Exploiting Proximity-Aware Tasks for Embodied Social Navigation

Enrico Cancelli[1*]   Tommaso Campari[2*]   Luciano Serafini[2]   Angel X. Chang[3,4,5]   Lamberto Ballan[1]

[1] University of Padova   [2] Fondazione Bruno Kessler (FBK)   [3] Simon Fraser University
[4] Canada-CIFAR AI Chair   [5] Amii

Abstract

Learning how to navigate among humans in an occluded and spatially constrained indoor environment, is a key ability required to embodied agents to be integrated into our society. In this paper, we propose an end-to-end architecture that exploits Proximity-Aware Tasks (referred as to Risk and Proximity Compass) to inject into a reinforcement learning navigation policy the ability to infer common-sense social behaviours. To this end, our tasks exploit the notion of immediate and future dangers of collision. Furthermore, we propose an evaluation protocol specifically designed for the Social Navigation Task in simulated environments. This is done to capture fine-grained features and characteristics of the policy by analyzing the minimal unit of human-robot spatial interaction, called Encounter. We validate our approach on Gibson4+ and Habitat-Matterport3D datasets.

1. Introduction

Navigating safely in a dynamic scenario populated by humans who are moving in the same environment is necessary for embodied agents such as home assistants robots. To do so, as depicted in Figure 1, the agent should be able to dynamically and interactively navigate the environment by avoiding static objects and moving persons.

Recently, the development of photorealistic 3D simulators [35, 36, 21] has provided the tools to train embodied agents and experiment in large-scale indoor environments [8, 31, 15]. Thanks to these frameworks, several tasks and challenges have been introduced [1, 48, 14], fostering the development of accompanying techniques to solve these tasks. In particular, in the PointGoal Navigation task (where an agent is required to reach a specific location in an environment), an agent without any sensor/actuation noise trained for billions of steps can obtain almost perfect performance [42]. Other approaches [45, 27] obtained impressive
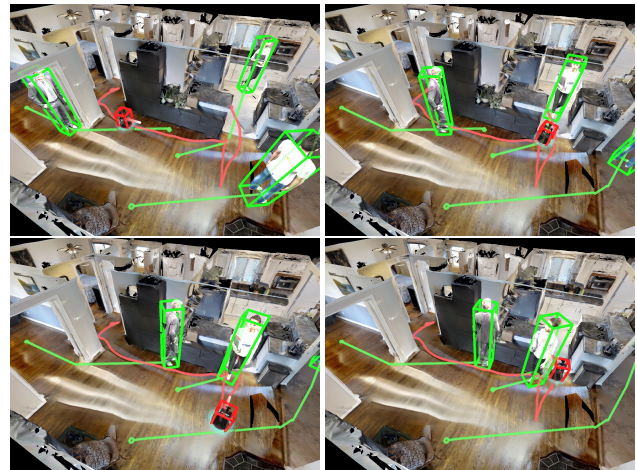

Figure 1: Illustration of an agent-person "encounter". From top-left to bottom-right: *i*) episode starts; *ii*) the embodied agent/robot sees a person; *iii*) it moves back to avoid a collision; *iv*) it reaches the goal by avoiding the person.

results even in the presence of noise. Another relevant task is Object Goal Navigation, where an agent is required to find and navigate to a specific object given the object category. This task requires both semantic and navigation capabilities and leads to the development of modular approaches based on semantic maps [9, 6, 32], as well as end-to-end trained models using reinforcement learning (RL) [46, 33].

However, despite encouraging progress on the challenging task, all the previously mentioned tasks frame navigation in a fundamentally static environment. The dynamic element introduced by sentient, moving human beings in the scene forces us to rethink how the current models are designed. A good navigation policy must not be just effective (i.e., able to achieve its goal) and efficient (i.e., able to achieve the objective through a close-to-optimal path) but also safe (reaching the destination without harming others). This social element is included in the Social Navigation Task (SocialNav) [44, 29], where an agent must tackle PointGoal Navigation in simulated indoor environments. To tackle this task, Yokoyama et al. [47] introduced a simple

---

but quite effective model that placed first in the iGibson 2021 Interactive challenge. However, the approach does not explicitly encode any social behaviour in its navigation policy. We believe that a clear encoding of human-agent interactions, as well as social behaviours, are required for safe navigation and interaction with humans. By modelling the movement of humans, the agent could prevent collisions or dangerous behaviours and adapt its path to the dynamic environment in which it is navigating. We encode these "signals" by introducing two *Proximity-Aware Tasks*, referred as *risk* and *proximity compass*. These auxiliary tasks model the present and future danger for the agent's action.

Additionally, we define a fine-grained evaluation protocol for the SocialNav task to better analyse the performances of agents during human-agent interactions. Our evaluation protocol is inspired by a similar attempt [30] in robotics, which consisted of collecting statistics about specific types of interaction between humans and a robot (through questionnaires). We propose an automated evaluation by identifying and characterizing *encounters* between human and agent. To do so, we extract short sub-sequences where an interaction with a human becomes a predominant factor influencing navigation, and we establish a set of rules for classifying each encounter based on the type of human-agent spatial relationship through time. Finally, we also introduce a dataset of episodes on top of HM3D [31] for Embodied Social Navigation to assess our agents in different environments.

In summary, the contributions of this work are threefold: (1) A novel architecture for embodied social navigation which is based on Proximity-Aware tasks; we show the effectiveness of the model on two public datasets. (2) A new encounter-based evaluation protocol for analysing social navigation models. (3) A set of episodes for evaluating embodied social navigation based on the HM3D dataset (called HM3D-S).

2. Related Work

Embodied Navigation. In the last few years, there has been growing interest in studying indoor navigation in an embodied setting [14]. This was mainly possible thanks to large-scale datasets consisting of 3D indoor environments [8, 36, 31], and to simulators that allow researchers to simulate navigation inside these 3D environments [35, 36, 21]. To study embodied AI, many tasks [1] were proposed including: PointGoal Navigation [42], ObjectGoal Navigation [4], Embodied Question Answering [41], and Vision and Language Navigation (VLN) [2, 22]. To tackle these problems, where an agent operates in static, single-agent environments, modular approaches were proposed [7, 10, 9, 32], exploiting SLAM, Path Planning and exploration strategies, and end-to-end RL-trained policies [40, 27, 45, 6, 46], without exploiting any explicit map.

In this paper, we focus on the Embodied Social Navigation Task [29], where the agent has to navigate in environments with moving humans. This task was introduced in the iGibson [44] SocialNav challenge. SocialNav adds new challenges to visual navigation since modelling of human movements is required to avoid collisions. Modular approaches relying on static maps are harder to adapt in this context since humans are constantly moving and therefore harder to track.

Socially-Aware Navigation. Socially (or human) aware representations and models have been studied by researchers in robotics, computer vision and human social behaviour analysis [26]. Work on collision-free multi-agent navigation [5, 39, 13, 23] and navigation in dynamic environments [3] has been extended to navigation in the presence of humans [19, 16, 12, 24, 11]. Chen et al. [12] employs collision avoidance algorithms like CADRL [13] and introduces common-sense social rules to reduce uncertainty while minimizing the risk of incurring in the Freezing Robot Problem [38]. Other works [16, 11] attempt to model human-agent interaction using techniques like spatio-temporal graph [24]. These methods have been typically tested on minimalist simulation environments that provide complete knowledge with simple obstacles and often assume collaboration between moving agents. In contrast, we focus on SocialNav in large indoor environments, where the environment is unknown during test time.

Egocentric Visual SocialNav. Recently, an increasing number of work has been investigating SocialNav given egocentric visual data. Large-scale egocentric datasets of socially compliant demonstrations [20] and trajectories of agents navigating in social environments [34, 25] have enabled the training of social navigation agents via imitation learning. Despite the usefulness of such large datasets, it is still useful to train agents in simulation, so that it is possible to observe corner cases that are not easily collectable in the real world. Using simulation, agents are typically trained using either imitation learning [37], end-to-end reinforcement learning [47], or a combination of a planner with RL [29].

Tai et al. [37] used depth maps to train an agent using imitation learning and studied the policy behaviour in a set of simulated interactions. Pérez-D'Arpino et al. [29] focused on constrained indoor environments and uses a combination of a global planner with complete map access and a low-level RL policy that uses LIDAR data. However, this approach requires prior knowledge about the environment for the path planner to work. In our work, we focus on exploring methods applicable to unknown environments without having to build a map. Yokoyama et al. [47] proposed an end-to-end RL-trained policy for Social Navigation. This model extracts embeddings from the Depth and the GPS+Compass sensors and feeds them to a GRU,

together with the previous action. However, this approach does not model social properties like proximity. Instead, our model learns and uses these concepts with two Proximity-Aware tasks, that help the agent during the episodes.

Evaluation of SocialNav. Prior works in SocialNav [17] are usually evaluated using overall success rate, path efficiency, collision avoidance, or similarity to human paths. The success rate is limited since the agent is dealing with humans and it is preferable to navigate safely in order to avoid collisions, even if this means having a lower success rate. In the iGibson challenge, SocialNav is also evaluated using STL (success weighted by time length) and PSC (personal space compliance), the percentage of timesteps that the agent is more than a threshold distance (0.5m) from pedestrians. In contrast to these metrics, we propose a fine-grained evaluation based on the types of *encounters* the agent has with pedestrians.

While there have been attempts at fine-grained evaluation for Social Navigation, there is no standard automatic evaluation based on agent-pedestrian encounters. Most notably, Pirk et al. [30] defined an evaluation protocol for social agents based on human questionnaires. This cannot be easily applied to simulations, where test sets contain thousands of episodes. In this paper, we propose an *automatic* evaluation protocol to measure the social capabilities of our models (see Section 5).

3. Background: the SocialNav Task

In Embodied Social Navigation [44, 29, 47], as in Point-Goal Navigation, the agent aims to reach a target location, but a collision with a human subject constitutes a failure and will terminate the episode. An episode e is characterized by the agent trajectory $\boldsymbol{\alpha}$, the tuple of human trajectories $(\mathbf{p}^i)$, and the target goal g. The agent trajectory $\boldsymbol{\alpha}$ is a sequence of positions and rotations of the agent from the beginning to the end of the episode t_{end}. Formally, $\boldsymbol{\alpha} = \{\alpha_t\}_{t \in [0, t_{\text{end}}]}$ where $\alpha_t \in SE(2)^1$ is the 2D translation and rotation of the agent with respect to the origin at time t. Similarly, the trajectories of humans in the same episode are sequences of positions and rotations associated with the i-th human. Formally, $\mathbf{p}^i = \{p_t^i\}_{t \in [0, t_{\text{end}}]} \ \forall i \in \mathcal{P}$ with $p_t^i \in SE(2)$. In our simulation, the movement of each person is constrained by a start point and an endpoint, with the person moving back and forth between those two points along the shortest path.

The target goal $g \in \mathcal{G}$ is specified by the 2D position in world coordinates. The agent must at any point in time provide an action (lin_vel, ang_vel) $\in [-1, +1]^2$, representing the normalized linear forward velocity and the normalized clockwise rotational angular velocity (where +1 is the max velocity and -1 the max backward/counter-clockwise velocity). The stop action is called when the agent is within 0.2

---

meters from the target goal point. The agent has 500 actions (or steps) to reach the target location. If it collides with a human, the episode terminates immediately.

4. Method

Overview. Figure 2 shows an outline of our framework. It comprises two main modules: (i) *Proximity feature extraction*, and (ii) *Policy architecture*. The *Proximity feature extraction* module refines proximity information obtained from the simulator to extract features that describe some aspect of social interactions (ground truth proximity features). The *Policy architecture* extracts from the RGB-D and the GPS+Compass sensors an embedding that serves as input for our Proximity-Aware tasks. These tasks refine this embedding and create *n* task embeddings (one per task) which are then fused together through state attention. An action is sampled from the state attention output.

4.1. Policy Architecture

Our policy network comprises the following modules: *i)* two encoders (the *Visual backbone* and the *Position Encoder*) that create an embedding from the RGB-D and the GPS+Compass sensors; *ii)* a *Recurrent State Encoder* that accumulates such embedding through a series of recurrent units; *iii)* a *State Attention* module that fuses the outputs of such units through an attention mechanism to produce the action the robot has to perform.

We encode each RGB-D frame x_t using a CNN (*Visual Backbone*) $f(\cdot)$ to a visual embedding $\phi_t^v = f(x_t)$. The position and rotation of the agent α_t are encoded using a linear layer $g(\cdot)$ to obtain the embedding $\phi_t^p = g(\alpha_t)$. The outputs of these two encoders are then concatenated into the final embedding $\phi_t^f = \phi_t^v \oplus \phi_t^p$. To accumulate embeddings over time, we follow Ye et al. [45]'s design for PointNav and implement our state encoder as a stack of parallel recurrent units. Each unit at each timestep is fed ϕ_t^f, and outputs its internal state, called *belief*.

The key idea of having multiple beliefs is that each recurrent unit can focus on a specific navigation aspect. The final decision about what action the robot should take is sampled by weighting each belief according to the situation. For this reason, all beliefs $\mathcal{B}$ are subsequently fused through the *State Attention* module to compute the mean $\vec{\mu}_t$ and standard deviation $\vec{\sigma}_t$ of the normal distribution from which we sample the action a_t. Formally, given $\{RU^{(i)}\}_{\forall i \in \mathcal{B}}$ a set of recurrent units, the encoded beliefs h_t are defined as:

$$h_t := \{h_t^{(i)}\}_{\forall i \in \mathcal{B}} \leftarrow \{RU^{(i)}(h_{t-1}^{(i)}; \phi_t^f)\}_{\forall i \in \mathcal{B}} \quad (1)$$

The fusion mechanism of the state attention module SA is:

$$\vec{\mu}_t, \vec{\sigma}_t \leftarrow SA(h_t, \phi_t^f) = FC_a(\text{Attention}(h_t, FC_k(\phi_t^f), h_t))$$
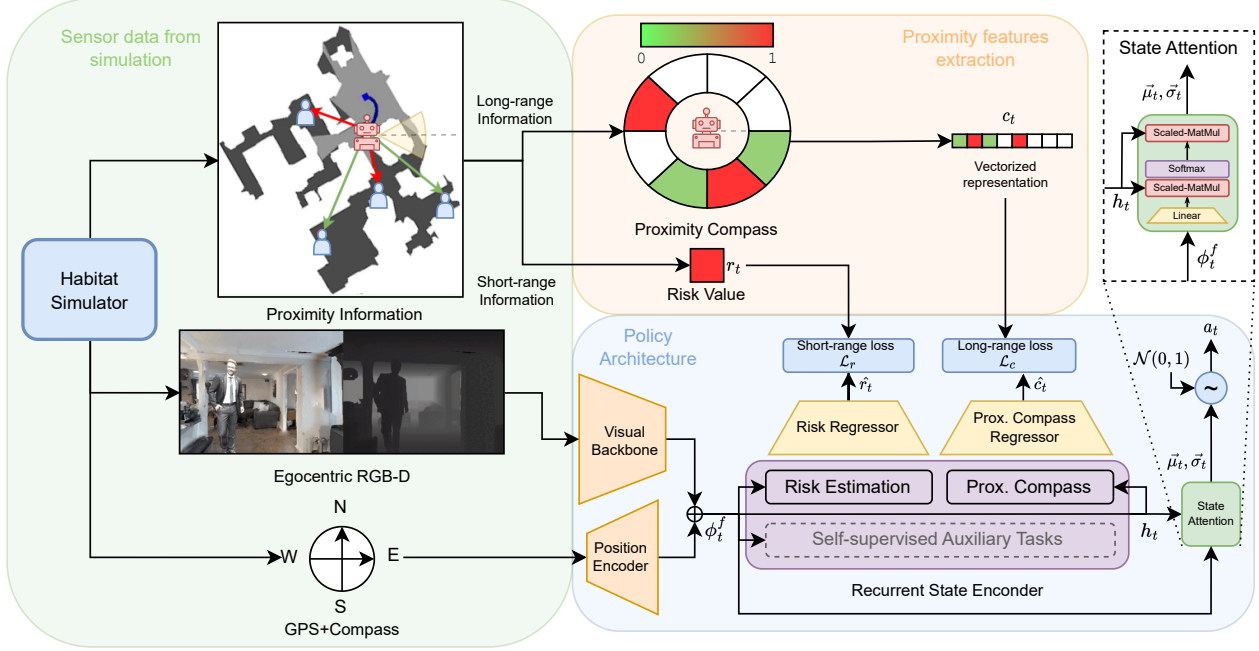

Figure 2: Pipeline and model overview. *Proximity information* is extracted from Habitat Simulator (left rectangle) and is processed through a *Proximity Feature extraction* procedure (top-right). The policy (bottom-right) uses RGB-D and GPS+Compass data as input and, during training, is conditioned by the extracted proximity features.

where $\text{Attention}(Q, K, V) \mapsto \text{Softmax}(\frac{QK^T}{\sqrt{d_k}})V$ and FC_a and FC_k are two linear layers.

4.2. Proximity-Aware Tasks

With multiple beliefs, we can inject different signals in our embeddings, e.g., social dynamics occurring in an episode. To do so, during training, we condition each belief with a unique auxiliary loss jointly optimized with the action and value ones during the optimization step of the policy network. This is done by processing each belief with a specific type of *Proximity feature*, through a *Regressor network* (see Fig. 3), that computes our *Proximity-Aware tasks* predictions. Each auxiliary task is responsible for predicting the proximity features in the time range $[t, t+k]$, conditioned by the corresponding belief $h_t^{(i)}$ and the sequence of performed actions $\{a_j\}_{j\in[t,t+k]}$, where k is the number of future frames to predict. Formally, for a given sequence of proximity features $\{s_j\}_{j\in[t,t+k]}$, the task aims to optimize the following auxiliary loss:

$$\mathcal{L}_f = \frac{\sum_{j\in[t,t+k]}\text{MSE}(s_j, \hat{s_j})}{k} \quad (2)$$

where $\{\hat{s_j}\}_{j\in[t,t+k]} = \mathcal{M}(h_t^{(i)}, \{a_j\}_{j\in[t,t+k]})$ and $\mathcal{M}$ is the regressor network. The proximity features are only fed to the model at training time and regressor networks are detached during evaluation.

We design two types of proximity tasks corresponding to two social features: (i) *Risk Estimation*, and (ii) *Proximity Compass*. Our design has the benefit of being easily extensible with other, possibly more complex social tasks and to be also compatible with general purpose self-supervised tasks like the ones used in Ye et al. [45] (e.g., CPC|A [18] or ID [28, 46]).

To exploit different proximity features, we extract from the simulator the relative position of every person w.r.t. the agent. We refer to this data as *Proximity Information*:

$$SI_t \stackrel{\text{def}}{=\!=} \{\delta_t^i := (\text{pos}(p_t^i) - \text{pos}(\alpha_t)) \in \mathbb{R}^2\}_{\forall i \in \mathcal{P}}$$

where the function $\text{pos}(\cdot)$ extracts the position from an element of $\boldsymbol{\alpha}$ or $\mathbf{p^i}$.

Risk Estimation. *Risk Estimation* is a Proximity-Aware Task designed to deal with short-range social interactions, to inform the agent about imminent collision dangers. Given SI_t, we define the *Risk value* as a scalar representing how close the agent and the nearest person are up to a maximum distance D_r. This value ranges from 0 (the nearest neighbor is further than D_r meters away) to 1 (the agent and person are colliding). Formally:

$$\text{risk}_t = \text{clamp}\left(1 - \frac{\min\{||\delta_t^i||_2 \mid \delta_t^i \in SI_t\}}{D_r}, 0, 1\right) \quad (3)$$

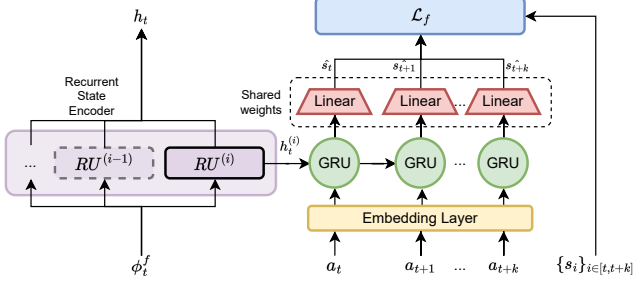where $\text{clamp}(\cdot, 0, 1)$ limits the value to the $[0, 1]$ range.

Figure 3: Regressor network. Actions $a_t...a_{t+k}$ are used as input, and a linear layer processes the GRU's hidden states to obtain the predicted proximity features $\hat{s}_t...\hat{s}_{t+k}$; $\{s_i\}_{i \in [t,t+k]}$ is the ground truth used by $\mathcal{L}_f$ (from Eq. 2).

Proximity Compass. The *Proximity-Aware Task* models the long-distance component of social dynamics. This feature captures not only social interaction on a larger area with radius $D_c > D_r$ but also a weak indication of the direction a person may come. Much like humans can make guesses about people's whereabouts based on previous observations, partial knowledge of the environment, and a person's trajectory; we expect our agent to acquire similar knowledge at training time.

Such information is represented through a *Proximity Compass*. In the compass, north represents the direction the agent is looking, and the quadrant is partitioned into a finite number of non-overlapping sectors. Given each person $i \in \mathcal{P}$, $\theta_{a \to i}$ represents the angle of the segment connecting the agent to that person w.r.t. the north of the compass. These angles are associated with a specific sector. We compute the risk value for each sector among people in the same quadrant. The entire compass is represented as a vector by unrolling the sequence of sectors from the north going clockwise. Formally if we have k number of equivalent sectors, the vector $\text{comp}_t \in \mathbb{R}^k$ is defined as:

$$\text{comp}_t[j] = \left[\text{clamp}\left(1 - \frac{\min\{||\delta_t^i||_2 \mid \delta_t^i \in \Theta_j\}}{D_c}, 0, 1\right)\right]$$

$$\text{with } \Theta_j = \left\{C\delta_t^i \in SI_t \mid \theta_{a \to i} \in \left[\frac{2\pi}{k} \cdot j, \frac{2\pi}{k} \cdot (j+1)\right)\right\}$$

$$\forall j \to [0, k-1]$$

4.3. Implementation details

The visual backbone used in our experiments is a modified version of a ResNet-18 for RGB-D frames. We use the same implementation used in [47]; this produces an output embedding of size 512. For the agent's position encoder, we use a linear layer with an output size of 32. Our recurrent state encoder is implemented as a stack of single-layer GRUs with a hidden dimension of 512 each. Regressors for both proximity tasks are implemented as single-layer GRUs with hidden size 512. The sequence of actions $\{a_i\}_{i \in [t,t+k]}$

is passed through an embedding layer and then fed to the input of the GRU. The initial hidden state is initialized to $h_t^{(i)}$, and k is set to 4. See Figure 3 for the complete scheme.

Each model was trained on 3 GPUs Nvidia A5000 with 8 parallel environments for each machine using the DD-PPO [42] algorithm. Our reward function is the following:

$$r_t = -\Delta_d + r_{\text{slack}} - \beta_{c \wedge b} \cdot (I_{\text{coll}} + \cdot I_{\text{back}}) + \beta_{\text{succ}} \cdot I_{\text{succ}} \quad (4)$$

where Δ_d is the potential reward based on the geodesic distance to the goal, r_{slack} is a slack reward (a small penalty given for every step to avoid unnecessary actions or loops), and $I_{\text{coll}}, I_{\text{back}}, I_{\text{succ}}$ represent the indicator functions respectively of a collision with objects in the environment, the linear velocity being less than 0 and success. $\beta_{c \wedge b}$ and β_{succ} are coefficients. We used the same parameters as in [47], that are -0.002 for r_{slack}, 0.02 for $\beta_{c \wedge b}$ and 10.0 for β_{succ}.

5. An evaluation protocol for SocialNav

We introduce an evaluation protocol to analyze the performance of navigation agents on the SocialNav task. We group *encounters* between the agent and other pedestrians in the environments into four classes (*frontal*, *intersection*, *blind corner*, *person following*) and propose a set of metrics to analyze the different types of encounters.

Encounter. We formally define an *encounter* taking place in episode e between the agent and a specific pedestrian $i \in \mathcal{P}$ as a subsequence of trajectories $\boldsymbol{\alpha}$ and $\mathbf{p^i}$ in a given timeframe $[t_1, t_2] \subseteq [0, t_{\text{end}}]$ such that the following constraints are met:

- **Time Constraint**: the timeframe $[t_1, t_2]$ is larger than a threshold $T_{\min}$;
- **Spatial Constraint**: the geodesic distance between the agent and person i $\forall t \in [t_1, t_2]$ is less than a threshold $D_{\max}$;
- **Heading Constraint**: person i is in front of the agent for the first T_{front} timesteps. That is, given the agent's heading angle θ_t^a, $\theta_t^{a \to i}$ the angle of the segment connecting the agent to person i and a threshold $\Theta_{\max}$, $|\theta_t^a - \theta_t^{a \to i}| \leq \Theta_{\max}$ holds $\forall t \in [t_1, t_1 + T_{\text{front}}]$.

Inclusion rule. To characterize encounters between the agent and a human, we devise a heuristic called *inclusion rule (IR)*. Each IR is defined by the following parameters:

- Δ_t^i, Δ_t^a represent, respectively, an approximation of the general direction of the trajectory of the agent and a person i in the timeframe $[t_1, t]$ with $t \in [t_1, t_2]$, where t_1, t_2 are the start and the end timesteps of an encounter;
- intersect is a binary value that is 1 if robot and pedestrian paths intersect and 0 otherwise;
- $\text{blind}(t)$ is a time-conditioned binary value indicating whether the agent can see the person at time step t;

(a) Frontal approach   (b) Intersection

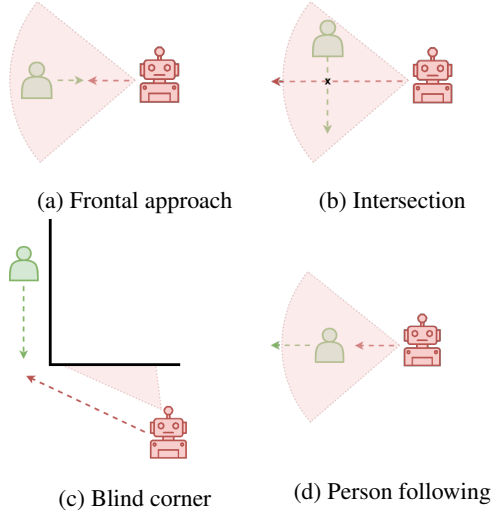(c) Blind corner   (d) Person following

Figure 4: The four different classes of encounter. The dashed line represents the general direction of the agent and the person involved. The red area represents the agent's field of view at the beginning of the encounter.

- d_diff(t): difference between the geodesic and the euclidean distance between the agent and the person.

Encounter classification. We define four categories of encounters (inspired by Pirk et al. [30]), and their respective *inclusion rule* (see Figure 4):

- **Frontal approach**: The robot and the human come from opposite directions and have trajectories that are roughly parallel. Thus the agent shall deviate slightly to avoid a frontal collision. IR: $(\neg \, \mathrm{blind}(t) \ \forall t \in [t_1, t_1 + T_{\mathrm{view}}]) \wedge \pi - \Delta_{\mathrm{slack}} \leq |\Delta_{t_2}^i - \Delta_{t_2}^a| \leq \pi + \Delta_{\mathrm{slack}}$, where Δ_{slack} is a slack value (in radians) on the angle π and T_{view} is the number of initial timesteps in which the person must be visible by the agent.
- **Intersection**: The robot and the human's trajectory intersect at approximately $90°$. In this situation, an agent may want to stop and yield to the human or decrease its linear velocity and slightly deviate. IR: $(\neg \, \mathrm{blind}(t) \ \forall t \in [t_1, t_1 + T_{\mathrm{view}}]) \wedge \frac{\pi}{2} - \Delta_{\mathrm{slack}} \leq |\Delta_{t_2}^i - \Delta_{t_2}^a| \leq \frac{\pi}{2} + \Delta_{\mathrm{slack}} \wedge \mathrm{intersect}$.
- **Blind Corner**: An agent approaches a person from an initially occluded position, like a corner or a narrow doorway. In situations with such a limited visibility, the agent should act cautiously to avoid crashes. IR: $(\mathrm{blind}(t) \ \forall t \in [t_1, t_1 + T_{\mathrm{blind}}]) \wedge \mathrm{d\_diff}(t_1) \leq 0.5$ where T_{blind} is the number of initial timesteps in which the person must not be visible by the agent.
- **Person following**: A person and the agent travel in the same direction. The agent must maintain a safe distance from the person and a relatively low linear velocity. IR: $(\neg \, \mathrm{blind}(t) \ \forall t \in [t_1, t_1 + T_{\mathrm{blind}}]) \wedge |\Delta_{t_2}^i -$

$\Delta_{t_2}^a| \leq \Delta_{\mathrm{slack}}$.

Metrics. For each encounter category, we evaluate:

- **Encounter Survival Rate (ESR)** is the percentage of encounters (in a specific category) without a human collision (e.g., if in the Blind Corner encounter the agent collided with a human in the 20% of the cases, the ESR will be 80%);
- **Average Linear-Velocity (ALV)** is the average linear velocity of the agent in an encounter;
- **Average Distance (AD)** is the average distance of the agent w.r.t. the human in an encounter.

6. Experiments

Datasets and training procedure. We performed our experiments using Gibson4+ and HM3D-S, a new dataset based on HM3D and adapted for social navigation. Gibson4+ contains 86 scenes taken from the original Gibson dataset [43]. For training, we used 64 scenes, while 8 and 14 environments were used for validation and test, respectively. HM3D-S is a dataset for Embodied Social Navigation that we generated on top of HM3D [31]. It consists of 900 scenes (800 used for training, 30 for validation, and 70 for test) with, on average, a larger walkable surface compared to Gibson4+. We have generated 8M episodes for the training set (10k per scene), 450 for the validation set (15 episodes per environment), and 490 for the test set (7 episodes per environment). Each episode is obtained by selecting a starting point and a goal point for the agent from the navigable area of the environment (such that it exists a navigable path from one to the other). Pedestrians are included as in [47]; namely, each person is positioned on a starting point and navigates back and forth to an endpoint with a random linear velocity between 0.45 and $0.5m/s$. On Gibson4+, we trained each model for ≈ 100M steps of experience (2.5 days training). On HM3D-S, we fine-tuned our models for ≈ 40M steps (1-day training) starting from the final checkpoint obtained on Gibson4+. This was done to reduce the computational cost of each training.

6.1. Results

Evaluation Metrics. We used standard evaluation metrics for Point Goal Navigation such as *Success Rate* and *Success weighted by Path Length* (SPL) (as in previous works [1]) and STL (success weighted by time length). To evaluate its safety properties, we used *Human Collision Rate*, which is the percentage of episodes that end with failure caused by hitting a person, and PSC (personal space compliance). We run all our experiments with five runs to assess the mean and standard deviation for every metric, as done in [47].

Baseline models. We compared our approach to two baseline models: the model presented in [47] (referred as to *Baseline*) and a "simplified" version of our model that only

Method	Sensors		Aux	Proximity		Metrics (Gibson4+)					Metrics (HM3D-S)				
	RGB	Depth		Risk	Compass	Success	SPL	H-Coll	PSC	STL	Success	SPL	H-Coll	PSC	STL
Baseline [47]		✓				72.65±1.6	47.43±1.2	24.35±1.9	90.13±0.6	41.95±2.5	62.76±2.2	36.69±1.1	29.29±2.2	93.33±0.5	35.10±1.6
Baseline+RGB [47]	✓	✓				74.28±1.8	44.84±0.7	23.78±1.3	91.57±0.7	42.95±0.3	61.43±0.5	34.84±0.6	29.23 ± 0.7	93.16±0.6	35.60±0.5
Aux [45]	✓	✓	✓			73.4±2.0	52.08±1.4	23.40±1.5	91.48±0.4	44.49±1.6	63.62±1.6	42.27±1.2	24.79±2.2	94.18±0.5	39.91±1.7
Risk only	✓	✓		✓		74.90±1.7	50.25±1.1	22.56±1.2	92.34±0.4	43.47±1.0	66.22±1.2	45.26±0.8	24.47±1.7	94.87±0.5	40.03±1.1
Compass only	✓	✓			✓	75.08±1.5	50.55±1.0	22.49±1.1	91.16±0.9	44.21±0.5	67.32±1.7	45.74±1.0	23.54±1.7	94.24±0.3	41.47±1.6
Aux+Risk	✓	✓	✓	✓		75.61±1.8	51.43±0.2	21.04±1.4	90.79±0.7	47.10±0.6	68.16±0.8	45.64±0.2	22.00±1.6	94.96±0.3	42.20±0.8
Aux+Compass	✓	✓	✓		✓	75.63±1.2	52.60±1.6	23.17±1.2	91.00±0.4	45.52±1.9	67.94±1.4	45.76±1.0	23.78±2.0	94.98±0.2	40.80±1.4
Proximity tasks	✓	✓		✓	✓	76.6±1.8	52.81±1.2	20.47±0.4	**93.03±1.3**	47.31±1.2	68.35±0.5	45.83±0.5	**21.72±1.2**	**95.03±0.1**	40.99±0.8
Proximity+Aux	✓	✓	✓	✓	✓	**77.24±1.1**	**55.23±1.4**	**19.50±1.0**	92.78±0.4	**48.71±0.5**	**70.16±1.1**	**47.60±1.0**	22.09±1.3	94.48±0.5	**43.24±1.0**

Table 1: Social Navigation evaluation on Gibson4+ and HM3D-S. For each model are listed the type of input data it uses (*Sensors* column) and, eventually, what kind of self-supervised *Aux tasks* or *Proximity tasks* the model employs. The metrics reported are *Success* rate, *SPL*, Human-Collisions Rate (*H-Coll.*), *PSC* and *STL*.

uses a set of 3 self-supervised auxiliary tasks (inspired by [45]): namely, 2 CPC|A tasks (respectively using 2 and 4 steps) and GID (4 steps). These are referred as to *Aux tasks*. Moreover, the *Baseline* model only uses the depth channel as input. Since we believe that RGB input is fundamental for people recognition and trajectory prediction, we also introduce an adapted version of this model that uses all the information from RGB-D frames (called *Baseline+RGB*).

Proximity-aware models and auxiliary tasks. Firstly, to evaluate each proximity task contribution, we experimented with single-GRU models equipped with just one Proximity-Aware task at a time (i.e., *Risk-only* or *Compass-only*). Since the tasks deal with two different aspects of social navigation (short-range and long-range), we then tried to combine them in a two-GRU model (referred as to *Proximity tasks*). Finally, we combined our approaches with the self-supervised auxiliary tasks presented in [45], which reported state-of-the-art performance on PointGoal Navigation. We have thoroughly investigated the benefit of combining them with single proximity tasks (*Aux+Risk* and *Aux+compass*), as well as combining them all in our final model (*Proximity+Aux tasks*). We now discuss our results and highlight the main takeaways.

Performance analysis and comparison to prior work. Table 1 reports the social navigation performance (on the test set) for both Gibson4+ and HM3D-S. In both cases, *Aux tasks* appears as the strongest of our baselines (highest SPL and STL and lowest Human-Collision for both datasets), reaching comparable performances to single proximity task models while having a higher SPL. Our initial hypothesis that integrating the Baseline with an RGB signal could benefit performances was partially supported by the results on Gibson4+. However, the trend shifted on HM3D-S (the baseline has higher Success and SPL, comparable Human-Collision and STL). This happens because of the higher quality of scene reconstruction in Gibson4+ (scenes have been manually rated and are among the best in the original Gibson dataset). Comparatively, HM3D-S has more reconstruction errors that, while leaving depth-only policies unaffected, may impair the performance of RGB-enabled models. Overall, since human-agent interaction is sparse, we

find that PSC is not very informative (as its values are very close to 100% and the difference in performance between comparable models is almost always non-significant). We argue that our fine-grained evaluation protocol is a better tool for assessing personal space violations.

Moreover, we notice that both models that use just one Proximity-aware task perform similarly on Gibson4+ (sub 0.7% of difference between metrics). However, this changes on HM3D-S, where Compass-only slightly outperforms Risk-only (+1.1% Success, -0.93% h-collisions, +1.44% STL). This difference is expected since the proximity compass task explicitly aims to deal with long-range proximity information. Being HM3D scenes larger in size, the proximity compass role becomes more important.

Adding self-supervised tasks significantly increases SPL and Success performances (both for single-task and all Proximity-tasks models). It also appears to positively affect Human Collision when combined with Risk (-1.52% in Gibson4+, -2.47% in HM3D-S). It also boosts STL performance (with the only exception of Aux+Compass on HM3D-S). We hypothesize that self-supervised tasks, since they are either action-based contrastive tasks (CPC|A) or tasks that try to retrieve the inverse dynamics of navigation (GID), help Proximity-Aware models to have smoother trajectories thanks to a more accurate linear and angular velocity dialing. This claim will be substantiated by the fine-grained analysis reported in the next section. Overall, the best results are obtained by combining all tasks together in the same model.

6.2. Fine-grained evaluation

We report the results obtained on the Gibson4+ dataset by applying our evaluation protocol (see Section 5) to understand better how each model operates and the types of encounters they have. Figure 5 summarizes the statistics, in terms of number of encounters and ESR, collected for each encounter class by different models during 500 randomly sampled validation episodes.

Policy behaviour analysis. Looking at the relationships between the number of encounters and ESR figures, there seem to be two types of policies: a first group with a high

(a) Number of encounters
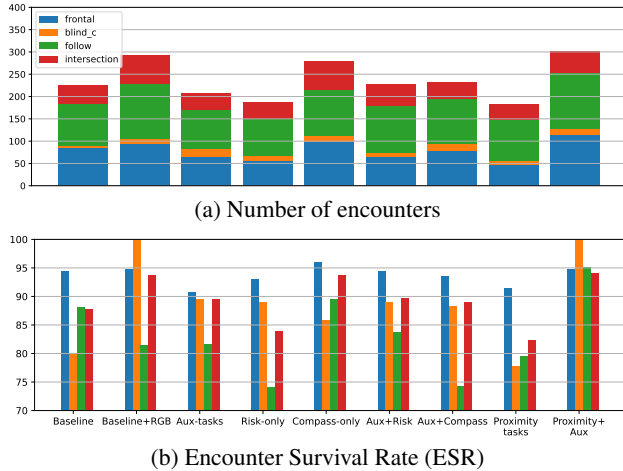

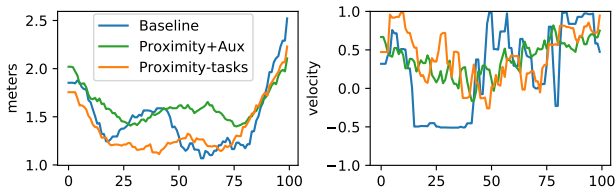(b) Encounter Survival Rate (ESR)

Figure 5: Number of encounters and ESR for all the models.

Figure 6: Average distance (left) and linear velocity (right) *vs* percentage of completion for Blind-Corner encounters.

Figure 7: Two success cases: a frontal encounter (top) and an intersection encounter (bottom).

number of encounters and a high ESR, and a second group that tends to avoid encounters and has low/medium ESR. Those two types describe different approaches to social navigation: either risking to interact to access potentially more efficient routes or keeping a safe distance from humans (for example, prioritizing less populated areas, waiting for people to move away before crossing a room).

An example of a policy that avoids encounters is *Risk only*, which has one of the lowest ESR for Following (74.11%) and Intersection class (86.85%) and the lowest number of encounters (188 in total). The opposite is true for *Compass only*, which has a high overall ESR for every encounter class and one of the highest numbers of encounters (279 in total). It is interesting to note how the two best-performing policies, *Proximity tasks* and *Proximity+Aux*, adopt each of these different approaches while remaining comparable in coarse-grained metrics.

Reacting to sudden danger. A critical ability that a Social-Nav policy must possess is the capability to react to immediate and sudden danger in situations with limited visibility. The class of encounters that better represents this is the *blind corner* class. To investigate how our two best models react compared to the baseline, we plotted the average distance (AD) and average linear velocity (ALV) vs percentage of completion of all blind corner encounters (see Figure
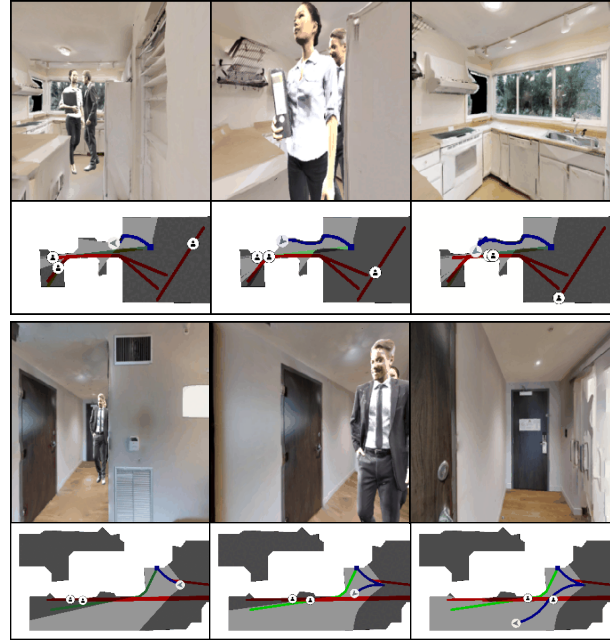
6). We notice how the ALV curves are not smooth, reflecting uncertainty and high risk. However, while the baseline needs to brake and backtrack ($\approx$ -0.5 ALV between 20% and 40% of the episodes), the other models tend to maintain a positive and proportionate ALV velocity throughout the episode. We also see that the *Proximity+Aux* ALV curve is smoother than the one for *Proximity tasks*. The same phenomenon is true for each single Proximity-Aware task model and their self-supervised task counterpart (see supplementary material). This supports the claim that self-supervised tasks provide a smoothing effect on action dynamics under uncertainty.

Qualitative results. Figure 7 shows two qualitative examples of successfully managed encounters in Gibson4+. The first example (on top) depicts a Frontal encounter. After seeing people (first frame), the agent moves to the side and yields, letting them move away (second frame). Finally, the agent can reach the goal (third frame). In the second example (an intersection encounter), The agent sees a pedestrian (first frame), then it yields, letting the pedestrian pass (second frame), and finally, it continues on its path.

Limitations. A significant limitation of Social navigation in the Embodied-AI literature is the simplicity of trajectory and human movement simulation. To the best of our knowledge, among the ones currently used (such as Habitat, iGibson or AI2Thor), no simulators implement humans with complex navigation patterns and natural-looking animations. This lack implies that social elements like social cues and proxemics are currently not exploited by Embodied-AI

models. Our work inherited such problems. However, even under this simplistic approach, the behaviour learned by our agents and the properties we discussed in this work represent a strong first step for future work in this area.

7. Conclusion

We introduced a model for Embodied Social Navigation based on two *Proximity-Aware tasks*. Our experiments show that exploiting social signals, alone or in combination with self-supervised auxiliary tasks, is an effective strategy in complex and crowded scenarios. Our model can avoid the majority of encounters by using only Proximity-Aware tasks. Furthermore, by combining Proximity-Aware and auxiliary tasks [45], it can prevent human collisions with high confidence in almost all encounter classes, despite the increased number of them. In our future work, we will focus on simulating more natural human behaviours and we will experiment on sim2real domain transfer.

References

[1] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir R. Zamir. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018.

[2] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[3] Georges S. Aoude, Brandon D. Luders, Joshua M. Joseph, Nicholas Roy, and Jonathan P. How. Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns. *Autonomous Robots*, 35(1):51–76, 2013.

[4] Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. ObjectNav revisited: On evaluation of embodied agents navigating to objects. *arXiv preprint arXiv:2006.13171*, 2020.

[5] Jur van den Berg, Stephen J Guy, Ming Lin, and Dinesh Manocha. Reciprocal n-body collision avoidance. In *Robotics Research*, pages 3–19. Springer, 2011.

[6] Tommaso Campari, Paolo Eccher, Luciano Serafini, and Lamberto Ballan. Exploiting scene-specific features for ob-

ject goal navigation. In *Proc. of the European Conference on Computer Vision Workshops (ECCVW)*, 2020.

[7] Tommaso Campari, Leonardo Lamanna, Paolo Traverso, Luciano Serafini, and Lamberto Ballan. Online learning of reusable abstract models for object goal navigation. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[8] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. In *Proc. of the International Conference on 3D Vision (3DV)*, 2017.

[9] Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Ruslan Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[10] Devendra Singh Chaplot, Ruslan Salakhutdinov, Abhinav Gupta, and Saurabh Gupta. Neural topological slam for visual navigation. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[11] Changan Chen, Yuejiang Liu, Sven Kreiss, and Alexandre Alahi. Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[12] Yu Fan Chen, Michael Everett, Miao Liu, and Jonathan P. How. Socially aware motion planning with deep reinforcement learning. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[13] Yu Fan Chen, Miao Liu, Michael Everett, and Jonathan P. How. Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[14] Matt Deitke, Dhruv Batra, Yonatan Bisk, Tommaso Campari, Angel X. Chang, Devendra Singh Chaplot, Changan Chen, Claudia Pérez D'Arpino, Kiana Ehsani, Ali Farhadi, et al. Retrospectives on the embodied ai workshop. *arXiv preprint arXiv:2210.06849*, 2022.

[15] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Jordi Salvador, Kiana Ehsani, Winson Han, Eric Kolve, Ali Farhadi, Aniruddha Kembhavi, et al. Procthor: Large-scale Embodied AI using procedural generation. *arXiv preprint arXiv:2206.06994*, 2022.

[16] Gonzalo Ferrer, Anais Garrell, and Alberto Sanfeliu. Social-aware robot navigation in urban environments. In *Proc. of thr European Conference on Mobile Robots*, 2013.

[17] Yuxiang Gao and Chien-Ming Huang. Evaluation of socially-aware robot navigation. *Frontiers in Robotics and AI*, 8:420, 2022.

[18] Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Bernardo A. Pires, and Rémi Munos. Neural predictive belief representations. *arXiv preprint arXiv:1811.06407*, 2018.

[19] Jérôme Guzzi, Alessandro Giusti, Luca M Gambardella, Guy Theraulaz, and Gianni A Di Caro. Human-friendly robot navigation in dynamic environments. In *Proc. of the*

IEEE International Conference on Robotics and Automation (ICRA), 2013.

[20] Haresh Karnan, Anirudh Nair, Xuesu Xiao, Garrett Warnell, Sören Pirk, Alexander Toshev, Justin Hart, Joydeep Biswas, and Peter Stone. Socially compliant navigation dataset (SCAND): A large-scale dataset of demonstrations for social navigation. *IEEE Robotics and Automation Letters*, 7(4):11807–11814, 2022.

[21] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

[22] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *Proc. of the European Conference on Computer Vision (ECCV)*, 2020.

[23] Pinxin Long, Wenxi Liu, and Jia Pan. Deep-learned collision avoidance policy for distributed multiagent navigation. *IEEE Robotics and Automation Letters*, 2(2):656–663, 2017.

[24] Yazhou Lu, Xiaogang Ruan, and Jing Huang. Deep reinforcement learning based on social spatial–temporal graph convolution network for crowd navigation. *Machines*, 10(8): 703, 2022.

[25] Roberto Martin-Martin, Mihir Patel, Hamid Rezatofighi, Abhijeet Shenoi, JunYoung Gwak, Eric Frankel, Amir Sadeghian, and Silvio Savarese. JRDB: A dataset and benchmark of egocentric robot visual perception of humans in built environments. *IEEE transactions on pattern analysis and machine intelligence*, 2021.

[26] Ronja Möller, Antonino Furnari, Sebastiano Battiato, Aki Härmä, and Giovanni Maria Farinella. A survey on human-aware robot navigation. *Robotics and Autonomous Systems*, 145:103837, 2021.

[27] Ruslan Partsey, Erik Wijmans, Naoki Yokoyama, Oles Dobosevych, Dhruv Batra, and Oleksandr Maksymets. Is mapping necessary for realistic pointgoal navigation? In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[28] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *Proc. of the International Conference on Machine Learning (ICML)*, 2017.

[29] Claudia Pérez-D'Arpino, Can Liu, Patrick Goebel, Roberto Martín-Martín, and Silvio Savarese. Robot navigation in constrained pedestrian environments using reinforcement learning. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[30] Sören Pirk, Edward Lee, Xuesu Xiao, Leila Takayama, Anthony Francis, and Alexander Toshev. A protocol for validating social navigation policies. *arXiv preprint arXiv:2204.05443*, 2022.

[31] Santhosh K. Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alex Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X. Chang, et al. Habitat-matterport 3D dataset (HM3D): 1000 large-scale 3D environments for embodied AI. *arXiv preprint arXiv:2109.08238*, 2021.

[32] Santhosh K. Ramakrishnan, Devendra Singh Chaplot, Ziad Al-Halah, Jitendra Malik, and Kristen Grauman. Poni: Potential functions for objectgoal navigation with interaction-free learning. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[33] Ram Ramrakhya, Eric Undersander, Dhruv Batra, and Abhishek Das. Habitat-web: Learning embodied object-search strategies from human demonstrations at scale. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[34] Andrey Rudenko, Tomasz P Kucner, Chittaranjan S Swaminathan, Ravi T Chadalavada, Kai O Arras, and Achim J Lilienthal. Thör: Human-robot navigation data collection and accurate motion trajectories dataset. *IEEE Robotics and Automation Letters*, 5(2):676–682, 2020.

[35] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied AI research. In *Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[36] Bokui Shen, Fei Xia, Chengshu Li, Roberto Martín-Martín, Linxi Fan, Guanzhi Wang, Claudia Pérez-D'Arpino, Shyamal Buch, Sanjana Srivastava, Lyne Tchapmi, et al. iGibson 1.0: a simulation environment for interactive tasks in large realistic scenes. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[37] Lei Tai, Jingwei Zhang, Ming Liu, and Wolfram Burgard. Socially compliant navigation through raw depth inputs with generative adversarial imitation learning. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[38] Peter Trautman and Andreas Krause. Unfreezing the robot: Navigation in dense, interacting crowds. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010.

[39] Jur Van den Berg, Ming Lin, and Dinesh Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2008.

[40] Saim Wani, Shivansh Patel, Unnat Jain, Angel X. Chang, and Manolis Savva. MultiON: Benchmarking semantic map memory using multi-object navigation. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[41] Erik Wijmans, Samyak Datta, Oleksandr Maksymets, Abhishek Das, Georgia Gkioxari, Stefan Lee, Irfan Essa, Devi Parikh, and Dhruv Batra. Embodied Question Answering in Photorealistic Environments with Point Cloud Perception. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[42] Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. DD-PPO: Learning near-perfect pointgoal navigators from 2.5 billion frames. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2019.

[43] Fei Xia, Amir R. Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson Env: Real-World Perception for Embodied Agents. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*

(CVPR), 2018.

[44] Fei Xia, William B Shen, Chengshu Li, Priya Kasimbeg, Micael Edmond Tchapmi, Alexander Toshev, Roberto Martín-Martín, and Silvio Savarese. Interactive gibson benchmark: A benchmark for interactive navigation in cluttered environments. *IEEE Robotics and Automation Letters*, 5(2):713–720, 2020.

[45] Joel Ye, Dhruv Batra, Erik Wijmans, and Abhishek Das. Auxiliary Tasks Speed Up Learning PointGoal Navigation. In *Proc. of the International Conference on Robot Learning (CoRL)*, 2020.

[46] Joel Ye, Dhruv Batra, Abhishek Das, and Erik Wijmans. Auxiliary Tasks and Exploration Enable ObjectNav. In *Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[47] Naoki Yokoyama, Qian Luo, Dhruv Batra, and Sehoon Ha. Learning Robust Agents for Visual Navigation in Dynamic Environments: The Winning Entry of iGibson Challenge 2021. *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[48] Fengda Zhu, Yi Zhu, Vincent Lee, Xiaodan Liang, and Xiaojun Chang. Deep learning for embodied vision navigation: A survey. *arXiv preprint arXiv:2108.04097*, 2021.