Socially-Aware Navigation: A Non-linear Multi-Objective Optimization Approach

SANTOSH BALAJEE BANISETTY, Dept. of Computer Science and Engineering, University of Nevada Reno
SCOTT FORER, Dept. of Mechanical Engineering, University of Nevada Reno
LOGAN YLINIEMI*, Dept. of Mechanical Engineering, University of Nevada Reno
MONICA NICOLESCU, Dept. of Computer Science and Engineering, University of Nevada Reno
DAVID FEIL-SEIFER, Dept. of Computer Science and Engineering, University of Nevada Reno

Mobile robots are increasingly populating homes, hospitals, shopping malls, factory floors, and other human environments. Human society has social norms that humans mutually accept, obeying these norms is an important signal that someone is participating socially with respect to the rest of the population. For robots to be socially compatible with humans, it is important for robots to obey these social norms. In prior work, we demonstrated a Socially-Aware Navigation (SAN) planner, based on Pareto Concavity Elimination Transformation (PaCcET), in a hallway scenario optimizing two objectives so that the robot does not invade the personal space of people. In this paper, we extend our PaCcET based SAN planner to multiple scenarios with more than two objectives. We modified the Robot Operating System’s (ROS) navigation stack to include PaCcET in the local planning task. We show that our approach can accommodate multiple Human-Robot Interaction (HRI) scenarios. Using the proposed approach, we were able to achieve successful HRI in multiple scenarios like hallway interactions, an art gallery, waiting in a queue and interacting with a group. We implemented our method on a simulated PR2 robot in a 2D simulator (Stage) to validate all the scenarios. A comprehensive set of experiments show that our approach can handle multiple interaction scenarios, hence can be a viable option for a Unified Socially-Aware Navigation (USAN).

CCS Concepts:
• Mathematics of computing → Mathematical optimization;
• Human-centered computing; • Computer systems organization → Robotics;

Additional Key Words and Phrases: human-robot interaction, socially-assistive robotics, socially-aware navigation

ACM Reference Format:

*L. Yliniemi is currently with Amazon Robotics. This work was completed prior to joining Amazon.

Authors’ addresses: Santosh Balajee Banisetty, santoshbanisetty@nevada.unr.edu, Dept. of Computer Science and Engineering, University of Nevada Reno, 1664 N. Virginia Street, Reno, Nevada, 89557; Scott Forer, sforer@nevada.unr.edu, Dept. of Mechanical Engineering, University of Nevada Reno, 1664 N. Virginia Street, Reno, Nevada, 89557; Logan Yliniemi, logan@unr.edu, Dept. of Mechanical Engineering, University of Nevada Reno, 1664 N. Virginia Street, Reno, Nevada, 89557; Monica Nicolescu, monica@unr.edu, Dept. of Computer Science and Engineering, University of Nevada Reno, 1664 N. Virginia Street, Reno, Nevada, 89557; David Feil-Seifer, dave@cse.unr.edu, Dept. of Computer Science and Engineering, University of Nevada Reno, 1664 N. Virginia Street, Reno, Nevada, 89557.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.

XXXX-XXXX/2019/7-ART $15.00
https://doi.org/10.1145/1122445.1122456
1 INTRODUCTION

Social norms such as driving on the right or left side of the road (depending on the country one lives in), turn-taking rules at four-way stops and roundabouts, holding doors for people behind us, and maintaining an appropriate distance when interacting with another person (actual distance depending on the type of interaction) are crucial in our day-to-day interactions. People use these actions as signals that they are participants in a social order, violating these principles is jarring at best (i.e., a person becoming confused at another person’s behavior), and can provoke hostility at worst (i.e., getting upset at someone for cutting in line).

As socially assistive robots (SAR) [16] are becoming an integral part of our society, these robots taking roles while remaining unconstrained to such social norms is a growing concern in the robotics community. In the last decade, social and personal robots have attained immense interest among researchers and entrepreneurs alike; the result is an increase in the efforts both in industry and academia to develop applications and businesses in the personal robotics space. Smart Luggage [14], showcased at 2018’s consumer electronics show (CES), a robotic suitcase that will follow the owner during travel demonstrates the commercial viability of interpersonal navigation. Some companies and start-ups develop robotic assistants for airports and shopping malls to assist people with directions and shopping experience. Robot domains, especially SAR, benefit from navigation as such movement extends the reachable service area of the robot. However, navigation if not appropriately performed can cause an adverse social reaction [34]. In an ethnographic study where nurses in hospital settings interacted with an autonomous service robot, one of the participants quoted the following statement:

“Well; it almost ran me over... I wasn’t scared... I was just mad... I’ve already been clipped by it. It does hurt.”

Robots currently deployed in human environments have prompted adverse reactions from people encountering them [23]. Some people are not willing to interact with these robots, and some of them even kicked them, demonstrating a very hostile attitude towards service robots. Incidents such as these pose both challenges and opportunities for the HRI researchers, as a result, recent years have seen a tremendous growth of publication in areas related to socially-aware navigation such as human detection and tracking, social planner incorporating social norms, human-robot interaction studies to understand the problem from a human’s perspective.

As seen from the examples so far [23, 34], SAR systems without social-awareness can cause problems in human environments. Following the social norms posed by human society may help robots be more accepted in homes, hospitals and workplaces. Robots by means of their navigation behavior should not treat humans as dynamic objects, as shown in our results in Figure 1. The figure illustrates a comparison between a traditional planner (green trajectory) that optimizes for performance (time, distance, etc.) and a socially-aware planner (blue trajectory) which optimizes for social norms along with performance metrics. The traditional planner treats both human and an object the same way (not maintaining enough distance around human and object), however, that is not acceptable. It is acceptable to get close to an object, but a similar behavior around a person is not acceptable.

Socially-Aware Navigation planners, including our method, considers the theory of proxemics and other social norms such that the robot does not invade the space around the human. Proxemics [21] codifies this notion of personal space; researchers interested in SAN are investigating methods to integrate the rules of proxemics into robot navigation behavior.

Our approach to SAN using PaCcET addresses the following limitations of the existing approaches:
Fig. 1. The PaCcET local planner (blue) compared with the traditional ROS local planner (green), which does not account for social norms. The traditional planner generated a trajectory that is close to both human and the object (black box), treating them alike. Our approach, the PaCcET-based SAN planner, generated a trajectory that diverged around the human, thereby respecting the personal space of the human.

(1) Many current approaches depend on exocentric sensing, hence, limiting the robot’s services to a particular environment [15].
(2) Approaches may require a large amount of training data [1, 22, 26, 28, 36].
(3) The environment/scenario is a singleton, i.e., Only a hallway, a room, etc., is considered, or considers only an approach behavior or a passing behavior [15].
(4) Planners are optimized for single or few objectives with linear combination or weighted sum [17, 18].

This paper builds on our previous work from [20] and provides a comprehensive evaluation of our SAN planner in multiple scenarios in a simulation. We extend our prior work to include more objectives which include inter-personal distance, adherence to a social goal and group proxemics as well as providing a more diverse set of simulated environments for evaluation. The remainder of this paper is structured as follows. In the next section, we review related works. In Section 3, we present our approach to SAN using PaCcET. In Section 4, we apply our method to various scenarios in simulation to validate the proposed approach. Next, in Section 5, we discuss our present and on-going work.

2 RELATED WORK

2.1 Socially-Aware Navigation

When localization and navigation were relatively new in the field of robotics, Rhino and Minerva [8, 9, 42], tour guide robots, successfully navigated and gave museum tours for visitors. These robots were some of the pioneering works in the field of robot navigation which performed various navigation functions like mapping, localization, collision avoidance, and path planning. These robots also exhibited primitive navigation behaviors around people in a dynamic human environment. Robust long-term navigation in the indoor environment was made possible by Marder et al. [30] when a PR2, a mobile manipulation robot completed a 26.2 mile run in an office environment. This state of the art navigation technique (traditional planner) can generate a collision-free path and maneuver a robot on that path to get to a goal. However, these algorithms are not sophisticated enough to deal with social interactions that occur while navigating in highly dynamic human environments.

There is a rapidly growing HRI community that is addressing social navigation planners. Most of the SAN research can be broadly classified (based on areas concerning social navigation) into
Planning, Perception, and Behavior Selection. Most of the work is concentrated in the planning area (in turn classified into local planning and global planning). The solutions to SAN associated problems range from simple cost functions to more advanced deep neural networks; we present some of them here.

Social Force Model (SFM) [24] is one of the popular approaches to SAN which mimics human navigation behavior. Building upon prior work, Ferrer et al. programmed a robot to obey the social forces during navigation activities [17, 18]. The method also extends SFM to allow a robot to accompany a human while providing a method for learning the model parameters. Kivrak et al. [27] also adopted SFM to be used as a local planner to generate a socially aware trajectory in a hallway scenario. Silva et al. [40] took a shared effort approach to solve the human-robot collision avoidance problem using Reinforcement Learning. Simulated results show that the approach enabled the agents (human and robot) to avoid a collision mutually. Our proposed approach is validated not in a single context but multiple contexts like hallway interactions, joining a group of people, waiting in a line, and an art gallery interaction.

Dondrup et al. [13] proposed a combination of well-known sample-based planning and velocity costmaps to achieve socially-aware navigation. The authors used a Bayesian temporal model to represent navigation intent of robot and human based on Qualitative Trajectory Calculus and used these descriptors as constraints for trajectory generation. Alonso et al. [2] presented a \( \epsilon \)-cooperative collision avoidance method in dynamic environments among interactive agents (robots or humans). A variation of the Turing test was administered which determines whether participants can differentiate between human motions and artificially generated motions. Aroor et al. [3] formulated a Bayesian approach to develop an online global crowd model using a laser scanner. The model uses two new algorithms, CUSUM-A* (to track the spatiotemporal changes) and Risk-A* (to adjust for navigation cost due to interactions with humans), that rely on local observation to continuously update the crowd model. Unlike other model-based approaches, our method does not require any training data to perform socially-aware navigation.

Okal et al. [36] presented a Bayesian Inverse Reinforcement Learning (BIRL) based approach to achieving socially normative robot navigation using expert demonstrations. The method extends BIRL to include a flexible graph-based representation to capture the relevant task structure that relies on collections of sampled trajectories. Kim et al. [26] presented an Inverse Reinforcement Learning (IRL) based framework for socially adaptive path planning to generate human-like trajectories in dynamic human environments. The framework consists of three modules: a feature extractor, learning module, and a path planning module. Kretzschmar et al. [28] proposed a method, to learn policies from demonstrations, to learn the model parameters of cooperative human navigation behavior that match the observed behavior concerning user-defined features. They used Hamiltonian Markov chain Monte Carlo sampling to compute the feature expectations. To adequately explore the space of trajectories, the method relied on the Voronoi graph of the environment from start to target position of the robot. Our proposed method performs Pareto concavity elimination thereby considers non-linear solution space.

Human motion prediction is of vital importance in SAN as it allows the robot to plan and execute its motion behaviors according to the predicted human motion. In contrast to traditional human trajectory prediction approaches which use hand-crafted functions (social forces), Alahi et al. [1] proposed an LSTM (Long Short-Term Memory) model which can observe general human motion and predict their future trajectories. Hamandi et al. [22] developed a novel approach using
deep learning (LSTM) called DeepMoTIon, trained over public pedestrian surveillance data to predict human velocities. The DeepMoTIon method used a trained model to achieve human-aware navigation, where the robots imitate humans to navigate in crowded environments safely.

Although SAN research is dominated by planning related advancements, for a long-term HRI in human environments, we need to understand how cooperative human navigation works. Psychologists and roboticists are looking into social cues, and their effects on HRI to better understand the socially-aware navigation problem. Suvei et al. [41] investigated the problem of “how a robot can get closer to people that it wants to interact with?” In a 2x2 between-subject study, the authors investigated the effect of social gaze cue on the personal space invasion using a human-sized mobile robot. The results from a 2 x 2 between-subjects experiment, with/without personal space invasion, and with/without a social gaze cue, indicate that social gaze did play a role in participants’ perceived safety of the robot.

2.2 Multi-Objective Optimization

It is easy to think of a task as a single objective function, where there is a goal or cost function that we are trying to either minimize or maximize. Ideally, this would always give the optimal solution for a task; however, this is not always the case. More often than not, multiple variables contribute to a cost function. An example of this is from basic economics, where there exists a market for a widget. As the supply of this widget goes up, the demand decreases and vice-versa. This phenomenon would be known as a supply and demand curve where one objective is the supply, and the other is the demand. In this case, the seller would want to find the optimal amount of supply such that demand provides the optimal amount of profit. If graphed, this supply and demand curve, as shown in Figure 2, the points on the line would be Pareto optimal points; no point dominates any other. In this case, the seller is trying to maximize both objectives; therefore the hollow circles are the dominated points as there exist solid circle points that are better in both objectives. Typically, multiple Pareto optimal points are forming a set which is the solution type that many multi-objective algorithms use [11].

![Figure 2: Multi-Objective solution space](image)

Multi-objective optimization has already started to play a role in real-world applications [31]. Some examples of real-world multi-objective scenarios are high speed civilian aircraft transportation [32], urban planning [4], and for designing trusses [10]. The trajectory planner used in this paper builds off of a pre-existing one that utilizes a multi-objective approach along with linear combination [30].
2.2.1 PaCcET. In some cases, optimizing a single objective does not yield the desired performance and therefore multiple objectives need to be considered when evaluating a policy’s fitness. A standard method is to multiply a preset scalar value to each objective’s fitness score and then add them all together. In some domains this can lead to an optimal set of policies; however, in some complicated domains, this method will yield suboptimal policies. A solution to this is to use a multi-objective tool, such as PaCcET, to evaluate policies on multiple objectives [45, 46] properly. PaCcET works by first obtaining an understanding of the solution space and finding the Pareto optimal solutions. Next PaCcET transforms the solution space and then compares each solution giving a single fitness score representative of how well each solution performed in the transformed space.

At a high level, PaCcET works by transforming the Pareto front in the objective space in a way that it is forced to be convex. Transforming to objective space allows the linear combination of transformed objectives to find a new Pareto optimal point. PaCcET iteratively updates this transformation to always force non-explored areas of the Pareto front to be more highly valued than points dominated by the Pareto front or points that are on the explored areas of the Pareto front.

PaCcET has seen a variety of applications: it has been used to extend the life of a fuel cell in a hybrid turbine-fuel cell power generation system [12], the operation of the electrical grid on naval vessels [37], the coordination of multi-robot systems [44], and for the efficient operation of a distributed electrical microgrid [38], where a series of small power generation systems coordinate to meet the demands of consumers. In each of these applications, it has been shown that PaCcET functions at or above the solution quality of other techniques like NSGA-II or SPEA2 [45], with as low as one-tenth of the run-time.

For this project, PaCcET was used over other multi-objective tools because of its computational speed [45] as shown in Figure 3. PaCcET was used to evaluate the possible trajectories developed in the local planner. At each time step, the sensor data is analyzed, and the desired features are evaluated for each of the potential future trajectory points. PaCcET then uses the fitness values for each feature of every future trajectory to develop the solution space and obtain the optimal future trajectory. Since at each time step, future trajectories are developed independently, PaCcET develops a brand new solution space at each time step. By developing a new solution space at each time step, the local planner can be optimized in real time.

![PaCcET computational speed](image)

**Fig. 3. PaCcET computational speed** - Percentage of hypervolume dominated in Kursawe’s (KUR) problem in comparison with two successful multi-objective methods, SPEA2 and NSGA-II. This plot that PaCcET proceeds faster towards the Pareto front.
3 METHOD

In this section, we detail our methodology of a socially-aware navigation planner, the features or objectives that we used to optimize the trajectories, and how PaCcET was implemented in the local trajectory selection process. Figure 4 shows the overall high-level block diagram of the proposed approach. It is built on top of the well established ROS navigation stack by modifying the local planner using PaCcET. The overall function of the local trajectory planner at each time step is to generate an array of possible future trajectory points and evaluate each future trajectory point based on a predefined feature set as shown in Figure 5. In previous work, the features were assumed to have either no relationship or a simple linear relationship with one another; however, this is not always the case, and therefore we need to consider the possibility that the features are not only dependent on each other but also have nonlinear relationships.

Fig. 4. Block diagram of proposed approach, a modification of ROS navigation stack’s local planner using PaCcET based non-linear optimization.

Fig. 5. **Navigation Planner** - The navigation planner selects a short-term trajectory (green points represent potential trajectory end-points) from the pool of possible trajectories (black points), optimized for adherence to a long-term plan (blue line), obstacle avoidance, and progress toward a goal, and in the case of this paper, interpersonal distance.

3.1 Features/Objectives

In the traditional navigation planner, the features extracted were each assigned their own cost (e.g., the path distance cost ($\Delta_{path}$), the length that the robot has already traveled, the goal distance...
cost \((\Delta_{\text{goal}})\), the distance the robot is from the goal) [30]. The path distance will have a linear relationship with the goal distance since the change in one has a direct linear impact on the other. Once each feature has a cost associated with it, each cost is multiplied by a pre-tuned scalar and then added together, thus giving a linear combination, or weighted sum, in this case, the cost function shown in Equation 1. We can think of this cost function as an objective, where each possible future trajectory point has a cost or fitness associated with that objective. Since the purpose is to minimize the overall cost function, the planner will take the best path possible that minimizes the function, which in this case will minimize both features.

\[
\text{cost}(v_x, v_y, v_\theta) = \alpha(\Delta_{\text{path}}) + \beta(\Delta_{\text{goal}})
\]  

(1)

More recently, this cost function has been adapted to include a heading difference \((\Delta_{\text{heading}})\) feature and an occupancy \((\Delta_{\text{occ}})\) cost feature, where the heading difference is the distance that the robot is from the global path and the occupancy cost is the cost used to keep the robot from hitting something. The same approach as in the previous cost function is taken in Equation 2. By taking a closer look at just the heading difference and how that might affect the path distance or the goal distance it becomes less clear if there is only a linear relationship between the four. For example, if there is an obstacle in the robot’s path, it will try and minimize goal distance by changing its heading, thus increasing the heading feature cost. In turn, this also increases the path distance cost, though this may or may not be linear.

\[
\text{cost}(v_x, v_y, v_\theta) = \alpha(\Delta_{\text{path}}) + \beta(\Delta_{\text{goal}}) + \gamma(\Delta_{\text{heading}}) + \delta(\Delta_{\text{occ}})
\]  

(2)

Building upon prior work done in this area, we include socially-aware navigation features such as interpersonal distance \((ID)\), distance from a group \((GD)\) and distance from a social goal \((SGD)\). As a way to dissuade the robot from getting too close to a human, a cost function was developed to penalize the robot at an exponential rate as the interpersonal distance decreases as seen in Equation 3 (for every human in the interaction scenario). Although we could penalize the robot based on this at all times, it is not necessary if the interpersonal distance is so significant that it would not be considered as a socially inappropriate distance. Therefore the robot is only penalized if the interpersonal distance is less than or equal to 1.5 meters. The interpersonal distance threshold was chosen to be 1.5 meters to ensure that the robot remains in social space and does not invade the personal space of the person [21].

\[
ID_f = e^{1/ID}
\]  

(3)

In order for the robot to not get too close to a group of people or not to get in between them, we penalized the robot based on \(GD\) whenever it is close to a group using Equation 4.

\[
GD_f = e^{1/GD}
\]  

(4)

Contrary to Equation 3 and 4, Equation 5 is like a reward to get closer to a social goal rather than an actual goal. With this feature in place, the robot tends to reach a social goal for a particular scenario while still adhering to the final goal location. For example, the social goal for reaching the front of a desk when others are waiting in a line is the end of the line. So, the robot will reach the social goal first (end of the line) and eventually reaches the desk when it is the robot’s turn.

\[
SGD_f = e^{SGD}
\]  

(5)
Instead of adding these features cost into the previous cost function, Equation 2, we make the assumption that its relationship with other features might be nonlinear and therefore gets treated as separate objectives. Since we know that the above cost function, Equation 2, works sufficiently enough from previous work [30], we can treat it as its objective as well. Instead of optimizing on just one objective, we need to optimize on multiple objectives, hence our multi-objective approach. Using a multi-objective tool like PaCcET requires computational time and since this is intended to work in real time any chance to improve the computation time should be utilized. Treating the first four features used in the previous cost equation as a single objective not only speeds up this process, but in turn allows for the possibility to add even more features to our local trajectory planner. Using PaCcET to do the multi-objective transformations we essentially get a new cost function with a PaCcET fitness denoted by $P_f$, which was modeled under the assumption of nonlinear relationships between the objectives. Equation 6 shows how $P_f$ is a transformation function dependent on multiple variables.

$$P_f = T_f(Obj_1, Obj_2, ..., Obj_n)$$

In this work, we are only interested in objectives like interpersonal distance, distance from a group and distance from social goal. The first objective is the original cost function (Equation 2), which is the linear combination of the path distance, goal distance, heading difference, and occupancy cost. The remaining objectives are the social features, such as interpersonal distance, distance from social goal, etc. Equation 7 shows the PaCcET fitness function with our proposed objectives.

$$P_f = T_f(cost(v_x, v_y, v_{\theta}), ID_{f1}, ..., ID_{fn}, GD_f, SGD_f)$$

where, $ID_{f1}, .. ID_{fn}$ are the cost functions associated with interpersonal distance between $n$ people and the robot.

### 3.2 Trajectory Planning

The robot’s trajectory can be broken into three parts, the global planner, the local planner, and low-level collision detection and avoidance. The global trajectory planner works by using knowledge of the map to produce an optimal route given the robot’s starting position and the goal position. The global path is created as a high-level planning task based upon the robot’s existing map of the environment; this is regenerated every few seconds in order to take advantage of shorter paths that might be found or to navigate around unplanned obstacles. The role of the traditional local planner is to stay in line with the global path unless an obstacle makes it deviate from the global path. The low-level collision detector works by stopping the robot if it gets too close to an object. We use the traditional global trajectory planner, and low-level collision detector [30] and make adaptations to the local trajectory planner to incorporate interpersonal distance and PaCcET.

The algorithms 1, 2 and 3 can be summarized as follows:

1. Discretely sample, the robot control space.
2. Depending on the type of the robot, for each sampled velocity ($V_x$, $V_y$ and $V_{\theta}$) perform a forward simulation from the robot’s current state for a short duration to see what would happen if the sampled velocities were applied.
3. Score the trajectories based on metrics:
   - a) Score each trajectory from the previous step for metrics like distance to obstacles, distance to a goal, etc. Discard all the trajectories that lead to a collision in the environment.
   - b) For all the valid trajectories, calculate the social objective fitness scores like interpersonal distance and other social features and store all the valid trajectories.
(4) Perform Pareto Concavity Elimination Transformation (PaCcET) on the stored trajectories to get a PaCcET fitness score and sort the trajectories from lowest to highest PaCcET fitness score.

(5) For a given time step, grab the trajectory with index 0 from the sorted list of valid trajectories.

Algorithm 1 shows the primary functions of the local trajectory planner and how the future trajectory points were stored to be used with PaCcET. The trajectory planner is called every time step, which in this case is every 0.1 second. Once the trajectory planner is called, the Transform_Human_State function is called to transform human pose to the robot’s Odom reference frame, which allows the interpersonal distance corresponding to each possible trajectory to be calculated in the Generate_Trajectory function. Now there are two methods of calculating the possible trajectories. The first is assuming that the robot can only move forward, backward, and turn. To produce the possible trajectories for this physical set up we loop through every combination of a sample of linear velocities ($v_x$) and angular velocities ($v_\theta$) to generate trajectories (For a holonomic robot, a slight change in $v_y$ is also used to generate possible trajectories). Once a trajectory is created, we determine if it is valid based on the constraints for the first objective. For example, trajectories that would make the robot hit a wall, obstacle or human are not considered strong trajectories and therefore will not be stored in the Store_Trajectory function. By not storing these invalid trajectories the speed at which PaCcET runs can be improved.

**Algorithm 1:** Local Trajectory Planner Algorithm. The trajectory planner generates multiple trajectories ($T$) given a number of $v_x$ samples and $v_\theta$ samples and calculates the independent cost for each feature. The cost for each feature is based on the robots sensing of the human’s state ($H_s$) and the robot’s state ($R_s$). At the end of a time step the best trajectory ($T_B$) is returned.

**Input:** $V_x$ samples, $V_\theta$ samples, $H_s$, $R_s$

**Output:** Best_Trajectory ($T_B$)

```plaintext
for Each time step do
    Transform_Human_State($H_s$, $R_s$)
    for Each $V_x$ do
        $T$ ← Generate_Trajectory($T$, $H_s$)
        if valid trajectory then
            Store_Trajectory($T$
        for Each $V_\theta$ do
            if Valid Trajectory then
                Store_Trajectory($T$
            $T$ ← Generate_Trajectory($T$, $H_s$)
        if Holonomic Robot then
            $T$ ← Generate_Trajectory($T$, $H_s$)
            if Valid Trajectory then
                Store_Trajectory($T$
        Run_PaCcET($T$
    Return $T_B$
```

, Vol. 1, No. 1, Article . Publication date: July 2019.
The second method assumes that the robot is capable of holonomic movement can translate with any $V_x$, $V_y$, $V_\theta$ that are less than velocity limits. Given these movements, we again loop through all the possible movements given the predefined number of $V_x$ samples, $V_y$ samples, and $V_\theta$ samples. Again, if the trajectories are valid, they are stored. Once all the valid trajectories are stored for all possible movements, the Run_PaCcET function runs, giving back the best possible trajectory, ($T_B$), based on its multi-objective transformation process.

In order to run a multi-objective tool like PaCcET, each objective’s fitness needs to be calculated. Algorithm 2 details the Generate_Trajectory function from Algorithm 1. The first function that needs to be performed is the Calculate_State function as the robot’s position, and velocity are used to determine the fitness values for the objectives. Using the state information the Compute_Path_Dist, Compute_Goal_Dist, Compute_Occ_Cost, and Compute_Heading_Diff functions are used to calculate the fitness values associated with the four pieces of the first objective. Using those fitness values the first objective’s fitness is calculated by the Compute_Cost function. Distance-based features like interpersonal distance of each person, group distance and social goal distance are calculated as shown in Algorithm 2 lines 8 - 10. Once all the objectives have their fitness values, the trajectory along with the fitness values is returned to the local trajectory planner algorithm which saves all the valid trajectories and calls PaCcET Algorithm 3 to perform optimization.

Algorithm 2: Generate Trajectory Algorithm. The generate trajectory function take in an instance of a trajectory ($T$) and the humans’ state ($H_s$) to compute the cost function for each feature. The trajectory ($T$) is then returned to the local trajectory planner.

<table>
<thead>
<tr>
<th>Input: $T, H_s$</th>
<th>Output: $T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \leftarrow$ Calculate_State($T$)</td>
<td>$path_dist \leftarrow$ Compute_Path_Dist($S$)</td>
</tr>
<tr>
<td>$goal_dist \leftarrow$ Compute_Goal_Dist($S$)</td>
<td>$occ_cost \leftarrow$ Compute_Occ_Cost($S$)</td>
</tr>
<tr>
<td>$heading_diff \leftarrow$ Compute_Heading_Diff($S$)</td>
<td>$cost \leftarrow$ Compute_Cost($path_dist, goal_dist, occ_cost, heading_diff$)</td>
</tr>
<tr>
<td>$for$ Each person $do$</td>
<td>$ID \leftarrow$ Calculate_Interpersonal_Distance($H_s, S$)</td>
</tr>
<tr>
<td></td>
<td>$GD \leftarrow$ Calculate_Group_Distance($H_s, S$)</td>
</tr>
<tr>
<td></td>
<td>$SGD \leftarrow$ Calculate_Social_Goal_Distance($H_s, S$)</td>
</tr>
<tr>
<td></td>
<td>Return Trajectory($T$)</td>
</tr>
</tbody>
</table>

3.3 Integrating PaCcET

At the end of Algorithm 1, all the valid trajectories have been stored along with their objective fitness scores in a vector of type trajectory. Algorithm 3 details the primary functions for determining a single fitness value from multiple objectives. In order to run PaCcET, the objectives for each trajectory must be stored in a vector of type double which is done in the Store_0bjectives function. Before running PaCcET’s main functions, an instance of PaCcET must be created. Next, the solution space and Pareto front are created by giving each trajectory to the Pareto_Choose function. Now that the Pareto front and its geometry has been calculated, PaCcET can transform the
solution space and give a single fitness value for each trajectory in the Compute_PaCcET_Fitness function. Once each trajectory has their PaCcET fitness, they are sorted from best to worst in the Sort_Trajectories function, which allows the function to not only ascertain the best trajectory easily, but is also useful for debugging purposes. Algorithm 3 concludes by returning the best trajectory to the local trajectory planner algorithm.

Algorithm 3: PaCcET Algorithm. PaCcET (P), takes in the vector of valid possible trajectories T to compute the multi-objective space and the PaCcET fitness (P_f) for each trajectory.

Input: T
Output: T_B
1. for Each trajectory do
2. Store_Objectives(T)
3. P ← Initialize_PaCcET()
4. for Each trajectory do
5. Pareto_Check(T)
6. for Each trajectory do
7. P_f ← Compute_PaCcET_Fitness(T)
8. Sort_Trajectories(T)
9. Return T_B

3.4 Social Goal

Fig. 6. Figure illustrating the computation of social goal in O-formation scenario. The red star represents the social goal.

Computing the social goal location is important because often, the actual goal location may not be an appropriate location for interaction and explicitly commanding the social goal would not be possible. A social goal can be defined as an appropriate location for a robot to involve in human-robot interaction. For example, in a front desk like scenario, the end of the line can be considered as a social. For this work, the social goals for each interaction scenario were geometrically computed, for O-formation scenario (joining a group), we fit a circle with the people in a group and find a socially appropriate spot to join the group as shown in Figure 6.

We find angle made by every person with the center of the formed circle using the law of cosines, equation 8 as shown below:
\[ c^2 = a^2 + b^2 - 2ab \cos(\theta) \]  
\[ \theta_{ij} = \cos^{-1}\left(\frac{(2R^2 - D_{ij})}{2R^2}\right) \]

Where \( D_{ij} \) is the Euclidean distance between person \( i, j \) and \( R \) is the radius of the circle formed by all the people in the group. Out of all the \( \theta_{ij} \)'s, we pick one half of the widest angle as approach angle denoted by \( \theta_a \). Now, joining a group problem (O-formation) boils doing to finding the intersection of two circles, one formed by the people in the group and the other formed in the wide-open sector with the center as the locations of either of the people making the widest sector. The equations of the two circles to solve for are given as follows:

\[(x - h)^2 + (y - k)^2 = R^2 \]  
\[(x - h_p)^2 + (y - k_p)^2 = r^2 \]

Where, \((h, k)\) is the center of the circle formed by the group of people, \((h_p, k_p)\) is the location of one of the person that formed the widest sector. The radius \( r \) in Equation 11 is obtained by solving for \( c \) in Equation 8 where, \( \theta = \theta_a \), \( a \) and \( b \) equals \( R \), radius of the group formation. There are two solutions when solving Equations 10 and 11, we further filter one social goal from the two solutions.

Similarly, we can fit a straight line as shown in Figure 7 for waiting in a queue scenario and social goal location would be the end of the line considering personal space of the last person in the line. Hence, in this case, the solution boils down to solving for the intersection of a line and a circle.

The equation of the line formed by the people can be found by fitting a line of form \( y = mx + c \) with the people locations. The circle formed using the last person’s location as the center and a comfortable distance that the robot should maintain around the last person as the radius is of the form \((x - k)^2 + (y - h)^2 = r^2\). The two solutions to the line and circle intersection can be obtained using quadratic roots, and the social goal is further filtered to the solution farthest to the actual goal (desk).

The social goal calculation for scenario 2 (art gallery) is hand-selected. Computing the social goal in this scenario is beyond the scope of this paper. Art gallery scenario requires a perception pipeline that can detect the location of the art on display, the area, perimeter of the artwork to compute a useful social goal location to interact with a person viewing the art. The social goal for scenario 2 will be addressed when we implement our proposed method on a real PR2 robot, see section 5.
4 RESULTS

In order to validate our proposed approach, we considered four different scenarios namely, a hallway scenario, an art gallery scenario, forming a group, and waiting in a queue. The hallway scenario was divided into four sub-scenarios namely, passing a stationary human, passing a walking human in the same direction, an encounter with a human walking on the appropriate side, and passing a walking human in the opposite direction. All this was tested in simulation using the 2D simulator, Stage on a machine with an Intel 6th-generation i7 processor @3.4 GHz, 32 GB of RAM. The simulated environment for each hallway experiment was the second-floor hallway of the Scrugham Engineering and Mines building at the University of Nevada, Reno. The map of the building used in the simulation was built using the gmapping package for SLAM on the PR2. The simulated PR2 is identical to the real-world PR2 for sensing and movement capabilities and is using AMCL for localization on the map. The simulated PR2 uses a 30-meter range laser scanner.
that is identical to the real PR2 robot’s laser scanner. The humans in the simulation exhibit very simple motion behaviors. Follow-up scenarios are simulated in a 25m x 25m open space in the stage environment.

4.1 Simple scenario

In the first experiment, the robot was tasked with getting to a goal while passing close to a static simulated human. Figure 8a shows that when using the traditional planner, the robot made sure to avoid a collision with the simulated human, but did not consider any social distance. The same will be the case for the other experiments as well, since the traditional planner does not consider interpersonal distance into its cost function. The PaCcET-based planner did consider interpersonal distance, and therefore the robot deviated from a more straight lined path as a way to satisfy the second objective. Once the threshold for the interpersonal distance was no longer an issue, the robot only needed to minimize the first objective, therefore, returning to a straight-line path. It is worth noting that in all the experiments conducted the robot also considered a wall as an obstacle and was required to disregard trajectories that would lead to a collision with the wall, which is why the robot did not deviate from global trajectory even more.

The second experiment was developed to mimic a passing scenario where the robot has a set goal but needs to pass by a simulated human who is traveling much slower in the same direction. Figure 8b shows that with the traditional trajectory planner it merely made sure that a collision would not take place as it tried to minimize its cost function. The PaCcET-based planner deviated from its global trajectory in order to consider the interpersonal distance objective, then returned to the global trajectory once the threshold for the interpersonal distance was no longer an issue.

Similar to the previous experiment, the third experiment involves both the simulated human and robot moving; however, in this case, the simulated human is now moving at a normal walking speed in the opposite direction of the robot. The robot and simulated human pass close to one another but not close enough to cause a collision. Figure 8c shows that the traditional trajectory planner altered its path ever so slightly to ensure that a collision would not happen, where the PaCcET-based trajectory planner not only ensured that a collision would not take place but also considered interpersonal distance and provided the simulated human with additional space while passing.

The previous experiments show that when using a PaCcET-based trajectory planner interpersonal distance can be considered in selecting a local trajectory in both static and dynamic conditions where a collision is not imminent; however, the case of a collision that would occur unless either the simulated human or the robot moves out of the way also needs to be considered. This experiment considers a simulated human who is not paying attention or unwilling to change their course and walking directly towards the robot. Figure 8d shows that the traditional trajectory planner was successful at avoiding the collision as expected; however, it did so while minimizing its cost function as much as possible which caused the robot to get very close to the simulated human. When using the PaCcET-based trajectory planner, the robot not only avoided the collision but also gave the simulated human additional space as to satisfy the interpersonal distance objective. It is worth noting that once the interpersonal distance threshold was no longer an issue, the robot used its holonomic movement for a short time as a way to quickly minimize the heading difference portion of the original cost function objective.

4.2 Complex Scenarios

In the previous section, 4.1, we showed that by just considering one social feature, i.e., interpersonal distance, our approach was able to account for personal space while navigating a hallway (with different maneuvers of a human partner). In this section, we will see the results of our approach
applied to complex social scenarios. We staged these scenarios keeping in mind a few situations a social robot might encounter during its deployment in a human environment.

Figure 9 shows the behavior of our social and traditional planners in an art gallery situation (three objectives). We considered an art gallery scenario, but it can be generalized to other similar scenarios like a tour guide robot in a museum or an attraction. For this scenario, we staged a human–robot interaction consisting of a robot presenting a piece of art (hanging to a wall) to a human standing nearby. Both the traditional planner (red line) and the SAN planner (blue line) were given the same goal (represented as a red star) and start (START) locations. The traditional planner steered the robot to the goal location, cutting the standing person from the back (inappropriate). On the other hand, the SAN planner steered the robot to a location that is appropriate to present the details of the art to the human (social goal). The SAN planner approached the social goal, leaving enough personal space based on the interpersonal distance feature.

Figure 10 shows the behavior of our social and traditional planners in the waiting in a queue situation (five objectives). Here, we considered a front desk interaction, but this can be generalized to other similar social scenarios where a robot or a human is required to form a line before reaching...
Fig. 11. Scenario 4: The robot is joining a group where the robot with SAN planner forms an O-formation in order to interact with the group. The traditional planner generates the red trajectory and places the robot in the center of the group. Proposed SAN planner generated the blue trajectory which leads the robot to form an O-formation.

the goal. For example, social scenarios like getting coffee from a public coffee machine, taking an elevator, etc. In this context, we staged a human-robot interaction consisting of a robot that wants to interact with a front desk representative of an office building where other people were being served on a first come first served basis. Both the traditional planner (red line) and the SAN planner (blue line) were given the same goal (represented as a red star) and start (START) locations. The trajectory planner tried to steer the robot to the goal location and stopped at an inappropriate location (besides the person currently being served) as the traditional planner treated the human as an object. On the other hand, SAN planner steered the robot to an appropriate location, i.e., end of the line positioning the robot behind the last person (social goal), considering personal space as well.

Figure 11 shows the behavior of our social and traditional planners in Joining a group situation. Here, we considered an HRI situation where the robot is required to join a group of three people. However, this can be generalized to interact with more people. Both the traditional planner (red line) and the SAN planner (blue line) were given the same goal (represented as a red star) and start (START) locations. The trajectory planner steered the robot to position it at an awkward location (middle of an interacting group) as the traditional planner did not account for group proxemics and group dynamics. On the other hand, SAN planner steered the robot to an appropriate location, i.e., a vacant spot in the circle, considering group proxemics.

4.3 Robustness

Figure 12 shows results with variations in scenarios 3 and 4 (waiting in a queue and joining a group). While figures 12a and 12b capture variations in locations of people and the orientation of the queue formed by them, figures 12c and 12d capture interesting results.

Figure 12c shows a variation in the open spot where the robot needs to join. In this case, the open spot is in a tricky location as the robot has to approach the group from the back. Our proposed approach found a way around the group to the social goal. On the other hand, the traditional planner leads the robot to the center of the group while getting in between two people (blue and green shirts).

Figure 12d not only differs in the size of the circle formed but also is a variation of O-formation where the robot is leading the conversion opposing to joining a conversation. When joining a group for discussion, we tend to main a uniform spacing between every member of a group wherein
(a) Scenario 3 (location change): The robot is joining a line, formed in front of a desk scenario. The traditional planner generated the red trajectory, guiding the robot between the first two people (inappropriate). The blue trajectory, our proposed approach, leading the robot to join the line (appropriate).

(b) Scenario 3 (location and orientation change): The robot is joining a line, formed in front of a desk scenario. The traditional planner generated the red trajectory, guiding the robot to the front of the desk, cutting the line (inappropriate). The blue trajectory, our proposed approach, leading the robot to join the line (appropriate).

(c) Scenario 4 (change in groups open spot): The traditional planner generated the red trajectory and placed the robot in the center of the group while navigating between two people (inappropriate). Proposed approach generated the blue trajectory which leads the robot to form an O-formation (appropriate).

(d) Scenario 4 (Robot leading the group): The traditional planner generated the red trajectory and placed the robot in the center of the group. Proposed approach generated the blue trajectory which leads the robot to form an O-formation (appropriate).

Fig. 12. Changes in scenarios 3, 4 with different people locations and formations.

joining a group to lead the conversation, all the members of the group except the lead squish together so that the leader can make eye contact with all the members (with as little field of view as possible). In this case, the traditional planner guides the robot to the center of the group whereas the proposed method guides the robot to a social goal location where the robot can effectively interact with the group.

All the above experiments were conducted in simulation, but in the real world, these interactions can be different, for example, in the Joining a group scenario, at first, there might not be as wide of an open spot as needed but as someone approaches to join the conversation, we humans tend to widen up the group by adjusting the dynamics of the group accordingly. We are planning to
address the human side of the interaction with follow-up work, and detailed our plans in the next section, 5.

5 DISCUSSION AND FUTURE WORK

With a series of experiments in simulation using a multi-objective optimization tool like PaCcET at the local planning stage of navigation, we showed that social norms related to proxemics could be addressed in a SAR system which in turn can aid the acceptance of robots in human environments. We showed that our approach could handle a simple, single person interaction in a hallway scenario. We also showed that our approach could handle sub-scenarios (different types of interaction in a scenario) such as a stationary human in a hallway, passing behavior, etc. We demonstrated the generalizability of our approach by introducing multiple humans in complex social scenarios with multiple features like interpersonal distances, group proxemics and social goal distance (adherence to social goal). Finally, we showed that our modified local planner can adjust to changes in the scenarios like the location of people, different line formations and O-formations.

We will extend this work by implementing the PaCcET local planner on a real PR2. We will validate this system using its conformity to social metrics defined by the social parameters we discussed as well as surveying the perception of social intelligence of the resultant behavior. We plan to utilize not just the distance-based feature but also features related to orientation like heading angle, heading difference between the robot and the people/group, etc along with features related to environment like position of the agents (robot and people) in the environment (for example, in a hallway, distance from the right side of the hallway) [39].

When examining the social impact of a SAN system, it is important that any instruments used are properly assessing social intelligence. Kruse et al. [29] identified Comfort, Sociability, and Naturalness as challenges that SAN planners should tackle in a collaborative human environment [29]. We identified other challenges like predictability, legibility, safety, acceptance, etc., and working on providing clear definitions and metrics/methods for measuring them. Perceived Social Intelligence (PSI) is an important parameter we identified which has importance in robot motion in human environments. Social intelligence is the ability to interact effectively with others to accomplish one’s goals [19]. Social intelligence is critically important for any robot that will be around people, whether engaged in social or non-social tasks. Some aspects of robotic social intelligence have been included in HRI research [7, 25, 33, 35], but current measures are brief and often include extraneous variables. We designed a comprehensive instrument for measuring the PSI of robots [5, 6]. This should more precisely measure the social impact of our approach.

6 CONCLUSION

We presented a novel approach to the socially-aware navigation (SAN) problem at a local planning stage, by transforming the Pareto front to an objective space (forced to be convex) using Pareto Concavity Elimination Transformation (PaCcET) method. PaCcET was implemented in local planning of well established ROS navigation stack to deal with spatial communication at a low-level planning stage. We validated the developed system in simulation to show the applicability of our proposed approach with multiple scenarios involving multiple humans. Follow-up study will implement the proposed method on a real PR2 and capture the social aspect of the navigation behaviors using scales that already exist and new scales that our group is currently investigating.

ACKNOWLEDGMENT

The authors would like to acknowledge the financial support of this work by the National Science Foundation (NSF, #IIS-1719027), Nevada NASA EPSCoR (#NNX15AI02H), and the Office of Naval
We would like to acknowledge the help of Vineeth Rajamohan and Andrew Palmer.

REFERENCES


[23] Isobel Asher Hamilton. 2018. People kicking these food delivery robots is an early insight into how cruel humans could be to robots. https://www.businessinsider.com/


