Appendix

A Differentiable MPC

We provide some additional details regarding the learnable MPC, including the structure of the static cost and differentiation of the optimal solution w.r.t. the parameters of the cost neural networks.

Static Cost Function: Recall that the structure of our cost function inside the model predictive controller takes the form $\bar{c}(\mathbf{x}, \mathbf{u}, t) + c_{\text{learn}}$ where $c_{\text{learn}}(\mathbf{x}, \mathbf{u}; C_0, \theta)$ is the context-dependent residual cost. Let $\mathbf{u} := (\mathbf{u}_v, \mathbf{u}_\omega)$ be the 2D control vector consisting of the body-aligned forward speed $\mathbf{u}_v$ and turn rate $\mathbf{u}_\omega$, and let $\mathbf{x} := (\mathbf{p}, \phi)$ be the state vector consisting of the 2D plane position $\mathbf{p}$ and orientation ϕ. The static cost function takes the form:

$$\bar{c}(\mathbf{x}, \mathbf{u}, t) := \sum_{i=1}^{6} w_i \bar{c}_i(\mathbf{x}, \mathbf{u}, t) \tag{10a}$$

$$\bar{c}_1(\mathbf{x}, \mathbf{u}, t) := \mathbb{I}_{\mathbf{u}_v \geq 0} |\mathbf{u}_v|^4, \quad \bar{c}_2(\mathbf{x}, \mathbf{u}, t) := \mathbb{I}_{\mathbf{u}_v < 0} |\mathbf{u}_v|^4, \quad \bar{c}_3(\mathbf{x}, \mathbf{u}, t) := |\mathbf{u}_\omega|^4 \tag{10b}$$

$$\bar{c}_4(\mathbf{x}, \mathbf{u}, t) := \sum_{j=1}^{n_c} \text{ReLU}^2(\text{margin} - d_j(\mathbf{x})) \tag{10c}$$

$$\bar{c}_5(\mathbf{x}, \mathbf{u}, t) := \bar{w}_T(t) \|\mathbf{p} - \mathbf{g}\|^2, \qquad \bar{c}_6(\mathbf{x}, \mathbf{u}, t) := \bar{w}_T(t)(1 - \cos(\phi - g_\phi)) \tag{10d}$$

$$\bar{w}_T(t) := \begin{cases} \dfrac{1}{T(1 + w_T)} & \text{if } t < T \\ \dfrac{T w_T + 1}{T(1 + w_T)} & \text{if } t = T. \end{cases} \tag{10e}$$

where $d_j(\mathbf{x})$ is the signed distance from the j^{th} collision-point on the robot body to the nearest obstacle, margin is a user-set margin threshold, $\mathbf{g} \in \mathbb{R}^2$ is a 2D goal position, g_ϕ is a goal orientation, and $\{w_i\}_{i=1}^6, w_T$ are a set of non-negative weights. Thus, the static/hand-engineered cost function consists of an asymmetric control penalty, margin-offset collision penalty, and a time-weighted goal-reaching penalty.

Tracking MPC: The final policy action is determined as the solution to a secondary, non-learnable MPC problem (termed "tracking MPC"), taking the form of (1) but with the cost comprising of only the static portion, i.e., $\bar{c}$. As defined above, this cost features a goal-reaching penalty. For the "higher-level" MPC problem (i.e., the problem solved by either `Performer-MPC` or `RMPC`), this goal is determined from problem context, e.g., a global waypoint. For the "tracking MPC" problem, this goal is set as an intermediate state extracted from the optimal state trajectory solution to the "higher-level" MPC problem. In the `EP` case, this goal is directly output by the `EP` policy. Using such a dual/tracking-MPC structure allows a more fair comparison between the three policies since the lowest-level control action is output in an identical manner.

Jacobian of Optimal MPC solution: The key tool for computing the desired Jacobian is the implicit function theorem, stated below (note: we suppress the dependence of the MPC cost function J_c on the context C_0 for readability):

Theorem A.1 (Implicit Function Theorem)**.** *If in the neighborhood of* $(\mathbf{u}^\star, \theta_k)$ *where* $\nabla_{\mathbf{u}} J_c(\mathbf{u}^\star, \theta_k) = 0$*, the Hessian* $\nabla^2_{\mathbf{u}} J_c(\mathbf{u}^\star, \theta_k)$ *is non-singular, then we have:*

$$\partial_\theta \mathbf{u}^\star(\theta_k) = -\left[\nabla^2_{\mathbf{u}} J_c(\mathbf{u}^\star, \theta_k)\right]^{-1} \nabla^2_{\theta, \mathbf{u}} J_c(\mathbf{u}^\star, \theta_k) \tag{11}$$

The Hessian term above need not be explicitly materialized, since by chaining Eqns 4 and 11, the VJP can be efficiently calculated as,

$$\nabla_\theta J_l(\mathbf{u}^*(\theta_k)) = -\left[\left[\nabla^2_{\mathbf{u}} J_c(\mathbf{u}^\star, \theta_k)\right]^{-1} \nabla_{\mathbf{u}} J_l(\mathbf{u}^*(\theta_k))\right]^T \nabla^2_{\theta, \mathbf{u}} J_c(\mathbf{u}^\star, \theta_k) \tag{12}$$

The term inside the large square brackets may be computed as the solution to the following quadratic problem:

$$\underset{\delta\mathbf{v}:=(\delta\mathbf{v}_0,\ldots,\delta\mathbf{v}_{T-1})}{\arg\min} \quad \frac{1}{2}\delta\mathbf{v}^T\nabla^2_{\mathbf{u}}J_c(\mathbf{u}^\star,\theta_k)\delta\mathbf{v} + \delta\mathbf{v}^T\nabla_{\mathbf{u}}J_l(\mathbf{u}^*(\theta_k)),$$

which in turn decomposes into a TV-LQR problem [87] (Thm. A.1). Finally, the dot product with $\nabla^2_{\theta,\mathbf{u}}J_c(\mathbf{u}^\star,\theta_k)$ may be computed by differentiating the co-state equations associated with the gradient $\nabla_{\mathbf{u}}J_c(\mathbf{u}^\star,\theta_k)$.

B Speed studies over various Performers' architectures

We run speed studies over various Performer-MPC variants to choose the most suitable one for on-robot deployment characterized by strict latency constraints. The tests were run for 100×100 resolution images and two architecture sizes: large and medium (details below).

Large and medium architecture: The large architecture consists of $l = 6$ layers and $h = 3$ heads with $\mathrm{mlpdim} = 1024$. The medium architecture consists of $l = 3$ layers and $h = 1$ head with $\mathrm{mlpdim} = 64$. Both apply GELU nonlinearity in the MLP-layers. For 100×100 images, the large architecture has $24,581,732$ parameters, and the medium architecture has $8,334,884$ parameters.

Tested Performer variants: We test 13 different Performer versions: one Performer-ReLU and 12 Performer-Exps. For Performer-Exps, we test both redrawing and no-redrawing, as well as a range of different random projections (rps) including 8, 16, 32, 64, 128, and 256.

Results: Our first set of results with patch size 5×5 is presented in Table 3 (we report there wall-clock time). To obtain the desired speed $< 0.1\mathrm{sec}$ per MPC call with the average number of MPC iterations around $\mathrm{nb} = 10$, we look for time per MPC-iteration $< 10\mathrm{ms}$. Because this requirement is satisfied by the medium-size Performer-ReLU variant, we deploy it as our vision backend on-robot. We then study the inference speed of this variant for different patch sizes and note that we can run it efficiently on robot with almost pixel-to-pixel (1×1 patch size) attention resolution (for completeness, we include Table 4 here again, where we report CPU-times to accurately distill time taken by the MPC from other factors such as I/O time).

C Experimental Setup

We further provide details about our experimental setup.

C.1 Robot Setup

Our robot can move at a linear speed between $[-0.8, 0.8]$m/s, and can rotate at an angular speed between $[-1.2, 1.2]$rad/s. The on-board software stack can perform SLAM (Simultaneous Localization And Mapping) and generate at each time step an occupancy grid with 0.05×0.05m cells (occupied or free) around the robot within $[-7.5, 7.5]$m of range for both x and y axis. For most experiments such as in tight spaces and blind corners, we clip the occupancy grid to $[-2.5, 2.5]$m (100×100 cells) since there is enough local information to make navigation decisions. Only for the pedestrian obstruction scenario we reduce the occupancy grid to $[-4.5, 4.5]$m (180×180 cells) so the human can be detected early to leave enough decision making time.

C.2 Data Collection for Doorway Traversal

To learn to avoid local minima in the doorway traversal scenario, we use an artificial expert to generate 2000 training episodes. Each episode is generated by (1) randomly sampling a start configuration on one side of the door and a goal position on the other side, (2) running a coarse Dijkstra's search to generate sparse global way points leading from the start through the doorway to the goal, and (3) feeding these way points on by one (i.e., using the solution of the last way point as the initial condition for the MPC solve for the next one) to an off-line MPC planner with 100 planning horizon and 200 maximum iterations (instead of 20 horizon and 20 iterations of the online MPC deployed on our robot). Such a heavy-duty planner requires too much computation to run onboard our robot.

Table 3: Full speed ablation tests for different variants of Performers with 100×100 input images. The architecture deployed on the real robot is denoted in **bold** (Extension of Tab. 1).

model size	Performer type	redraw	nb rps	wall-clock time per MPC-iteration [ms]
medium	**ReLU**	**N/A**	**N/A**	**8.3**
medium	Exp	False	8	21.5
medium	Exp	False	16	21.6
medium	Exp	False	32	22.2
medium	Exp	False	64	23.6
medium	Exp	False	128	24.7
medium	Exp	False	256	26.8
medium	Exp	True	8	73.6
medium	Exp	True	16	67.6
medium	Exp	True	32	74.0
medium	Exp	True	64	75.0
medium	Exp	True	128	79.0
medium	Exp	True	256	81.6
large	ReLU	N/A	N/A	13.8
large	Exp	False	8	72.9
large	Exp	False	16	66.9
large	Exp	False	32	77.4
large	Exp	False	64	83.0
large	Exp	False	128	83.0
large	Exp	False	256	74.0
large	Exp	True	8	104.1
large	Exp	True	16	128.7
large	Exp	True	32	116.2
large	Exp	True	64	97.1
large	Exp	True	128	143.5
large	Exp	True	256	149.3

Table 4: Speed ablation over patch sizes for medium-sized Performer-ReLU with 100×100 inputs.

patch sizes	1×1	2×2	4×4	5×5	10×10
number of params (M)	15.7	9.93	8.50	8.33	8.16
CPU-time per MPC-iteration (ms)	11.3	2.6	2.3	1.5	0.5

For the 2000 expert trajectories of 100 horizon, we apply a moving window of size 20 and extract 80 trajectories of 20 horizon, but keep the original goal of the 100-horizon trajectory on the other side of the wall, as expert demonstration data for the Performer-MPC to learn (the starting locations of these expert trajectories are shown in Fig. 7, with a few rare failure cases of the offline planner displayed by a few erratic points).

C.3 Data Collection for Highly Constrained Maneuvers

To learn agile maneuvers in the highly constrained obstacle course, we collect data from a human expert via joystick teleoperation. The human expert teleoperates the robot to randomly navigate in a collision-free manner in the cluttered obstacle course (Fig. 3 b). The entire demonstration lasts around 30 minutes. For each data point, we set the goal as the 200-th future state on the demonstrated trajectory. At around 60Hz state rate and 0.5m/s driving speed, the goal is roughly 2m in front of the robot. In addition to this demonstration of desirable navigation behavior, we further collect about 10 minutes of demonstration starting from failure locations, e.g., where the robot is stuck and unable to recover from such situations, to address the distribution shift problem.

C.4 Data Collection for the Two Social Scenarios

We defer the data collection details for the two social scenarios to Appendix D after the social scenarios are formally defined to facilitate understanding.

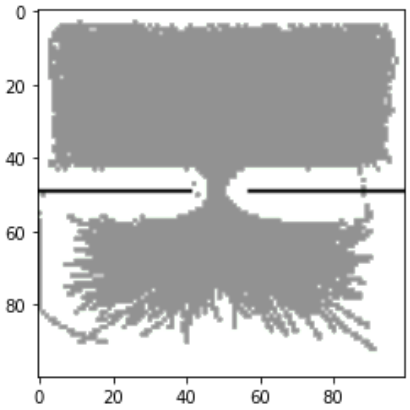

Figure 7: Expert Demonstrations Generated by an Off-Line MPC Planner Guided by a Dijkstra's Global Planner (x and y axis units are in occupancy grid cells).

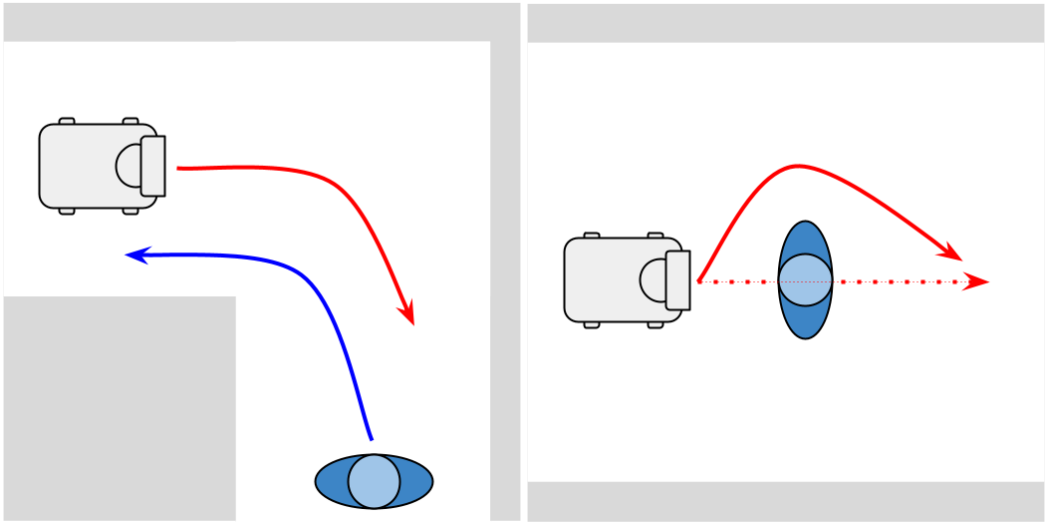

Figure 8: Social Navigation Scenarios. (a) In the `blind corner` scenario, robots should swing wide or slow down to avoid possible collisions with approaching humans who are not visible. (b) In the `pedestrian obstruction` scenario, the robot should go around the visible obstructing human with a comfortable passing distance.

C.5 Training Time

Our bilevel optimization takes around 0.36 seconds per iteration, i.e., a forward inference pass and a backward training pass (Fig. 2) to update the learnable parameters θ (Eqn. 2) on four TPUs. As shown in Fig. 4, it takes around 15, 60, 1.5, and 10 hours for our Performer-MPC model to converge for the doorway traversal, highly constrained maneuvers, blind corner, and pedestrian obstruction scenarios, respectively.

D Social Navigation Evaluation

To make evaluation of social navigation policies well-defined, realistic, scalable and repeatable, we use a social navigation benchmark based on a set of human-robot interaction scenarios to be evaluated using user surveys previously designed by Pirk et al. [95]. This benchmark consists of scenarios with well-defined roles and expected behavior for both humans and robots, along with a series of questions for human raters with answers defined on a five-level Likert scale [96]. To provide a concise metric for comparing policies, we average the answers to these questions into a

Blind Corner		Pedestrian Obstruction		General Questions	
BC1	The robot moved to avoid me.	PO1	The robot moved to avoid me.	G1	The robot adhered to social norms.
BC2	The robot stopped to let me pass.	PO2	The robot maintained a safe and comfortable distance.	G2	I would feel comfortable around the robot in this encounter.
BC3*	The robot nearly collided with me.	PO3*	The robot nearly collided with me.	G3	All things considered, I rate the robot motion as: `Very Poor` / `Poor` / `Neutral` / `Good` / `Very Good`.
BC4*	I had to move around the robot	PO4	It was clear what the robot wanted to do.		

Table 5: Questions for Each Social Scenario. A star indicates a question is negative, so that a 'Strongly Disagree' result should be aligned with 'Strongly Agree' for a positive question.

social navigation score reported in the main body of the paper (Fig. 6). This section justifies that choice by describing this social navigation protocol and the results of our experiments in more detail.

D.1 Social Navigation Scenarios

Crucially for our purposes, this protocol can be used to generate trajectories and evaluate whether they meet the scenario's social criteria, enabling us to create curated datasets of expert trajectories which score highly on our benchmark. For training Performer-MPCs, we chose two scenarios:

- `blind corner` (Fig. 8a), in which a robot is expected to apply some strategy (such as slowing down or swinging wide) to reduce the likelihood of collision with a possible unseen pedestrian coming around a corner.
- `pedestrian obstruction` (Fig. 8b), in which the robot is expected to drive around a human obstructing its path, while respecting the human's comfort distance.

Table 5 details the questions used to evaluate each scenario, along with a set of scenario-independent general questions. For most questions, the Likert scale is implemented as `Strongly Disagree, Disagree, Neutral, Agree, Strongly Agree`, except for question G3, which evaluates overall performance with `Very Poor, Poor, Neutral, Good, Very Good`.

D.2 Collecting Expert Demonstrations

We use the scenario definitions to collect a variety of expert demonstration datasets. For both scenarios, we collect around 30 episodes each scenario, which prove sufficient for training Performer-MPC. Note compare to taking visual RGB camera input which requires over 700 episodes each scenario [95], Performer-MPC takes occupancy grids as input and requires substantially smaller amount of data.

For our social scenario data collection, human expert trajectories are collected by individuals trained as both participants and evaluators of the social scenarios, who attempt in their runs to guide the robot according to the social norms defined in the scenarios. In turn, our evaluation of Performer-MPC against baselines in these scenarios uses the same definitions and questionnaires which guide the trajectories; thus our experimental results gauge how well Performer-MPCs can successfully navigate with respect to social norms whose cost functions are difficult to explicitly design.

For `pedestrian obstruction`, we discover that both `Performer-MPC` and `EP` tend to memorize the building configurations of the training environment (e.g., walls, chairs, and tables) and sometimes do not respond to the human properly during deployment. Therefore, we augment the existing training data by randomly shuffling the background (i.e., randomly removing or adding obstacle pixels to the surrounding area, but keeping the space around the robot-human interaction point intact). We posit that such data augmentation may not be necessary when we scale up our data collection to different building configurations, and more importantly, adding extra information to distinguish humans from obstacles (e.g., human detection, tracking, and prediction). For the `blind corner` scenario, swinging wide to avoid the inner side of the corner is part of the desired learning process, therefore such augmentation is not necessary.

We also randomly select goal locations between behind the human and the final robot position of each episode for the `pedestrian obstruction` scenario to improve the model's robustness against

different goals. Again, for `blind corner`, we find such augmentation not necessary and simply select the 300-th future state on the demonstrated trajectory as goal for each data point. In this way, most data points have a goal behind the corner. We posit that these two data augmentation techniques contribute to the much longer training time for `pedestrian obstruction` than that for `blind corner` (Fig. 4).

D.3 Human Evaluation Pilot Study

To evaluate the performance of our social navigation policies, we conduct a pilot study, gathering both `participant` and `observer` perspectives using the scenario questionnaires. Due to covid-19 restrictions and limited availability of participants for in-person user study participation (N=9), we ran this pilot study with members of our research team. In this pilot study, we aim to address the research question: *How well does our* `Performer-MPC` *social navigation policy perform on our social navigation metrics when compared against* `RMPC` *and* `EP`?

Beyond that research question, we also aim to explore several other variables. First, we want to see how these policies would perform when comparing previously seen environments (i.e., environments where the training set is collected) vs. unseen environments (i.e., novel environments not in the training set). Second, we want to examine how direct interactants (1st person) vs. bystanders (3rd person) would rate the robot's social navigation performance because we hypothesize that 1st person responses might be stronger, especially when it comes to perceptions of comfort and safety. Third, we want to test the policies' performances across at least two different social navigation scenarios, starting with `blind corner` and `pedestrian obstruction`.

As this is a large set of variables (navigation policy, performing in seen vs. unseen environments, measuring from 1st vs. 3rd person perspectives, and navigating in two different social navigation scenarios) and we have a very limited set of research study participants, we opt to run this as an exploratory pilot study, not as a fully controlled, counterbalanced, human subjects experiment. Altogether we run 120 sessions, gathering 240 sets of questionnaire responses from 1st and 3rd person perspectives in the course of one day in June 2022 on our campus (N=9). To minimize possible bias, neither pedestrians nor observers are aware of what policy is being tested during each episode, and the ordering of policies is randomized.

D.4 Evaluating Social Navigation Performance Factors

To assess the relationship between people's responses to our social navigation questionnaire items, we run principal component analyses (using varimax rotations) to see if the variables really do hang together. We also run reliability analyses to see if those items that load onto a single factor are indeed reliable measures of an underlying factor.

- `blind corner`: The three general questions in this set (Questions G1-G3) create a highly reliable factor, Cronbach's alpha = .92 (N=120). When we run PCA on the questions that are specific to the `blind corner` scenario (Questions BC1-B4), one of the items does not correlate strongly with the other three (BC2: The robot stopped to let me pass).
- `pedestrian obstruction`: We find that the three general social navigation performance questions (Questions G1-G3) create a highly reliable factor, Cronbach's alpha = .99 (N=240). For the four questions specific to the `pedestrian obstruction`, we find that all four questions (Questions PO1-PO4) also create a highly reliable factor, Cronbach's alpha = .97 (N=120).

Because these results show most these variables are highly correlated, we average them into a "social navigation score" for presenting our results in the main body of the paper concisely. The following section presents the detailed results of the social questionnaire evaluation.

D.5 Pilot Study Results

As we are unable to fully balance the experiment design (e.g., getting each of our participants to try out each of the 24 experiment conditions once), we cannot satisfy the statistical analysis assumptions of repeated measures ANOVAs. As such we are reporting upon the descriptive statistics (means and standard errors) of our pilot study data, but we recommend interpreting these results as pilot study findings, not statistically significant findings that indicate causal relationships.

Policy Tested	Eval Cond	BC1 Avoided Human	BC2 Stop for Human	BC3 No Fear Collision	BC4 No Dodge	G1 Social Norms	G2 User Comfort	G3 Perceived Quality	Social Score	Failed Episode	Num Samples
Regular MPC	Seen	1.65	2.00	1.75	1.95	1.90	2.00	1.90	1.88	0%	60
	Std. Err.	*0.08*	*0.11*	*0.14*	*0.15*	*0.12*	*0.13*	*0.12*			
	Unseen	2.10	2.15	2.50	2.05	2.60	2.60	2.65	2.38	0%	60
	Std. Err.	*0.09*	*0.09*	*0.11*	*0.14*	*0.11*	*0.11*	*0.10*			
Explicit Policy	Seen	3.90	1.90	4.65	4.70	4.65	4.70	4.60	4.16	0%	60
	Std. Err.	*0.14*	*0.14*	*0.06*	*0.06*	*0.06*	*0.06*	*0.06*			
	Unseen	3.40	1.80	4.00	3.70	2.95	3.25	3.10	3.17	20%	60
	Std. Err.	*0.18*	*0.14*	*0.17*	*0.20*	*0.17*	*0.18*	*0.18*			
Performer-MPC	Seen	3.20	1.75	4.60	4.35	4.30	4.45	4.50	3.88	0%	60
	Std. Err.	*0.17*	*0.07*	*0.10*	*0.10*	*0.08*	*0.09*	*0.09*			
	Unseen	3.75	1.90	4.60	4.20	4.15	4.25	4.25	3.87	0%	60
	Std. Err.	*0.13*	*0.08*	*0.06*	*0.09*	*0.10*	*0.09*	*0.08*			

Table 6: Social Navigation Questionnaire Results for `blind corner`.

Policy Tested	Eval Cond	PO1 Avoided Human	PO2 Comfort Distance	PO3 No Fear Collision	PO4 Motion Legible	G1 Social Norms	G2 User Comfort	G3 Perceived Quality	Social Score	Failed Episode	Num Samples
Regular MPC	Seen	1.74	1.42	1.47	1.63	1.47	1.53	1.58	1.55	0%	60
	Std. Err.	*0.13*	*0.07*	*0.08*	*0.11*	*0.07*	*0.07*	*0.07*			
	Unseen	1.95	1.85	2.00	2.00	2.00	2.00	1.95	1.96	0%	60
	Std. Err.	*0.21*	*0.20*	*0.19*	*0.17*	*0.19*	*0.19*	*0.19*			
Explicit Policy	Seen	4.45	4.15	4.40	3.80	4.15	4.30	4.15	4.20	5%	60
	Std. Err.	*0.08*	*0.12*	*0.10*	*0.12*	*0.10*	*0.10*	*0.10*			
	Unseen	4.30	4.15	4.05	3.80	4.10	4.10	4.05	4.08	5%	60
	Std. Err.	*0.16*	*0.15*	*0.15*	*0.17*	*0.15*	*0.15*	*0.15*			
Performer-MPC	Seen	4.57	4.38	4.48	4.19	4.29	4.38	4.43	4.39	0%	60
	Std. Err.	*0.07*	*0.10*	*0.10*	*0.07*	*0.12*	*0.08*	*0.08*			
	Unseen	4.00	4.15	4.15	4.05	4.15	4.15	4.05	4.10	0%	60
	Std. Err.	*0.19*	*0.16*	*0.16*	*0.15*	*0.16*	*0.16*	*0.15*			

Table 7: Social Navigation Questionnaire Results for `pedestrian obstruction`.

- `blind corner` (Tab. 6): The best performing policy on `blind corner` is the `EP` policy in the `seen` condition, with a combined score of 4.16 and individual questionnaire scores equal to or higher than both other policies. However, `EP` does not generalize well to the `unseen` condition, suffering a 20% failure rate and a drop in social score to 3.17; differences between the performance of these policies on individual questions are generally greater than the standard errors of these policies, potentially indicating a real difference that could be teased out with a larger study. In contrast, `Performer-MPC` generalizes well, with scores on `seen` and `unseen` of 3.88 and 3.87 respectively, with individual questions generally showing greater differences. `RMPC` is the worst performing policy in the overall social score and on most individual questions, except BC2, "The robot stopped to let me pass," on which it is slightly superior due to stopping for users. However, this illuminated issues on question BC2, which we discuss further below.
- `pedestrian obstruction` (Tab. 7): The best performing policy on `pedestrian obstruction` is `Performer-MPC` in the `seen` condition with overall social score of 4.39, with a relatively small drop to 4.10 in the `unseen` condition. `EP` also performs well with scores in `seen` and `unseen` of 4.20 and 4.08, respectively, though it fails to complete the task 5% of the time in both conditions. While the difference in `seen` performance of these policies is typically greater than the standard error of their performance, potentially indicating a real difference which could be teased out with a larger study, this is not true in the `unseen` case. `RMPC` is the worst performing policy in both social score and individual questions. Results within questions, between questions and within policies under given conditions are generally more consistent for `pedestrian obstruction` than `blind corner`.

Overall, we interpret these results to indicate that `Performer-MPC` has better generalization than `EP` and is comparable in social navigation performance to `EP`, and that both of those policies are superior to `RMPC` at social navigation. The social navigation score results presented in the main body of the paper are consistent with this detailed analysis.

However, the outlier question BC2 is worth further discussion. All other questions in our survey hang together to create highly reliable factors, but BC2 does not, and show lower scores for both `Performer-MPC` and `EP`, which otherwise score highly on the social navigation questionnaire. Discussion with the participants and analysis of robot behaviors in the episodes reveal that this question inadvertently prescribes a solution: that the robot should stop at the blind corner. However, our expert training demonstrations incorporate a different solution: swinging wide at the corner to avoid collisions, which our human robot drivers determine is the preferred solution based on the navigation speed and stopping distance of the robot. In contrast, `RMPC`, while not scoring well on most social questions, nevertheless stop for the user, giving it an artificially high score on this question even though its behavior is not very social due to stopping very close to the user. While we report the results of this question for completeness, for future work we plan to craft questions which evaluate social navigation without prescribing a solution.

D.6 Limitations and Future Work

This paper focuses on whether Performer-MPC can successfully learn from expert demonstrations derived from real-world scenarios and then be successfully deployed in those scenarios on-robot. Therefore, we select a limited set of social scenarios which enables us to evaluate this research question. These scenarios are necessarily limited to those which can be detected from the occupancy grid, which preclude the use of the visual gesture-based scenarios proposed by Pirk et al. [95]. Furthermore, data for these scenarios are collected by a limited number of human experts, and the policies for each scenario are trained separately. In future work, we plan to expand to a wider range of scenarios collected by a broader range of experts, and to train policies to solve sets of scenarios rather than a single scenario.

The user evaluation pilot study is a first step toward developing a more robust user study protocol for evaluating future versions of social navigation policies. Our pilot study is limited by its use of research team members as study participants; their perspectives on social navigation behavior are quite influenced by their experience with operating and running tests on these robots in their work. In the future, we will recruit user study participants from people, who are not part of our research team. Second, our pilot study is not properly balanced so we cannot run the usual statistical analyses necessary to evaluate the statistical significance of the effects we observed. Instead, we report upon the descriptive statistics for this paper and we will run a full user study as a next step in this research project. In future user studies, we will focus upon more targeted research questions so that the experiments are simpler to run, analyze, and interpret, and will ensure these questions focus on quality of social navigation without tying evaluation quality to mimicking a specific solution behavior.