Data Collection and Training

To train our *Gesture2Path* policy we collected a dataset of 277 trajectories (40 for follow, 30 for circle, 207 left and right). Each trajectory represents a 20-40 second sequence that includes a human-robot interaction. The human performs one of the four defined gestures, while the robot solves a point-to-point navigation task from a start to a goal position. All training data was collected in the real world in a large, open atrium space over several different days and times.

To collect our dataset we used the following procedure: an expert robot operator would connect to the real robot using a Logitech F710 gamepad. The robot operator would stand behind the robot and follow it while driving it forward. The gesture demonstrator would begin by standing in the middle of the room. Once the teleoperated robot started to move, the gesture demonstrator would perform the gesture for 2-3 seconds. Depending on the gesture scenario, the demonstrator would either stay in place or move appropriately.

We split the collected trajectories into 90% training and 10% validation data. We then trained our sequential neural network on subsequences ($k = 10$, $l = 10$, for most of our experiments) of RGB images and occupancy maps. For both data modalities we use the raw frames normalized into a [-1, 1] range. We used the Adam optimizer with a standard $l2$ loss and selected checkpoints of the network for our policies based on the lowest $l2$ error measured on the validation dataset. The training of our network usually converges after 120 epochs, which usually takes 6 hours of training on a single V100 GPU.

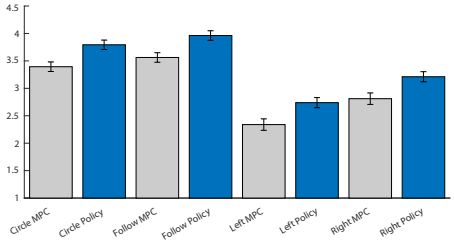

Fig. 6. Means and standard errors for all four questions for policy and gesture condition. MPC is in grey and *Gesture2Path* is blue. Participants rated questions on a scale of 1 Strongly Disagree to 5 Strongly Agree. Zero was reserved for Not Applicable. *Gesture2Path* rated higher than MPC on all questions. We performed a paired t-test on the MPC/*Gesture2Path* ratings for each gesture (i.e. comparing circle MPC to circle *Gesture2Path*). All of these tests were statistically significant with a p-value $< .01$. *Gesture2Path* performs better than MPC for each of these gesture scenarios.

4.3 Experiment Design

To validate the effectiveness of our gesture-aware policies we conducted a user-study to obtain in-situ experience ratings. In the following, we discuss the setup of our experiment for our *Gesture2Path* polices as well as for the MPC baseline.

User Study Design: We designed a within-subjects experiment where participants performed the four gestures while the robot was running either *Gesture2Path* or a baseline of the unmodified MPC policy. Note this baseline MPC policy is itself a highly performant navigation policy, exceeding in our internal tests a reimplementation of [15]. The study occurred in the same location where the training data was collected.

Participants arrived at the location and filled out an introductory survey asking about their dominant hand and number of months they had been working near these robots. The participants then performed each of the four gestures a total of 70 different times in the center of the room. Each the trials consisted of the robot crossing from one side of the room to the other; the human participants would perform the gesture in the center of the room (and during the follow scenario, walking to the edge of the room). Once the trial was completed the participant would immediately answer four questions about the robot's performance. The questionnaire was defined based on a 5-level Likert scale [34]. Each of the four questions were rated on a scale of 1-5, 1 being *Strongly Disagree* and 5 being *Strongly Agree*. Participants had the option to answer 0 if the question did not apply. The questions were: (1) The robot maintained a safe distance at all times (Safe); (2) It was clear what the robot wanted to do (Clarity); (3) The robot responded correctly to my gesture (Correct); (4) The robot paid attention to what I was doing (Trust).

We ran the gesture trials 10 times for each of our *Gesture2path* policies (Follow, Circle, Left/Right). For the MPC baseline we ran 10 trials for the Follow and Circle scenarios and 5 trials for the Left and Right scenarios. We hypothesized that the *Gesture2Path* policy would score more highly on questions 2-4 (Clarity, Correct, Trust) and lower on question 1 (Safe). We hypothesized that it would score lower on Safety since the robot's actions would be more varied. We ran-

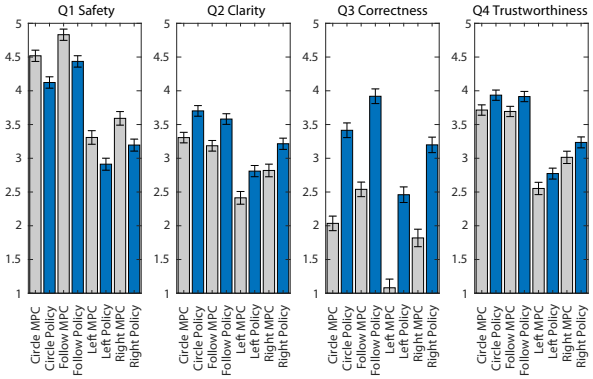

Fig. 7. All averages and standard errors for each question for each gesture, *Gesture2Path* is in blue, MPC is in grey. Question 3 (Correctness) shows the largest difference between *Gesture2Path* and MPC. A repeated measures ANOVA on Question 3 showed a statistically significant difference.

domized policy ordering for each participant. For example, one participant may have completed MPC Left/Right, *Gesture2Path* Left/Right, MPC Follow, Policy Follow, MPC Circle, Follow Circle. After the gesture trials, the participants filled out a closing survey asking about their demographics and work experience. The experiment took approx. 90 minutes to complete.

Participants: 12 individuals volunteered to participate in the user study, including 2 women and 10 men. 8 participants identified as Asian, 2 as White, 1 as Hispanic/Latino, and 1 Middle Eastern. 3 participants were ages 18-24, 8 were ages 25-34, and 1 declined to state. All participants listed their *right hand* as the dominant hand and all participants had previously used the robot.

5 RESULTS

All participant ratings for each gesture and condition (*Gesture2Path* and MPC) were averaged, as shown in Fig. 6. A paired t-test was performed for each gesture comparing policy and MPC group and all differences are statistically significant with a p-value $< .01$. Therefore, we conclude that the *Gesture2Path* policy overall performs better than MPC for all four gesture scenarios (again, see Fig. 6). We ran a repeated measures ANOVA on each question; the effect of *Gesture2Path* vs MPC on Question 3 (Correctness) showed a statistically significant difference: F(within groups df) = 115.563, p-value = .059. Separate repeated measures ANOVAs were performed to compare the effect of *Gesture2Path* vs MPC on Question 1 (Safety), Question 2 (Clarity) and Questions 4 (Trustworthiness). Fig. 6 shows that *Gesture2Path* (blue bars) is higher in all cases for Question 2 (Clarity) and Question 4 (Trustworthiness), while MPC (grey bars) is higher for Question 1 (Safety). However, for Questions 1, 2, and 4, the ANOVA did not show statistical significance. Therefore, the improvement that elevated *Gesture2Path* over MPC was the policy's correct response to the human gesture (Question 3). All paired t-tests and ANOVA analysis were performed with a Bonferroni correction (Fig. 7). Baseline MPC performed better than Gesture2Path for all four gestures on Question 1 (Safety). The higher safety rating follows from baseline MPC's tendency to give humans a large amount of space.

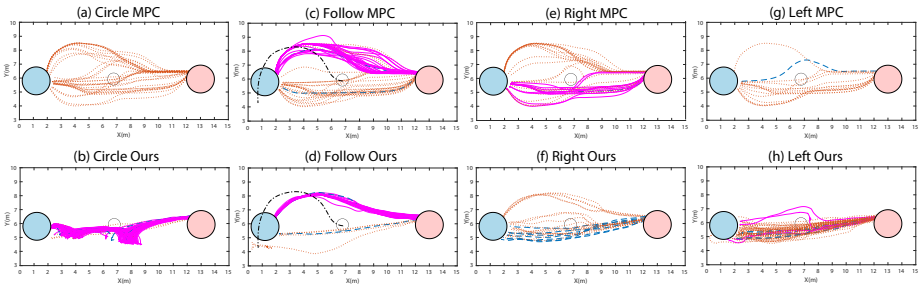

Fig. 8. 2D plots of robot trajectories and participant ratings (for Question 3) for all four gestures with *Gesture2Path* (Ours) and MPC. The robot starts in the red circle and must navigate to the blue dot. The human is the black circle in the center. For Follow plots, there is an additional black alternating dot-dash line to indicate the rough path that the human walks after they make the gesture in the center of the room. Favorable participant ratings (4 or 5 on a 5 point scale) are solid lines in magenta. Neutral ratings (3 on a 5 point scale) are the dashed line in blue. Negative ratings (1 or 2 on a 5 point scale) are the dotted lines in red. The MPC trajectories in the first row are all similar regardless of the gesture. The *Gesture2Path* policy robot trajectories in b) Circle and d) Follow are distinctive, correct, and consistent, with ratings capturing this effect. The *Gesture2Path* policy does not perform as well for Right and Left sides.

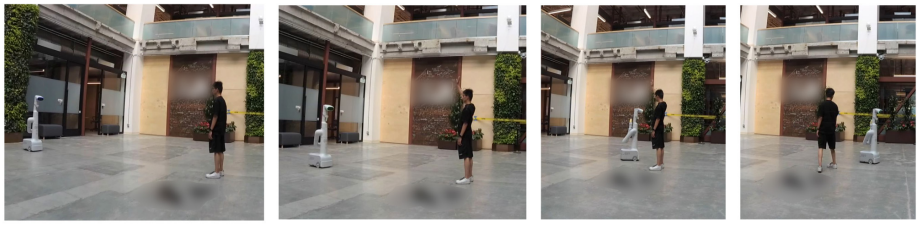

Fig. 9. MPC policy: As no image sequences are used, the policy is not able to generate trajectories in response to gestures. While the performed gesture is supposed to make the robot drive a circle, it only passes the human participant on their right side.

The *Gesture2Path* policies took longer than MPC in all cases. The average durations for each gesture in *Gesture2Path* | MPC order were: Circle (56 | 35), Follow (45 | 38), Left (45 | 38), and Right (46 | 33). For Circle and Follow, this was likely due to the gesture response trajectory requiring a longer distance for the robot to travel. For the Left/Right Policy, the additional time was likely due to the wider distance needed by the gesture. The *Gesture2Path* policy is slower (lower velocity) but more smooth (lower acceleration and jerk), as shown in Table I. This is logical as the policy imitates the human operator driving the robot during data collection. Paired t-tests for duration, velocity, acceleration, and jerk were performed on each Gesture2Path and MPC pairing for each gesture and the results were statistically significant with all p-values < 0.01. A comparison of human-robot distances of MPC and *Gesture2Path* for the Follow gesture showed the average *Gesture2Path* distance at 2.59m with 0.86 SD, where the average MPC distance was 2.78m with 1.11 SD. Our method is well within the human-

Table 1. Velocity, Acceleration, and Jerk for each Policy and Gesture Combination, Gesture2Path is abbreviated to "G2P".

Policy w/ Gesture	Velocity(m/s)	Acceleration(m/s^2)	Jerk(m/s^3)
G2P L/R	0.20555	0.0044	0.0321
MPC L/R	0.47382	0.0055	0.0390
G2P Follow	0.20977	0.0040	0.0293
MPC Follow	0.45282	0.0070	0.0503
G2P Circle	0.25233	0.0066	0.0454
MPC Circle	0.44694	0.0076	0.0551

robot proxemics social space [36] of 3.6m radius and the SD is lower, meaning the robot follows at a more consistent distance over the episode.

In Fig. 8 we show 2-dimensional plots of the robot navigating from the start to end positions. The positive participant ratings (4 or 5 out of 5) for Question 3 (Correctness) are shown in solid magenta. The neutral rating (3 out of 5) is show in dotted black. The negative ratings (1 or 2 out of 5) are shown in dotted red. These plots demonstrate that the participants have a clear understanding of the robot's correct response as the higher ratings correspond with the robot's correct behavior. The trajectories in Fig. 8 (e) and (g) differ because as the person gestures in that direction, their arm and hand create an obstacle that changes the distribution of the obstacles in the scene. This obstacle affects the MPC plan. We created two quantitative metrics in order to measure the correctness of the response to Left/Right and the Circle gesture. For the Left/Right gesture, we compute a binary success as to whether the robot spent more than 52% of the episode on the correct side. In this case, the *Gesture2Path* policy is 34% accurate while the MPC is 12% accurate. The Circle binary success was defined as the robot performing a change in orientation greater than 180 degrees. In this case, the *Gesture2Path* policy is 90% accurate while the MPC is 0% accurate.

Our *Gesture2Path* policy is able to generate socially compliant trajectories in response to the performed gestures. In contrast, as the MPC policy is not using image sequences as input, it is not able to generate trajectories based on the performed gestures – an example is shown in Fig. 9. Given the stochasticity of the scene, MPC may occasionally fail to generate a path, however *Gesture2Path* may fail due to MPC failing to generate a path, the sequential network failing to output correct target waypoint deltas, or failing to generate a correct behavior given the gesture done by the participant.

References

1. Althaus, P., Ishiguro, H., Kanda, T., Miyashita, T., Christensen, H.: Navigation for human-robot interaction tasks. In: IEEE ICRA. vol. 2, pp. 1894–1900 Vol.2 (2004). https://doi.org/10.1109/ROBOT.2004.1308100
2. Bandera-Rubio, J.e.a.: Vision-based gesture recognition in a robot learning by imitation framework (2010)
3. Barbed, O.L., Azagra, P., Teixeira, L., Chli, M., Civera, J., Murillo, A.C.: Fine-grained pointing recognition for natural drone guidance. In: CVPR Workshops. pp. 1040–1041 (2020)
4. Calinon, S., Billard, A.: Learning of gestures by imitation in a humanoid robot. Tech. rep., Cambridge University Press (2007)

5. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. pp. 4724–4733 (07 2017). https://doi.org/10.1109/CVPR.2017.502
6. Chaithanya, D., Gowda, D., Balakrishna, D., Gowda, D.M., Divyashree, M.: A survey on hand gesture controlled rover (2022)
7. Charalampous, K., Kostavelis, I., Gasteratos, A.: Recent trends in social aware robot navigation: A survey. Robotics and Autonomous Systems **93**, 85–104 (2017)
8. Chi, H., Ha, M., Chi, S., Lee, S., Huang, Q., Ramani, K.: Infogcn: Representation learning for human skeleton-based action recognition. In: CVPR. pp. 20186–20196 (2022)
9. Chiarella, D., Bibuli, M., Bruzzone, G., Caccia, M., Ranieri, A., Zereik, E., Marconi, L., Cutugno, P.: A novel gesture-based language for underwater human–robot interaction. J. mar. sci **6**(3), 91 (2018)
10. Cosgun, A., Christensen, H.I.: Context-aware robot navigation using interactively built semantic maps. Paladyn, Journal of Behavioral Robotics **9**(1), 254–276 (2018)
11. De Cillis, F., Oliva, G., Pascucci, F., Setola, R., Tesei, M.: On field gesture-based human-robot interface for emergency responders. In: IEEE SSRR. pp. 1–6 (2013)
12. Di Carlo, J., Wensing, P.M., Katz, B., Bledt, G., Kim, S.: Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. In: IROS. pp. 1–9 (2018). https://doi.org/10.1109/IROS.2018.8594448
13. Du, G., Zhang, P., Mai, J., Li, Z.: Markerless kinect-based hand tracking for robot teleoperation. Int. J. Adv. Robot **9**(2), 36 (2012)
14. Ende, T., Haddadin, S., Parusel, S., Wüsthoff, T., Hassenzahl, M., Albu-Schäffer, A.: A human-centered approach to robot gesture based communication within collaborative working processes. In: IEEE RSJ. pp. 3367–3374 (2011)
15. Francis, A., Faust, A., Chiang, L., Hsu, J., Kew, J.C., Fiser, M., Lee, E.: Long-range indoor navigation with prm-rl. IEEE T-RO **36**(4), 1115–1134 (2020)
16. Frostig, R., Sindhwani, V., Singh, S., Tu, S.: trajax: differentiable optimal control on accelerators (2021), http://github.com/google/trajax
17. Gao, Y., Huang, C.M.: Evaluation of socially-aware robot navigation. Frontiers in Robotics and AI **8** (2022). https://doi.org/10.3389/frobt.2021.721317, https://www.frontiersin.org/article/10.3389/frobt.2021.721317
18. Hertneck, M., Köhler, J., Trimpe, S., Allgöwer, F.: Learning an approximate model predictive controller with guarantees. IEEE Control Systems Letters **2**(3), 543–548 (2018). https://doi.org/10.1109/LCSYS.2018.2843682
19. Hoffman, G., Weinberg, G.: Gesture-based human-robot jazz improvisation. In: ICRA. pp. 582–587. IEEE (2010)
20. Huang, C.M., Iio, T., Satake, S., Kanda, T.: Modeling and controlling friendliness for an interactive museum robot (07 2014). https://doi.org/10.15607/RSS.2014.X.025
21. Iocchi, L., Bonanni, M.: Person-tracking and gesture-driven interaction with a mobile robot using the kinect sensor (2011)
22. Karnan, H., Nair, A., Xiao, X., Warnell, G., Pirk, S., Toshev, A., Hart, J., Biswas, J., Stone, P.: Socially compliant navigation dataset (scand): A large-scale dataset of demonstrations for social navigation. IEEE robot. autom. lett. pp. 1–8 (01 2022). https://doi.org/10.1109/LRA.2022.3184025
23. Kato, Y., Kanda, T., Ishiguro, H.: May i help you? design of human-like polite approaching behavior. In: HRI. p. 35–42. ACM (2015)
24. Katyal, K.D., Popek, K., Hager, G.D., Wang, I.J., Huang, C.M.: Prediction-based uncertainty estimation for adaptive crowd navigation. In: Artificial Intelligence in HCI (AI-HCI). p. 353–368 (2020). https://doi.org/10.1007/978-3-030-50334-5_24

25. Kim, H., Jordan, M., Sastry, S., Ng, A.: Autonomous helicopter flight via reinforcement learning. In: Thrun, S., Saul, L., Schölkopf, B. (eds.) NIPS. vol. 16. MIT Press (2003)
26. Kohl, N., Stone, P.: Policy gradient reinforcement learning for fast quadrupedal locomotion. In: IEEE ICRA. vol. 3, pp. 2619–2624 Vol.3 (2004). https://doi.org/10.1109/ROBOT.2004.1307456
27. Kruse, T., Pandey, A.K., Alami, R., Kirsch, A.: Human-aware robot navigation: A survey. Rob Auton Syst **61**(12), 1726–1743 (2013)
28. Mavrogiannis, C., Baldini, F., Wang, A., Zhao, D., Trautman, P., Steinfeld, A., Oh, J.: Core challenges of social robot navigation: A survey (2021)
29. Mirsky, R., Xiao, X., Hart, J., Stone, P.: Prevention and resolution of conflicts in social navigation–a survey. arXiv preprint arXiv:2106.12113 (2021)
30. Naseer, T., Sturm, J., Cremers, D.: Followme: Person following and gesture recognition with a quadrocopter. In: IROS. pp. 624–630. IEEE (2013)
31. Neto, P., Simão, M., Mendes, N., Safeea, M.: Gesture-based human-robot interaction for human assistance in manufacturing. J. Adv. Manuf. Technol. **101**(1), 119–135 (2019)
32. Perera, A., Wei Law, Y., Chahl, J.: Uav-gesture: A dataset for uav control and gesture recognition. In: ECCV Workshops. pp. 0–0 (2018)
33. Peters, J., Schaal, S.: Reinforcement learning of motor skills with policy gradients. Neural Networks **21**(4), 682–697 (2008). https://doi.org/https://doi.org/10.1016/j.neunet.2008.02.003, https://www.sciencedirect.com/science/article/pii/S0893608008000701
34. Pirk, S., Lee, E., Xiao, X., Takayama, L., Francis, A., Toshev, A.: A protocol for validating social navigation policies. ICRA, Workshop: Social Robot Navigation: Advances and Evaluation (2022)
35. Rios-Martinez, J., Spalanzani, A., Laugier, C.: From proxemics theory to socially-aware navigation: A survey. Int. J. Soc. Robot. **7**(2), 137–153 (Apr 2015). https://doi.org/10.1007/s12369-014-0251-1, https://doi.org/10.1007/s12369-014-0251-1
36. Rios-Martinez, J., Spalanzani, A., Laugier, C.: From proxemics theory to socially-aware navigation: A survey. International Journal of Social Robotics **7**(2), 137–153 (2015)
37. Saha, S., Lahiri, R., Konar, A., Lekova, A., Nagar, A.: A novel approach to tsk model based gesture driven robot movement. In: FUZZ-IEEE. pp. 1–6. IEEE (2017)
38. Saha, S., Lahiri, R., Konar, A., Nagar, A.: Gesture driven remote robot manoeuvre for unstructured environment. In: IEEE SSCI. pp. 1793–1800. IEEE (2018)
39. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. NIPS **27** (2014)
40. Tassa, Y., Mansard, N., Todorov, E.: Control-limited differential dynamic programming. In: ICRA. pp. 1168–1175 (2014)
41. Truong, X.T., Ngo, T.D.: "to approach humans?": A unified framework for approaching pose prediction and socially aware robot navigation. IEEE TCDS **10**(3), 557–572 (2018). https://doi.org/10.1109/TCDS.2017.2751963
42. Zhang, W., Cheng, H., Zhao, L., Hao, L., Tao, M., Xiang, C.: A gesture-based teleoperation system for compliant robot motion. Applied Sciences **9**(24), 5290 (2019)
43. Zhang, Z.: Microsoft kinect sensor and its effect. IEEE multimedia **19**(2), 4–10 (2012)