SEAN 2.0: Formalizing and Generating Social Situations for Robot Navigation

Nathan Tsoi, Alec Xiang, Peter Yu, Samuel S. Sohn, Greg Schwartz, Subashri Ramesh, Mohamed Hussein, Anjali W. Gupta, Mubbasir Kapadia, and Marynel Vázquez

***Abstract*—We present SEAN 2.0, an open-source system designed to advance social navigation via the training and benchmarking of navigation policies in varied social contexts. A key limitation of current social navigation research is that policies are often trained and evaluated considering only a few social contexts, which are fragmented across prior work. Inspired by work in psychology, we describe navigation context based on social situations, which encompass the robot task and environmental factors, and propose logic-based classifiers for five common examples. SEAN 2.0 allows a robot to experience these social situations via different methods for specifying and generating pedestrian motion, including a novel Behavior Graph method. Our experiments show that when data collected using the Behavior Graph method is used to learn a robot navigation policy, that policy outperforms others trained using alternative methods for pedestrian control. Also, social situations were found to be useful for understanding performance across social contexts. Other components of SEAN 2.0 include vision and depth sensors, several physical environments, different means of specifying robot tasks, and a range of evaluation metrics for social robot navigation. User feedback for SEAN 2.0 indicated that the system was "easier to navigate and more user friendly" than SEAN 1.0.**

***Index Terms*—Social HRI, Simulation and Animation, Autonomous Vehicle Navigation**

I. Introduction

WHILE a significant amount of work has been done to enable robots to effectively move in human environments [1], prior work has largely been fragmented by interaction scenarios [2]. For example, past work has focused on studying navigation in scenarios where robots cross human paths [3], [4], approach users [5], [6] or groups [7], [8], and move in crowded environments [9], [10]. This fragmentation raises the question: *how can we build robot navigation systems that handle different social contexts*?

Inspired by work in social psychology, we propose to reason about context for social navigation in terms of *social situations*, which consider the interplay between robot task and environmental factors. Social situations may occur in a given interaction *scenario*, consisting of three key elements: 1) the physical *environment* (such as a lab or warehouse), 2) *pedestrian behavior* in the environment, and 3) the robot's navigation *task* (involving motion from a start to a goal pose).

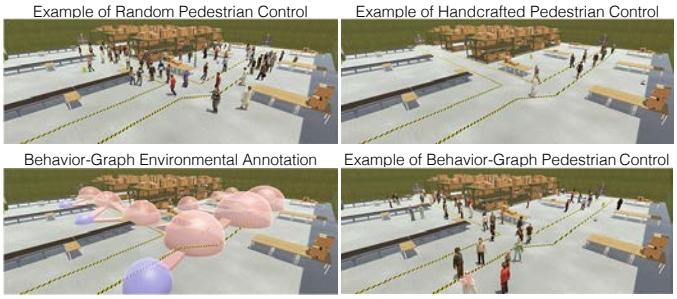

Fig. 1. Different methods for specifying pedestrian behaviors in SEAN 2.0. The Behavior Graph method is a novel approach that uses an environmental graph-based annotation (bottom-left) to generate behavior (bottom-right).

We contribute an open source system, the Social Environment for Autonomous Navigation (SEAN) 2.0, for training and benchmarking social navigation algorithms. Unlike other robotics simulation environments capable of high visual fidelity such as [11], SEAN 2.0 is designed so that robots can experience a range of different pedestrian behaviors, which result in varying social situations. To ground the concept of social situations in our system, we propose logic-based definitions for five social situations relevant to navigation. These definitions serve as situation classifiers in SEAN 2.0.

One approach to specifying pedestrian behaviors for simulated interactions in social navigation is to handcraft a starting pose and a goal pose for each pedestrian in the scene. Handcrafting pedestrian behavior is time consuming and specific to a single implementation, as evidenced by the limited number of social situations commonly employed when evaluating navigation policies [2]. Randomly choosing start and goal poses is an easy alternative to handcrafting; yet, as our experiments show, it is less likely to result in varied social situations in practice.

As part of SEAN 2.0, we propose a novel method for specifying pedestrian behavior based on a *Behavior Graph* annotation in the physical environment (Fig. 1, bottom). We define the Behavior Graph such that nodes represent either static group formations or navigation waypoints, and compute flow between nodes based on graph parameters. Pedestrians traverse the scene by walking between different nodes in the graph. This creates opportunities for the robot to experience different social situations while avoiding time-intensive handcrafting of pedestrian motion (as in SEAN 1.0 [12]).

SEAN 2.0 also provides a range of components to enable the training and benchmarking of navigation policies, including vision and depth sensors, several physical environments, differ-

Manuscript received: February 24, 2022; Revised June 22, 2022; Accepted July 26, 2022. This paper was recommended for publication by Editor G. Venture upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by a 2020 Amazon Research Award.

Nathan Tsoi, Alec Xiang, Peter Yu, Greg Schwartz, Subashri Ramesh, Anjali W. Gupta, and Marynel Vázquez are with Yale University, Arthur K. Watson Hall, New Haven, CT 06511, USA nathan.tsoi@yale.edu.

Samuel S. Sohn, Mohamed Hussein, and Mubbasir Kapadia are with Rutgers University, 110 Frelinghuysen Road, Piscataway, NJ 08854, USA.

Digital Object Identifier (DOI): see top of this page.

ent means of specifying robot tasks, and a range of evaluation metrics. To validate that SEAN 2.0 would be useful to the robotics community, we collected feedback from 7 roboticists who were early users of the system and incorporated their feedback in the final version of SEAN 2.0.

As part of our experimental evaluation, we studied the distribution of social situations that emerged in datasets gathered with different methods of pedestrian control according to logic-based social situation classifiers. We found that the Behavior Graph data resulted in more varied social situations than the data generated with handcrafted or random pedestrian motion. Also, policies trained on Behavior Graph data outperformed other learned policies that were trained using alternative methods for pedestrian behavior generation in SEAN 2.0. Finally, our experiments showed that analyzing navigation policies by social situation can reveal new insights about robot policy performance.

In summary, our five main contributions are: 1) SEAN 2.0, a novel system for training and benchmarking social navigation systems; 2) a logical formalization of social situations; 3) multiple methods for pedestrian behavior generation including a novel Behavior Graph approach; 4) validation that our system is useful to users outside our team via feedback from other roboticists; and 5) experiments that show the usefulness of social situations in SEAN 2.0 towards the training and evaluation of robot navigation systems.

II. Related Work

A. Simulation Frameworks for Social Navigation

Our work builds on developments in robotics simulators. Recently, robotics simulators such as CARLA [11], iGibson [13], and Habitat [14] have focused on creating high-fidelity environments and have started to provide basic control of individual pedestrians. Several works have extended the MORSE robotics simulator [15], adding humans that react to a robot [16] and humans in wide areas or narrow passages [17]. The MORSE simulator integrates with the Robot Operating System (ROS); however, visual fidelity is low in comparison to simulators based on game engines such as CARLA.

Crowd simulation frameworks such as Nomad [18], PEDSIM [19], and Menge [20] incorporate methods of individual and group behavior control into a system, but do not integrate robotics platforms to train and evaluate social robot navigation systems. Their strength lies in simulating pedestrian motion, not in integrating them in realistic physical environments or in the visualisation of pedestrians necessary for training state-of-the-art vision-based robotics algorithms.

Our prior work, SEAN 1.0 [12], was designed for training and evaluating social navigation algorithms. It supported integration with ROS and high-quality rendering of virtual pedestrians. However, it only provided simple waypoint navigation for the pedestrians, requiring time-consuming handcrafting of their behavior. More specifically, SEAN 1.0 allowed users to specify pedestrian's start and goal locations and implemented only one example set of such handcrafted pedestrian start and goal location annotations in three physical environments. SEAN 2.0 continues to provide the same ability to customize start and goal locations for pedestrians, but provides 13 sets of handcrafted start and goal positions per environment (39 total sets) across 5 social situations. This allows users of SEAN 2.0 to generate more varied pedestrian behavior with the handcrafted approach for pedestrian control out of the box. Further, we found that our proposed Behavior Graph, a novel method for specifying pedestrian behavior, was a superior method for training a navigation policy. See Section V for more details.

An alternative approach to evaluating social navigation policies is using prerecorded pedestrian trajectories as in SocNavBench [21]. Prerecorded pedestrian trajectories offer realistic motion, but are not reactive to the robot during policy rollout. Our work compliments [21] by allowing for dynamic human-robot interactions. To our knowledge, SEAN 2.0 is the only robotics simulation environment capable of high visual fidelity that provides easy-to-customize, dynamic pedestrian behaviors, including group formations.

B. Modeling Pedestrian Behaviors

Algorithms for the animation and control of virtual characters have been studied by different disciplines such as computer graphics [22], cognitive science [23], and computer vision [24]. The generation of collective behaviors has traditionally focused on modeling individual members of a crowd to elicit human-like behavior from the group. For example, flocks of animals inspired Reynolds et al.'s early work on modeling groups of pedestrians [25]. Pelechano et al. [26] focused their effort on imbuing human-like perception and decision making capabilities into individual agents to elicit more realistic group behavior. Collective behavior conditioned on a given environment can rely on annotations of the physical space, such as semantically relevant descriptors [27]. We take inspiration from these ideas and utilize environmental annotations in our proposed Behavior Graph.

To produce phenomena observed in human navigation, collision avoidance methods for individual agents are often used to compliment collective behavior. Such methods, often referred to as microscopic models, include the Social Forces model [28] and the velocity obstacle method of the ORCA model [29]. While ORCA's primary benefit is collision-free movement between a large number of agents, the Social Forces model is easily extended by the addition of new forces. For this reason, our proposed Behavior Graph relies on the Social Forces model. In the future, SEAN 2.0 could be extended with other microscopic models.

III. Formalizing Social Navigation Context

There are many definitions of context in disciplines related to social navigation. For example, in social signal processing, context has been defined as the who, what, when, where, and why of interactions [30]. In Human-Robot Interaction (HRI), the term context has been used to refer to high-level environmental concepts such as an art gallery or a dining hall [31]. Likewise, context has been used to describe the relationship between agents (human and robot) in the scene, such as agents in a static group formation or standing in a line

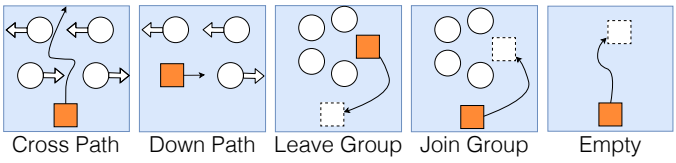

Fig. 2. A brief visual description of *social situations*. Pedestrians are denoted as white circles and the robot as an orange square.

[32]. Task-based context has also been explored in HRI, often in the domain of engagement [33].

In this work, we propose to reason about context in social robot navigation based on the notion of social situations proposed by Argyle et al [34] in psychology. Those authors studied the interplay between internal and external determinants for human behavior. In their work, social situations encompass the intrinsic *goal* of a person and the extrinsic *environment* in which this person acts. Individuals' goals arise from an underlying drive which satisfies a specific need.

Consequently, we propose to reason about social situations in robot navigation as a construct that considers the interplay between a robot's *task* and *environmental* factors. Consider for example a situation when a robot must cross a pedestrian path to reach the other side [35]. This situation arises from the combination of the environmental factor of pedestrian traffic and the robot's start location relative to a navigation goal. Similarly, a robot approaching a group [8] could be considered another example of a social situation. In this case, the task is navigating to a specific goal position in a conversational group and depends on people's spatial arrangement.

A. Logical Expressions for Social Situations

This section operationalizes the proposed notion of social situations in relation to five instances relevant to navigation, as shown in Fig. 2. In particular, we consider situations that involve both pedestrians in motion and static group formations. Although our proposed set of social situations may not be complete for all robot navigation applications, it helps demonstrate the value of formalizing social situations for mobile robotics.

We use logic to formally define the proposed situations. The domain of predicates, defined below, are vectors in $\mathbb{R}^2$ that represent position, orientation (as unitary direction vectors), or velocity. These vectors can be provided in simulation or estimated in the real world.

- **Nearby Agent:** $Near(\boldsymbol{x_1}, \boldsymbol{x_2})$ is true when two agents at positions $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ are separated by $||\boldsymbol{x_1} - \boldsymbol{x_2}|| < \mathsf{D}$.
- **Group Member:** $Member(\boldsymbol{x}, \boldsymbol{g})$ is true when agent $\boldsymbol{x}$ is a member of the group with center at position $\boldsymbol{g}$.
- **Walking:** $Walking(\boldsymbol{v})$ is true when an agent is moving at a velocity $\boldsymbol{v}$ where $||\boldsymbol{v}|| > \mathsf{V}$.
- **Perpendicular Trajectory:** $PerpTraj(\boldsymbol{d_1}, \boldsymbol{d_2})$ is true when the orientations of two agents $\boldsymbol{d_1}$ and $\boldsymbol{d_2}$ are perpendicular within an error of $\pm\mathsf{A}$ rad:

$$cos(\frac{\pi}{2} + \mathsf{A}) \leq \frac{\boldsymbol{d_1} \cdot \boldsymbol{d_2}}{||\boldsymbol{d_1}||||\boldsymbol{d_2}||} \leq cos(\frac{\pi}{2} - \mathsf{A})$$

- **Parallel Trajectory:** $ParTraj(\boldsymbol{d_1}, \boldsymbol{d_2})$ is true when the orientations of two agents $\boldsymbol{d_1}$ and $\boldsymbol{d_2}$ are parallel within an error of $\pm\mathsf{A}$ rad:

$$cos(\mathsf{A}) \leq \frac{\boldsymbol{d_1} \cdot \boldsymbol{d_2}}{||\boldsymbol{d_1}||||\boldsymbol{d_2}||}$$

All predicates are defined by simple geometric relationships except for group membership. We assume group membership is provided by the simulator or a method such as [36], [37] which reasons about conversational formations [38]. Section IV-G provides more details of our specific choice of other parameters for these predicates.

The five *Social Situations* (Fig. 2) are expressed as:

Cross Path: a robot is at a position $\boldsymbol{x_r}$ with orientation $\boldsymbol{d_r}$. Also, it is nearby an agent at $\boldsymbol{x_a}$, moving at velocity $\boldsymbol{v_a}$, with orientation $\boldsymbol{d_a}$ perpendicular to $\boldsymbol{d_r}$.

$$\begin{aligned} &CrossPath(\boldsymbol{x_r}, \boldsymbol{d_r}, \boldsymbol{x_a}, \boldsymbol{v_a}, \boldsymbol{d_a}) \equiv \\ &Near(\boldsymbol{x_r}, \boldsymbol{x_a}) \wedge Walking(\boldsymbol{v_a}) \wedge PerpTraj(\boldsymbol{d_r}, \boldsymbol{d_a}) \end{aligned} \tag{1}$$

Down Path: a robot is at position $\boldsymbol{x_r}$ with orientation $\boldsymbol{d_r}$. Also, it is nearby an agent at $\boldsymbol{x_a}$, moving at velocity $\boldsymbol{v_a}$, with orientation $\boldsymbol{d_a}$ parallel to $\boldsymbol{d_r}$.

$$\begin{aligned} &DownPath(\boldsymbol{x_r}, \boldsymbol{d_r}, \boldsymbol{x_a}, \boldsymbol{v_a}, \boldsymbol{d_a}) \equiv \\ &Near(\boldsymbol{x_r}, \boldsymbol{x_a}) \wedge Walking(\boldsymbol{v_a}) \wedge ParTraj(\boldsymbol{d_r}, \boldsymbol{d_a}) \end{aligned} \tag{2}$$

Joining Group: a robot at position $\boldsymbol{x_r}$ has a navigation goal $\boldsymbol{x_r'}$, which corresponds to a location that would make the robot a member of a group with a center at $\boldsymbol{g}$. The robot is also nearby an agent at $\boldsymbol{x_a}$, which is a member of the same group. Note that once the robot arrives at the goal, $JoinGroup$ is no longer true.

$$\begin{aligned} &JoinGroup(\boldsymbol{x_r}, \boldsymbol{x_a}, \boldsymbol{x_r'}, \boldsymbol{g}) \equiv Near(\boldsymbol{x_r}, \boldsymbol{x_a}) \wedge \\ &Member(\boldsymbol{x_a}, \boldsymbol{g}) \wedge Member(\boldsymbol{x_r'}, \boldsymbol{g}) \wedge \neg(\boldsymbol{x_r} = \boldsymbol{x_r'}) \end{aligned} \tag{3}$$

Leave Group: a robot that is currently at a position $\boldsymbol{x_r}$, had a starting position $\boldsymbol{x_r''}$ which made it a member of a group with a center at $\boldsymbol{g}$. The robot is nearby an agent located at $\boldsymbol{x_a}$, which is a member of the same group.

$$\begin{aligned} &LeaveGroup(\boldsymbol{x_r}, \boldsymbol{x_a}, \boldsymbol{x_r''}, \boldsymbol{g}) \equiv \\ &Near(\boldsymbol{x_r}, \boldsymbol{x_a}) \wedge Member(\boldsymbol{x_a}, \boldsymbol{g}) \wedge Member(\boldsymbol{x_r''}, \boldsymbol{g}) \end{aligned} \tag{4}$$

Empty: a robot at position $\boldsymbol{x_r}$ has no other agents nearby. Let X be the set of positions for all other agents in the environment, then:

$$Empty(X, \boldsymbol{x_r}) \equiv \forall \boldsymbol{x} \in X, \neg Near(\boldsymbol{x}, \boldsymbol{x_r}) \tag{5}$$

The satisfiability of these logical expressions depends on the agents in the environment. Only when a sufficient number of agents are present, both moving and in static group formations, are all non-empty expressions satisfiable. For example, in environments without group formations, $JoinGroup$ and $LeaveGroup$ are not satisfiable. The size of the environment also impacts satisfiability. For example, consider an environment with a robot and a pedestrian. The $Empty$ proposition is not satisfiable if the navigable space in the environment is smaller than the nearby distance D.

IV. SEAN 2.0 System

SEAN 2.0 builds on our prior work, the Social Environment for Autonomous Navigation version 1.0 [12]. SEAN 1.0 and SEAN 2.0 both use the Unity game engine and the Robot Operating System (ROS) [39] as underlying technologies, and can be integrated with online interactive surveys [40]. The core innovation in SEAN 2.0 is a variety of methods for specifying pedestrian behavior, including a novel Behavior Graph approach that induces the proposed social situations described in Sec. III. In addition, SEAN 2.0 provides improved

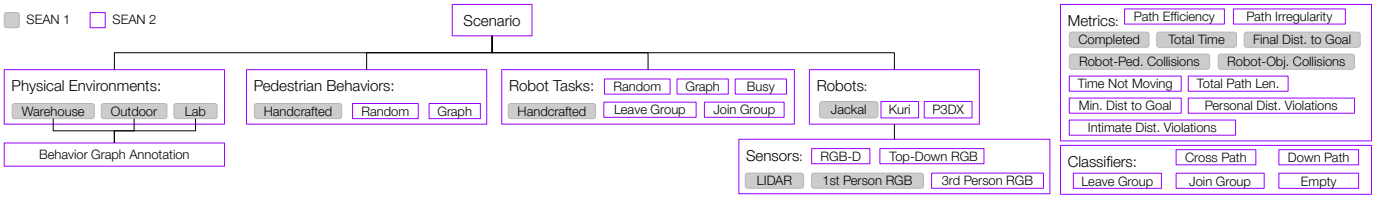

Fig. 3. SEAN 2.0 system architecture including components that were re-used or adapted from SEAN 1.0 (rounded box, grey) and new to SEAN 2.0 (purple). Connections denote relationships between components in a Scenario. The Scenario, Metrics, and Classifiers are part of the SEAN 2.0 Unity API and exist for all scenes. The SEAN 1.0 Trial Runner [12] is superseeded by Robot Tasks and the Metrics system in SEAN 2.0. Warthog is the only robot in SEAN 1.0 which is not in SEAN 2.0 due to it's unwieldy size relative to people. See the text for details.

simulated sensors to facilitate vision-based, trained policies and two new features. One new feature is a set of logic-based social situation classifiers and the other is a revamped software architecture. Our goal was to make SEAN 2.0 easily configurable for users of its graphical user interface and easily extensible for users of its programming interface. Fig. 3 shows the system components of SEAN 2.0, including those that are re-used or adapted from SEAN 1.0.

A. Software Design

The usability of a software system depends directly on the underlying design decisions [41]. Therefore, we designed SEAN 2.0 following the singleton design pattern [42] and taking a convention over configuration approach [43].

The SEAN Unity API is implemented as a singleton GameObject, through which all key components can be accessed. Unlike other scripts that may be added or removed from the Unity scene at various times, this object exists throughout the duration of the simulation and provides the logic necessary to wrap other elements that may be removed or added at various times. The singleton GameObject includes all elements discussed below such as pedestrian behaviors, robot tasks, social situation classifiers, metrics, and other utilities such as a simulated clock and a tool for creating ROS maps. The singleton GameObject can be added to any scene thereby making it compatible with SEAN 2.0.

Classes in SEAN 2.0 use a convention over configuration approach by providing sensible defaults [43]. This makes our system easier to use than SEAN 1.0, which had an ad-hoc design. Feedback from early users of SEAN 2.0 indicates that our design choices provide a better user experience than SEAN 1.0 (see Section V-A for more details).

B. System Architecture

Our system architecture was designed to encapsulate the elements of a *Scenario*, which consist of the *Physical Environment*, *Pedestrian Behavior*, and *Robot Task*. These elements are shown in Fig. 3 and correspond to objects in the SEAN 2.0 singleton GameObject. *Physical Environments* are locations in which a scenario occurs, such as a warehouse. *Pedestrian Behaviors* specify pedestrian motion. *Robot Tasks* specify a robot's start pose and goal pose.

C. Physical Environments

Environments correspond to the physical, static elements in a scenario and are composed of 3D meshes, textures, lights, and colliders that construct a Unity Scene. Static elements of the environment constrain agent motion and define a navigable area on the ground plane. SEAN 2.0 includes the *warehouse*, *lab*, and *outdoor* environments from SEAN 1.0 with annotations for our new pedestrian behaviors.

D. Pedestrian Behaviors

SEAN 2.0 supports three different approaches to high-level pedestrian control: *random*, *handcrafted*, and *graph*. High-level pedestrian motions are defined by their start and goal poses. They also depend on a low-level collision avoidance mechanism, which relies on the Social Forces model [28]. We extend the Social Forces model with consideration for pedestrians moving along one side of a hallway [44] and to stochastically vary the distances that they prefer to maintain from the robot [45]. The next sections describe the three methods for pedestrian behavior generation in SEAN 2.0.

Random Pedestrian Behavior: Start and goal locations are randomly chosen on the environment's navigable plane.

Implementation. There are no parameters other than the number of pedestrians in the scenario. This approach for pedestrian behavior generation is the easiest to implement, but no group formations are created by this method. We use this behavior as a baseline in our experiments to evaluate the effect of pedestrian density on policy performance relative to the other methods of pedestrian control in SEAN 2.0.

Handcrafted Pedestrian Behavior: Start, goal, and intermediary waypoint poses for the pedestrians are chosen manually in each environment and may be designed to resemble specific social situations. This is the most granular method of pedestrian control. The main challenge with this method is twofold: 1) it can be time consuming, and 2) the many low level decisions one has to make in regard to goal placement may not align with the intended high level behavior. This challenge is further discussed in Section V.

Implementation. In each of the environments, we implemented 13 unique sets of start and goal poses for pedestrians across five handcrafted scenarios to resemble each social situation described in Section III-A. Pedestrians in the Join Group and Leave Group scenarios are configured in a static group formation typical of conversational encounters [38]. In the Cross Path and Down Path scenarios, we specified navigation waypoints along a path so that the robot can cross parallel or travel perpendicular to the path of pedestrian waypoints. While choosing group locations and pedestrian waypoints, we also chose corresponding poses for the robot in specific tasks, described in Section IV-E.

Some handcrafted pedestrian behaviors are parameterized by the location of static group formations. Given a group center, we set the poses of individual group members by mimicking conversational formations in the Cocktail Party dataset [46] in a manner similar to [8].

Graph-based Pedestrian Behavior: We propose using a directed graph abstraction, which we call the Behavior Graph, to specify collective pedestrian behaviors. A graph annotation is overlaid in the environment (Fig. 1, bottom-left). The graph parameterizes pedestrian motion via two types of nodes that determine pedestrian behavior. Nodes serve as either 1) navigational waypoints (through which an unrestricted number of pedestrians can continuously flow) or 2) as a location for a conversational group formation (where a number of pedestrians can enter a static group formation for a specific duration). The graph edges connect nodes that pedestrians can navigate through.

On initialization, individual pedestrians are stochastically assigned to starting locations, which correspond to specific nodes from the Behavior Graph annotation. During the simulation, pedestrians without an assigned goal position are first assigned to group nodes, until all group nodes are at capacity. When pedestrians assigned to group nodes reach their destinations, they remain at the group for a given duration. Once all the group nodes have reached capacity, the remaining pedestrians are stochastically assigned to waypoint nodes. This allows the simulation to maintain group formations and it allows pedestrians to automatically transition between navigation and being part of conversational groups, which is not easily achievable with the random or the handcrafted pedestrian behaviors in SEAN 2.0.

The location of graph annotation nodes and the parameterization of accompanying edges can be used to modify pedestrian congestion in the environment. Depending on the graph's structure, certain areas may have more or less pedestrian congestion than what a user desires. Pedestrians can be directed away from the congested area by using edge weights associated with low or uni-directional flow.

Implementation. SEAN 2.0 provides one Behavior Graph annotation for each environment (warehouse, lab, and outdoor). Each Behavior Graph annotation consists of a graph where every pair of adjacent nodes is connected by two directed edges. Edges are weighted to control pedestrian congestion using one of three costs: c_{min}, 1, or c_{max}, where $1 < c_{max}$ and $0 < c_{min} < 1$. For example, consider the edges between nodes u and v. Users of SEAN 2.0 can constrain pedestrian flow using 4 sets of edge weights $\langle c_{uv}, c_{vu} \rangle$: $\langle 1, 1 \rangle$ for there to be medium flow between the nodes, $\langle c_{min}, c_{min} \rangle$ for high flow, $\langle c_{max}, c_{max} \rangle$ for low flow, and either $\langle 1, c_{max} \rangle$ or $\langle c_{max}, 1 \rangle$ for uni-directional flow.

The path to a pedestrian's goal node is computed over the edges between waypoint nodes using Dijkstra's algorithm [47], which considers the different edge costs and ensures that pedestrians do not disrupt groups. Pedestrians traverse the computed path using the Social Forces model [28], which allows them to perform local collision avoidance.

E. Robot Tasks

Tasks specify the robot's start (A) and goal (B) poses. Robot tasks can leverage ground truth information from the simulation, like group locations, to choose robot poses that may result in specific social situations.

SEAN 2.0 implements the following robot tasks:

- *RandomABNav*: uniformly samples a start and a goal pose for the robot from the navigable plane in the environment.
- *BusyABNav*: samples a start and a goal pose for the robot nearby the largest cluster of pedestrians. Pedestrian poses from SEAN 2.0 are clustered via k-means.
- *Join Group*: a group center is sampled from graph nodes associated with group formations. Then, a point in the group is sampled as the goal location for the robot and a further away point is sampled as its start pose.
- *Leave Group*: a group center is sampled from graph nodes associated with group formations. Then, a point in the group is sampled for the start location for the robot and a further away point is sampled for the goal pose.
- *Handcrafted*: assigns a start and goal pose specifically chosen by a scenario designer. We implemented 5 handcrafted tasks corresponding to the Cross Path, Down Path, Join Group, Leave Group and Empty social situations.

Handcrafted tasks can only be used with handcrafted scenarios as both robot and pedestrian poses are chosen by the scenario designer. Join and Leave Group tasks can only be used with the Behavior Graph method of pedestrian control as they depend on nodes in the graph. All other tasks are decoupled from the method of pedestrian control.

Tasks can be designed such that the robot is likely to experience a certain social situation. However, the social situations that we consider in this work only occur upon satisfaction of the propositions described in Section III. For example, a robot aiming to complete a Join Group task is not guaranteed to enter a Join Group social situation, but given the task design it is likely to experience this situation.

F. Sensor Integration

SEAN 2.0 provides a simulated RGB camera and a simulated depth sensor. By convention, each robot implementation includes one simulated depth sensor and three RGB cameras. The depth sensor is positioned in a first-person perspective and the RGB cameras provide three angles: first-person, third-person, and top-down. This default configuration ensures that a standard API is available when accessing sensor data for any robot, allowing for comparison across robots.

G. Social Situation Classifiers

As part of SEAN 2.0, we provide five rule-based classifiers that implement the propositional predicates defining our five social situations from Section III. The predicates that we define use parameters derived from Hall's work in proxemics [48] where applicable. For example, we consider two agents to be "nearby" when the distance between them is less than two times their personal space (1.2m). Our experiments in Section V indicate that social situation classifiers can help users better

understand the distribution of data resulting from different pedestrian control methods. They can also help users identify how well a robot navigation policy can learn from and perform in different social situations.

H. Metrics

We implement a range of social navigation metrics which are aggregated over the duration of a Robot Task, including:

Path Efficiency: ratio between the traveled and geodesic distance of the search-based path from the starting position.
Time Not Moving: seconds that the robot was not moving.
Intimate Distance Violation: number of times the robot approached a pedestrian within a distance of 0.45m.

Our documentation details all of the other metrics implemented in our system.[1] The underlying data needed to compute these metrics is available from the SEAN 2.0 API.

V. System Evaluation

We first studied the usefulness of SEAN 2.0 for the robotics community by collecting and incorporating feedback from seven researchers who used our system. Then, we studied how robot navigation datasets generated via SEAN 2.0 affected the training and evaluation of social navigation algorithms in environments with varying pedestrian behavior.

A. User Feedback About SEAN 2.0

We initially gathered feedback about SEAN 2.0 from four robotics researchers at Yale University, University of Washington, University of Massachusetts Amherst, and Carnegie Mellon University who were previously unfamiliar with the present work.[2] They installed and used SEAN 2.0 and then provided written feedback. Although one researcher noted that occasionally pedestrians exhibited "unnatural behavior like running into each other, the robot, and obstacles," in general the feedback was broadly positive. For example, one person said that the "lab scene and the warehouse scene both looked good." Another researcher familiar with the dynamics of the Kuri robot – one of the robots included in SEAN 2.0 – noted that the base dynamics were realistic and Kuri followed the navigation path "in a way that looks much like the real platform" when controlled by the ROS Navigation Stack. The one researcher who had prior experience with SEAN 1.0 noted that SEAN 2.0 was "definitely easier to navigate and more user friendly in terms of getting started with the simulator." This researcher later informed us via personal communication that they were able to successfully setup and use SEAN 2.0 to submit a paper for publication without our involvement. In contrast, when the user was using SEAN 1.0, we made many small changes to the SEAN 1.0 system to support their workflow.

Based on the helpful feedback from our users, we made a number of improvements to SEAN 2.0. First, we tuned the parameters of our social forces model to decrease pedestrian-on-pedestrian collisions. Second, we improved the documentation for system setup and the integration of new robot policies. Third, we fixed several bugs, including adding localization information for two robot components and resolving an edge case where the robot's physics simulation was unrealistic when it collided with a pedestrian.

After incorporating this initial feedback, we shared the system with three more researchers at Yale University. One said that the system was "easy to use," another mentioned that they did not run into any issues while using the system, and another noted that "I can see how one would be able to implement their own controller through this [system]."

B. Emergence of Social Situations

We studied the distributions of social situations in three datasets collected from SEAN 2.0, where each dataset corresponded to a different method of pedestrian control.

1) Data Collection: Sensor data and robot control information was collected in SEAN 2.0 while a human expert navigated the robot in the warehouse environment using a joystick. The expert was first familiarized with the analog joystick controls and then directed to navigate the robot to the goal in a polite manner, similar to the way they would navigate in real life. One hour of data was collected for each of the three proposed methods of pedestrian control: random, handcrafted, and graph-based. The random and graph methods were configured with an equal number of pedestrians ($n = 62$). The handcrafted scenarios had a variable number depending the scenario's design, but on average they had far fewer pedestrians ($n \approx [5, 10]$) due to the time and effort required to manually specify behaviors.

The social situation classifiers described in Section IV-G were parameterized as follows. The maximum distance at which two agents were considered nearby was $\mathsf{D} = 2.4$m, within Hall's personal space [48]. The minimum speed of an agent considered to be walking was $\mathsf{V} = 1.4$m/s, near the average human walking speed. We set a small error value $\mathsf{A} = \frac{\pi}{12} = 15^\circ$ within which a pedestrian trajectory was considered parallel or perpendicular to the robot.

During data collection, robot tasks were chosen to induce the robot to experience a uniform distribution of social situations depending on the method of pedestrian control. For the random method, we used the RandomABNav task for the entire hour of data collection. For the handcrafted method, we collected 12 min. of data from each of the 5 robot tasks designed to correspond to the 5 social situations formalized in Section III. For the graph, we collected data evenly between the Join Group and Leave Group robot tasks.

The collected data was divided into examples composed of 5 depth images, a local navigation plan with 5 points of

TABLE I
Percentage of examples belonging to each social situation.

	Social Situation				
Behavior	Cross Path	Down Path	Join Group	Leave Group	Empty
Random	14.51%	24.68%	0%	0%	66.44%
Handcrafted	0.73%	0.74%	3.43%	13.48%	81.62%
Graph	13.24%	22.08%	12.09%	19.51%	33.08%

[1] https://sean.interactive-machines.com/docs/metrics
[2] Our protocol was reviewed by our IRB and exempt from annual review.

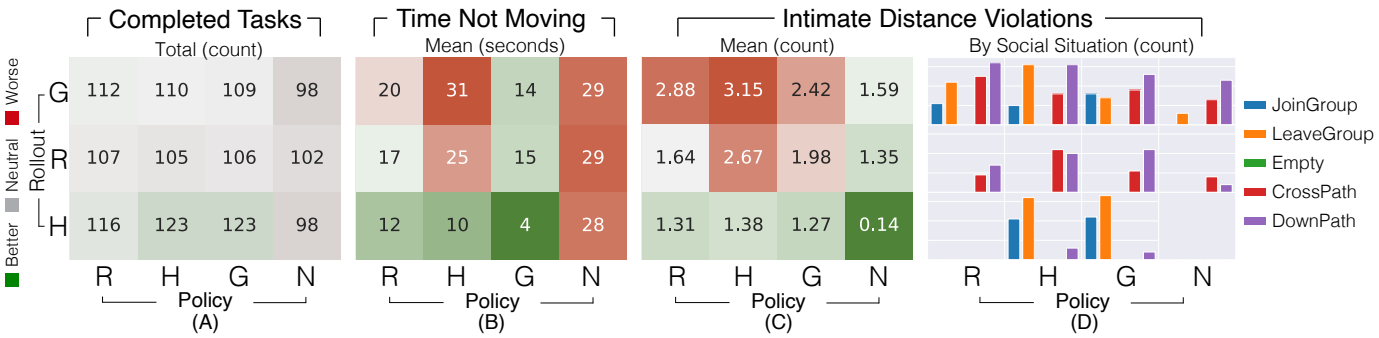

Fig. 4. Select results for 4 policies in 3 scenarios aggregated over 123 episodes. The three learned policies are trained on data from the Behavior Graph (G), Random (R), and Handcrafted (H) methods of pedestrian control. We also evaluate the ROS Navigation (N) policy with social cost layers [49]. Three metrics are shown: Completed Tasks (A), Time Not Moving (B), and Intimate Distance Violations (C and D). Values for Time Not Moving are computed in seconds and averaged over all episodes. All other plots show the total or average count of the occurrences of the metric over all episodes. The percentage gain for a specific metric can be calculated between grid-cells. For example, the Behavior Graph policy spends 55% less Time Not Moving than the Handcrafted policy when rolled out in the Behavior Graph environment.

the expert's most recent trajectory sampled at 1hz, a search-based global plan with the 10 nearest points between the robot position and the goal (where each point was at least 0.5m apart). Each example also contained five boolean flags corresponding to the social situation classifiers.

2) Results: Table V-B1 shows the distribution of social situations between the three datasets generated using SEAN 2.0 with different methods of pedestrian control. The number of examples in which the robot experienced the five social situation were not evenly distributed with the Handcrafted Behavior approach, even though we collected data for an equal amount of time in each Handcrafted scenario. In the Random pedestrian dataset, groups did not form, so no Join Group or Leave Group social situations were experienced. However, social situations were more evenly distributed among the Cross Path, Down Path, and Empty scenarios in comparison to data from the Handcrafted scenarios. The Behavior Graph dataset led to a more uniform distribution of social situations than the Handcrafted or Random datasets.

C. Evaluation of Navigation Algorithms

1) Experimental Setup: Whereas SEAN 1.0 [12] was evaluated using only the ROS navigation stack, we evaluated SEAN 2.0 as a benchmarking platform using two robot navigation methods: the ROS navigation stack with social cost layers [49], and a neural network controller following [50] (using depth images rather than LIDAR as input). For the latter method, we trained three controllers using each of the datasets created with the three different methods of pedestrian control outlined in Section V-B1. This was not possible with SEAN 1.0 because SEAN 1.0 only supported one method of pedestrian control.

We trained the neural network controllers end-to-end using PyTorch with equally weighted losses for the local planner and velocity controller modules. We used the AdamW optimizer with the default parameters, lr=0.001, wd=0.010, and a batch size of 1024 for all experiments. A search over a range of batch sizes ($\{128, 256, 512, 1024\}$) revealed similar performance so we chose a batch size which used the maximum amount of GPU memory, effectively decreasing the time required to train over a single epoch of the data. The training machine had 128GB of RAM, an Intel Xeon W-2155 CPU clocked at 3.30GHz, and an NVIDIA RTX 2080TI GPU. An early stopping window of 50 epochs was used after trying between 10 and 100 epochs. The loss did not always reach a local minima at 10 epochs, but the loss stabilized far before 100.

2) Results: Neural network navigation policies trained on the Handcrafted (H) dataset completed a similar number of episodes as policies trained on the Behavior Graph (G) data. Across all rollout environments, the average number of Completed Tasks was 112.7 for both H and G, as computed over the columns of Fig. 4A. However, polices trained on Handcrafted (H) data spent more time on average not moving than the policies trained using the Behavior Graph (G) method. The policy trained using Random (R) pedestrians paused for less time on average than the policy trained on Handcrafted (H) data, but paused for more time on average than the policy trained using the Behavior Graph (G). All learned policies spent less time not moving than the ROS Navigation Stack (N). The average Time Not Moving per policy was 16.3s for R, 22s for H, 11s for G, and 28.7s for N, as computed over the columns of Fig. 4B.

Intimate Distance Violations were more numerous for policies rolled out in the scenarios with the Behavior Graph (G) compared to the scenarios utilizing Random (R) and Handcrafted (H) pedestrians. The average number of Intimate Distance Violations per rollout scenario were 2.5 for G, 1.9 for R, 1.0 for H, as computed over the rows of Fig. 4C. Dissecting the data by social situations in Fig. 4D, we see the increase in violations in the Behavior Graph rollout scenario occurred mainly in the Cross Path and Down Path social situations. This type of analysis suggests that performing data augmentation for a learned policy and adjusting the training data distribution to include more samples from the under-performing social situation could increase performance. Additionally, dissecting metrics by social situation allows researchers to interrogate controller performance in specific contexts. For instance, delivery robots may need to be especially skilled at navigating down busy pathways in warehouse settings. Note that splitting the data by social situation was not possible in SEAN 1.0 as it did not contain a concept of social situations.

VI. Conclusion and Future Work

SEAN 2.0 enables the development of socially compliant navigation controllers in densely populated, dynamic environments. We designed SEAN 2.0 based on our desire to train and evaluate navigation algorithms that could adapt to varied social contexts. This led to the proposed formalization of social situations and a variety of features in SEAN 2.0, including sensors, navigation metrics, different means of specifying robot tasks, a set of logic-based classifiers for social situations, and multiple methods for pedestrian control. In particular, the newly proposed Behavior Graph enabled efficient specification of pedestrian motion while allowing for the emergence of varied social situations.

While our work showed how five social situations could be formalized, these are not intended to be a complete set. In the future, we hope to expand our work to include other social situation classifiers in SEAN 2.0. Moreover, the notion of social situations for navigation could be expanded to include other factors such as urgency and the configuration of objects in the environment. Also, we hope to perform a larger effort benchmarking different navigation policies.

Acknowledgements

We thank Dylan Glas, Adam Fineberg, and the early users of SEAN 2.0 for their helpful feedback.

References

[1] C. Mavrogiannis, F. Baldini, A. Wang, *et al.*, "Core challenges of social robot navigation: A survey," *arXiv:2103.05668*, 2021.
[2] Y. Gao and C.-M. Huang, "Evaluation of socially-aware robot navigation," *Front. Robot. AI*, 2022.
[3] C. Johnson and B. Kuipers, "Socially-aware navigation using topological maps and social norm learning," in *AIES*, 2018.
[4] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," in *IROS*, 2017.
[5] S. Satake, T. Kanda, D. F. Glas, M. Imai, H. Ishiguro, and N. Hagita, "How to approach humans? strategies for social robots to initiate interaction," in *HRI*, 2009.
[6] A. Favier, P.-T. Singamaneni, and R. Alami, "Simulating intelligent human agents for intricate social robot navigation," in *RSS Workshop on Social Robot Navigation*, 2021.
[7] X.-T. Truong and T.-D. Ngo, "To approach humans?: A unified framework for approaching pose prediction and socially aware robot navigation," *TCDS*, 2017.
[8] F. Yang and C. Peters, "Appgan: Generative adversarial networks for generating robot approach behaviors into small groups of people," in *RO-MAN*, 2019.
[9] S. Thrun, M. Bennewitz, W. Burgard, *et al.*, "Minerva: A tour-guide robot that learns," in *AAAI*, 1999.
[10] P. Trautman and K. Patel, "Real time crowd navigation from first principles of probability theory," in *ICAPS*, 2020.
[11] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *CoRL*, 2017.
[12] N. Tsoi, M. Hussein, J. Espinoza, X. Ruiz, and M. Vázquez, "Sean: Social environment for autonomous navigation," in *HAI*, 2020.
[13] C. Li, F. Xia, R. Martın-Martın, *et al.*, "Igibson 2.0: Object-centric simulation for robot learning of everyday household tasks," in *CoRL*, 2021.
[14] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, *et al.*, "Habitat: A Platform for Embodied AI Research," in *ICCV*, 2019.
[15] G. Echeverria, N. Lassabe, A. Degroote, and S. Lemaignan, "Modular open robots simulation engine: Morse," in *ICRA*, 2011.
[16] Y. Kaneshige, S. Satake, T. Kanda, and M. Imai, "How to overcome the difficulties in programming and debugging mobile social robots?" In *HRI*, 2021.
[17] A. Favier, P. Singamaneni, and R. Alami, "An intelligent human simulation (inhus) for developing and experimenting human-aware and interactive robot abilities," 2021.
[18] M. Campanella, S. Hoogendoorn, and W. Daamen, "The nomad model: Theory, developments and applications," *Tra. Res. Pro.*, 2014.
[19] C. Gloor, *Pedsim: Pedestrian crowd simulation*, silmaril.org, 2016.
[20] S. Curtis, A. Best, and D. Manocha, "Menge: A modular framework for simulating crowd movement," *Collective Dynamics*, 2016.
[21] A. Biswas, A. Wang, G. Silvera, A. Steinfeld, and H. Admoni, "Socnavbench: A grounded simulation testing framework for evaluating social navigation," *THRI*, 2021.
[22] M. Kapadia, N. Pelechano, J. Allbeck, and N. Badler, "Virtual crowds: Steps toward behavioral realism," *Syn. lec. com. vis.*, 2015.
[23] J. M. Wiener, S. J. Büchner, and C. Hölscher, "Taxonomy of human wayfinding tasks: A knowledge-based approach," *Spat. Cogn. Comput.*, 2009.
[24] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, "Human motion trajectory prediction: A survey," *Int J Rob Res*, 2020.
[25] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *SIGGRAPH*, 1987.
[26] N. Pelechano, J. M. Allbeck, and N. I. Badler, "Controlling individual agents in high-density crowd simulation," in *SCA*, 2007.
[27] C. Peters and C. Ennis, "Modeling groups of plausible virtual pedestrians," *CG&A*, 2009.
[28] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, 1995.
[29] J. Van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *ICRA*, 2008.
[30] A. Vinciarelli, M. Pantic, and H. Bourlard, "Social signal processing: Survey of an emerging domain," *Image Vis. Comput.*, 2009.
[31] A. Nigam and L. D. Riek, "Social context perception for mobile robots," in *IROS*, 2015.
[32] P. Althaus, H. Ishiguro, T. Kanda, T. Miyashita, and H. I. Christensen, "Navigation for human-robot interaction tasks," in *ICRA*, 2004.
[33] G. Castellano, I. Leite, A. Pereira, C. Martinho, A. Paiva, and P. W. McOwan, "Detecting engagement in hri: An exploration of social and task-based context," in *SOCIALCOM-PASSAT*, 2012.
[34] M. Argyle, A. Furnham, and J. A. Graham, *Social situations*. Cambridge University Press, 1981.
[35] C. Pérez-D'Arpino, C. Liu, P. Goebel, R. Martın-Martın, and S. Savarese, "Robot navigation in constrained pedestrian environments using reinforcement learning," *arXiv:2010.08600*, 2020.
[36] M. Swofford, J. Peruzzi, N. Tsoi, *et al.*, "Improving social awareness through dante: Deep affinity network for clustering conversational interactants," *PACMHCI*, 2020.
[37] S. Thompson, A. Gupta, A. W. Gupta, A. Chen, and M. Vázquez, "Conversational group detection with graph neural networks," in *ICMI*, 2021.
[38] A. Kendon, *Conducting interaction: Patterns of behavior in focused encounters*. CUP Archive, 1990, vol. 7.
[39] M. Quigley, K. Conley, B. Gerkey, *et al.*, "Ros: An open-source robot operating system," in *ICRA*, 2009.
[40] N. Tsoi, M. Hussein, O. Fugikawa, J. D. Zhao, and M. Vázquez, "An approach to deploy interactive robotic simulators on the web for hri experiments: Results in social robot navigation," in *IROS*, 2021.
[41] L. Bass, B. E. John, and J. Kates, "Achieving usability through software architecture," Tech. Rep., 2001.
[42] K. Stencel and P. Wegrzynowicz, "Implementation variants of the singleton design pattern," in *OTM*, 2008.
[43] N. Chen, *Convention over configuration*, vazexqi.com, 2006.
[44] S. Legros and B. Cislaghi, "Mapping the social-norms literature: An overview of reviews," *Perspect. Psychol. Sci.*, 2020.
[45] M. L. Walters, K. Dautenhahn, R. Te Boekhorst, *et al.*, "The influence of subjects' personality traits on personal spatial zones in a human-robot interaction experiment," in *ROMAN*, 2005.
[46] G. Zen, B. Lepri, E. Ricci, and O. Lanz, "Space speaks: Towards socially and personality aware visual surveillance," in *MPVA*, 2010.
[47] E. W. Dijkstra *et al.*, "A note on two problems in connexion with graphs," *Numerische mathematik*, 1959.
[48] E. T. Hall, *The hidden dimension*. 1966.
[49] D. V. Lu, D. Hershberger, and W. D. Smart, "Layered costmaps for context-sensitive navigation," in *IROS*, 2014.
[50] A. Pokle, R. Martın-Martın, P. Goebel, *et al.*, "Deep local trajectory replanning and control for robot navigation," in *ICRA*, 2019.