Socially-Aware Navigation Using Non-Linear Multi-Objective Optimization

Scott Forer[1], Santosh Balajee Banisetty[2], Logan Yliniemi[3], Monica Nicolescu[4], and David Feil-Seifer[5]

***Abstract*— For socially assistive robots (SAR) to be accepted into complex and stochastic human environments, it is important to account for subtle social norms. In this paper, we propose a novel approach to socially-aware navigation (SAN) which garnered an immense interest in the Human-Robot Interaction (HRI) community. We use a multi-objective optimization tool called the Pareto Concavity Elimination Transformation (PaCcET) to capture the non-linear human navigation behavior, a novel contribution to the community. We use autonomously sensed distance-based features that captures the social norms and associated social costs for a given trajectory point towards the goal. Rather than use a finely-tuned linear combination of these costs, we use PaCcET to select an optimized future trajectory point, associated with a non-linear combination of the costs. Existing research in this domain concentrates on geometric reasoning, model-based, and learning approaches, which have their own pros and cons. This approach is distinct from prior work in this area. We showed in a simulation that the PaCcET based trajectory planner not only is able to avoid collisions and reach the intended destination in static and dynamic environments but also considers a human's personal space in the trajectory selection process.**

I. Introduction

Recent technological advancements in sensing and computation have stimulated a greater interest in the application of autonomous agents to real-world interaction. In particular, researchers and commercial interests have experimented using such mobile robots to provide assistive services such as guiding and carrying luggage in complex, pedestrian dense environments (shopping malls, airports, and other public places) [1]. One such example is the work conducted by Shi *et. al* [2], where an autonomous robot was used to distribute flyers at a shopping mall. Robot domains, especially socially assistive robotics (SAR), benefit from navigation; such movement extends the reachable service area of the robot [3]. However, navigation, if not performed properly, can cause negative social reactions [4].

Socially Aware Navigation (SAN) utilizes space, distance, and movement as a spatial communication medium. For human-human interaction, humans understand spatial communication and navigate in such a way that social norms are obeyed. For HRI to match this human-human interaction property, the spatial communication between a human and a robot should not be neglected; it should be utilized to achieve human-friendly navigation [5]. An effective robot's actions, including actions involving spatial communication, must be suitable for a given social circumstance. Hence, utilizing spatial communication between a robot and a human is very important for assistive robots.

Proxemics [6], social rules for interpersonal distance, is an important aspect of navigation; researchers interested in SAN are investigating methods to integrate the rules of proxemics into robot navigation behavior. Kruse *et. al* [7] authored an extensive review of methods like hard-coded rules, geometric reasoning, Model-based Inverse Reinforcement Learning (IRL) that tried incorporating the rules of proxemics.

A current review of existing approaches show the following limitations:

1) Some of the approaches depend on exocentric sensing hence, limiting the robot's services to a particular environment.
2) The environment/scenario is a singleton, i.e., Only a hallway, a room, etc is considered or only an approach behavior, or a passing behavior is considered.
3) Planners are optimized for single or few objectives with a linear combination or weighted sum.

Our prior work [5] presented an approach utilizing a spatial model over distance-based features to generate trajectories that are socially appropriate, which was validated with human partners. Limitations 1 and 2 were addressed in our recent work [8], [9], where a Gaussian Mixture Model (GMM) based approach was used to select an appropriate trajectory for an autonomously sensed social scenario. An egocentric laser-based sensing method was used to calculate distance based features that were used in sensing the social scenario. In this work, we propose a novel multi-objective optimization approach for a socially-aware navigation planner.

Linear methods for socially-aware navigation may be inadequate for a number of reasons:

- Navigation rarely involves optimizing for a single objective. For example, humans optimize for path length, execution time and most importantly, social norms while walking from one place to the other.
- The human environment is too complicated for linear approaches to effectively approximate. For example,

[1]Scott Forer is with the Department of Mechanical Engineering, University of Nevada, Reno, 1664 N. Virginia Street, Reno, NV 89557-0171, USA sforer@nevada.unr.edu

[2]Santosh Balajee Banisetty with the Department of Computer Science and Engineering, University of Nevada, Reno, 1664 N. Virginia Street, Reno, NV 89557-0171, USA santoshbanisetty@nevada.unr.edu

[3]Logan Yliniemi is with the University of Nevada, Reno, 1664 N. Virginia Street, Reno, NV 89557-0171, USA logan@unr.edu

[4]Monica Nicolescu with the Department of Computer Science and Engineering, University of Nevada, Reno, 1664 N. Virginia Street, Reno, NV 89557-0171, USA monica@cse.unr.edu

[5]David Feil-Seifer is with the Department of Computer Science and Engineering, University of Nevada, Reno, 1664 N. Virginia Street, Reno, NV 89557-0171 USA dave@cse.unr.edu

when a human optimizes on multiple objectives some of the objectives have a relationship that causes non-linear trade offs.

For these reasons, a multi-objective optimization approach capable of handling multiple objectives independently is warranted.

Further details on one such approach is presented in section IV. The remainder of this work is organized as follows: Section II presents previous work that we build upon. Section III provides the necessary background. Section IV describes our experimental approach. Section V provides our experimental validations. Finally, Section VI concludes this work and provides directions of future research.

II. Related Work

When we talk about SAN, there are many scenarios that researchers address, such as maintaining pace with a person, following a people, approaching a person, and other social hallway behaviors. There are many ways of achieving one or more behaviors. For example, for a socially appropriate hallway behavior, one can increase the cost for traveling on the left side of the hallway so that the robot always navigates on the right side. Similar behavior can be achieved by utilizing supervised/unsupervised learning and reinforcement/inverse reinforcement learning techniques. We will discuss some of these methods in this section.

Recently, research and applications related to SAN are of great interest in the HRI field as it offers a way to incorporate social norms into a robot. Ferrer *et. al* [10] presented a social-force model (SFM) approach to accomplish a robot companion behavior where such model allows the robot to accompany a human partner to the desired goal. Dondrup *et. al* [11] proposed a combination of well-known sample-based planning and velocity costmaps to achieve socially-aware navigation. In this work, 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. Kruse *et. al* [12] concentrated behavior cues that can impact motion legibility. The author uses so-called directional costs to achieve SAN instead of using regular spatial cost which can be confusing in an unpredictable human environment. The said directional cost approach resulted in a less ambiguous robot motion while crossing a human. Lu *et. al* [13] proposed context-sensitive navigation using a layered approach to costmaps. The idea is to use different costmaps to handle a different situation, for example, one layer dedicated to tackling proxemics, another layer to handle hallways, etc. A similar approach combined with Inverse reinforcement learning (IRL) was proposed in [14].

Due to recent advancements in computational methods, IRL has gained popularity in the machine learning community. IRL can be used to train human navigation behavior policy in order for the robot to emulate social behavior [14], [15], [16], [17]. Ramirez *et. al* [14] proposed two planners that use the layered costmap approach in combination with IRL to solve the problem of "how and where to approach a person." Kuderer *et. al* [15] proposed a feature-based maximum entropy IRL to achieve a navigation policy from teleoperated interactions with humans. Ramon *et. al* [16] used Gaussian Process Inverse Reinforcement Learning (GPIRL) to train and evaluate a control policy on a publicly-available dataset. Kim *et. al* [17] proposed an IRL based framework that can perform adaptive path planning in a hallway setting with people.

While IRL can be used to emulate social behavior, such IRL approaches can have limitations. One such limitation is space exposition, where as the number of states and actions increase, the IRL needs a lot of training data. In this case, an expert would be required teach the robot by teleoperating it for any social scenarios the robot would need to emulate. As human environments are highly unpredictable and stochastic, an IRL-based approach might not scale efficiently.

III. Prior Work

A. 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 there are multiple variables that go into 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 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 most optimal supply amount such that there is enough demand to turn a profit. If this supply and demand curve were graphed, as shown in **Figure 1**, the points on the line would be Pareto optimal points i.e. no point dominates each other. In this case the seller is trying to maximize both objectives, therefore the hollow circles are the dominated points as there exists solid circle points that are better in both objectives. Typically there are multiple Pareto optimal points forming a set which is the solution type that many multi-objective algorithms use [18].

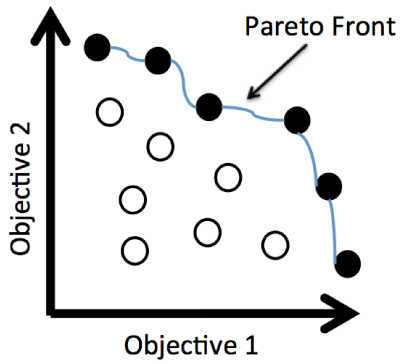

Fig. 1. *Multi-Objective solution space* - The Pareto front contains the non-dominated solutions base on the the two objectives.

Multi-objective optimization has already started to play a role in real world applications [19]. Some examples of real world multi-objective scenarios are high speed civilian aircraft transportation [20], urban planing [21], and for designing trusses [22]. The trajectory planner used in this paper builds off of a pre-existing one that utilizes a multi-objective approach along with linear combination [23].

B. PaCcET

In some cases optimizing a single objective does not yield the performance that is desired and therefore multiple objectives need to be considered when evaluating a policy's fitness. A common method is to simply 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 sub optimal policies. A solution to this is to use a multi-objective tool, such as PaCcET, to properly evaluate policies on multiple objectives [24], [25]. 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 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. This 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 [26], the operation of the electrical grid on naval vessels [27], the coordination of multi-robot systems [28], and for the efficient operation of a distributed electrical microgrid [29], 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 it functions at or above the solution quality of other techniques like NSGA-II or SPEA2 [24], with as low as one tenth of the run time.

For the purpose of this project PaCcET was used over other multi-objective tools because of its computational speed [24]. 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 trajectories. PaCcET then uses the fitness values for each feature of each future trajectory to develop the solution space and obtain the optimal future trajectory. Since at each time step, a future trajectory is developed independently, PaCcET develops a brand new solution space at each time step. By using PaCcET like this the local planner can be optimized in real time.

IV. Method

In this section we detail our methodology of the navigation planner, the features or objectives that we used to optimize the trajectories, and how PaCcET was implemented in the local trajectory selection process. The overall function of the local trajectory planner at each time step is to generate an array of possible future trajectories points and evaluate each future trajectory point based on the predefined feature set as shown in **Figure 2**. 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.

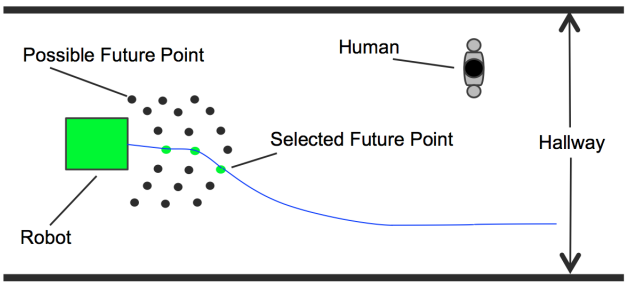

Fig. 2. *Navigation Planner* - The navigation planner selects a short-term trajectory (green points represent potential trajectory end-points) from the pool of possible trajectories (black points), optimized for adherence to a long-term plan (blue line), obstacle avoidance, and progress toward a goal, and in the case of this paper, interpersonal distance.

A. Features/Objectives

In previous work the features that are extracted are each assigned their own cost. For example the path distance cost is the length that the robot has already traveled and the goal distance cost is the distance the robot is from the goal. In this case 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 the linear combination, or weighted sum in this case, 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 to 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.

$$\text{cost}(v_x, v_y, v_\theta) = \alpha(\Delta_{path}) + \beta(\Delta_{goal}) \tag{1}$$

More recently this cost function has been adapted to include a heading difference feature and an occupancy (occ) cost feature, where the heading difference is the distance that the robot is from the global path and the occ cost is the occupancy 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 goal distance it becomes less clear if there is only the 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 cost heading feature cost. This in turn also increases the path distance cost, though this may or may not be linear.

$$\begin{aligned}\text{cost}(v_x, v_y, v_\theta) = \alpha(\Delta_{path}) + \beta(\Delta_{goal}) + \gamma(\Delta_{heading}) \\ + \delta(\Delta_{occ})\end{aligned} \tag{2}$$

Building upon the prior work done in this area, we include a socially-aware navigation feature such as interpersonal distance (ID). 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**. Although we could penalize the robot based on this at all times, it really isn't necessary if the interpersonal distance is so large that it wouldn't be considered a socially inappropriate distance. Therefore the robot is only penalized if the interpersonal distance is less than or equal to 1.5 meters.

$$ID_f = e^{1/ID} \tag{3}$$

Instead of adding this feature's cost into the previous cost function, we make the assumption that its relationship with other features might be nonlinear and therefore gets treated as its own objective. Since we know that the above cost function works sufficiently enough from previous work, we can treat that as its own objective as well. Now instead of optimizing on just one objective we need to optimize on multiple objectives, hence our multi-objective approach. As one can imagine 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. In this case 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 4** shows how P_f is a transformation function dependent on multiple variables.

$$P_f = T_f(Obj_1, Obj_2, ..., Obj_n) \tag{4}$$

In this work we are only interested in two objectives. The first objective is the original cost function, which is the linear combination of the path distance, goal distance, heading difference, and occ cost. The second objective is the interpersonal distance, a social feature. **Equation 5** shows the PaCcET fitness function with our proposed objectives.

$$P_f = T_f(cost(v_x, v_y, v_\theta), ID_f) \tag{5}$$

B. Trajectory Planning

The robot's trajectory can be broken into three parts, the global planner, the local planner, and a low-level collision detection and avoidance. The global trajectory planner works by using knowledge of the map to produce an optimal route given the robots staring position and the goal position. The global path is created as a high level panning task however, the global path can be recreated if the robot has to deviate to far from the current global path. The role of the traditional local planner is to stay in line with the global path unless an obstacle makes it so the robot has to deviate from the global path. The low-level collision detector simply works by stopping the robot if it gets too close to an object. In this work we use the traditional global trajectory planner and low-level collision detector [23] and make adaptations the local trajectory planner to incorporate interpersonal distance and PaCcET.

Algorithm 1 shows the main functions of the local trajectory planner 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 seconds. Once the trajectory planner is called the `Transform_Human_State` function is called to compute the human state from the human reference frame to the robots state from the robots 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 forwards, backwards, 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_θ) to generate 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 valid 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.

The second method is assuming that the robot is capable of holonomic movements i.e., the robot can move forwards, backwards, left, right, and turn. Given these movements, we again loop through all the possible movements given the predefined number of V_x samples, V_y samples, and V_θ 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, ($\mathcal{T}_\mathcal{B}$), 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**

Algorithm 1: `Local Trajectory Planner Algorithm`. The trajectory planner generates multiple trajectories (T) given a number of V_x samples and V_θ 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 ($\mathcal{T}_\mathcal{B}$) is returned.

Input: V_x samples, V_θ samples, H_s, R_s
Output: $Best\_Trajectory$

for *Each time step* **do**
  $Transform\_Human\_State\,(H_s, R_s)$
  for *Each* V_x **do**
    $\mathcal{T} \rightarrow Generate\_Trajectory\,(T,\ H_s)$
    if *valid trajectory* **then**
      $Store\_Trajectory\,(\mathcal{T})$
    for *Each* V_θ **do**
      $\mathcal{T} \rightarrow Generate\_Trajectory\,(T,\ H_s)$
      if *Valid Trajectory* **then**
        $Store\_Trajectory\,(\mathcal{T})$
  if *Holonomic Robot* **then**
    $\mathcal{T} \rightarrow Generate\_Trajectory\,(T,\ H_s)$
    if *Valid Trajectory* **then**
      $Store\_Trajectory\,(\mathcal{T})$
  $Run\_PaCcET\,(\mathbb{T})$
  $Return \Longleftrightarrow (\mathcal{T}_\mathcal{B})$

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 determining 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. In this work the interpersonal distance is also considered as its own objective and therefore is calculate in the `Calculate_Interpersonal_Distance` function. Once all the objectives have their fitness values, the trajectory is sent back to the local trajectory planner algorithm.

C. Integrating PaCcET

At the end of **Algorithm 2**, all the valid trajectories have been stored along with their objective fitness scores in a vector of type trajectory. **Algorithm 3** details the main 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_Objectives` function. Before

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

Input: T, H_s
Output: T

$\mathcal{S} \rightarrow Calculate\_State\,(\mathcal{T})$
$path\_dist \rightarrow Compute\_Path\_Dist\,(\mathcal{S})$
$goal\_dist \rightarrow Compute\_Goal\_Dist\,(\mathcal{S})$
$occ\_cost \rightarrow Compute\_Occ\_Cost\,(\mathcal{S})$
$heading\_diff \rightarrow Compute\_Heading\_Diff\,(\mathcal{S})$
$cost \rightarrow Compute\_Cost\,()$
$\mathcal{ID} \rightarrow Calculate\_Interpersonal\_Distance\,(H_s,\ \mathcal{S})$
$Return \Longleftrightarrow Trajectory(\mathcal{T})$

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_Check` 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 to not only to easily ascertain the best trajectory but is also useful for debugging purposes. **Algorithm 3** concludes by returning the best trajectory to the local trajectory planner algorithm.

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

Input: $\mathbb{T}$
Output: $\mathcal{T}_\mathcal{B}$

for *Each trajectory* **do**
  $Store\_Objectives\,(\mathcal{T})$
$\mathcal{P} \rightarrow Initialize\_PaCcET\,()$
for *Each trajectory* **do**
  $Pareto\_Check\,(\mathcal{T})$
for *Each trajectory* **do**
  $P_f \rightarrow Compute\_PaCcET\_Fitness\,(\mathcal{T})$
$Sort\_Trajectories\,(\mathbb{T})$
$Return \Longleftrightarrow (\mathcal{T}_\mathcal{B})$

V. Validation

When using a multi-objective tool like PaCcET, two key validations play a role in our experimental design: first, does using PaCcET yield local trajectories that get to the goal in an efficient manner; second, is there a clear distinction that the

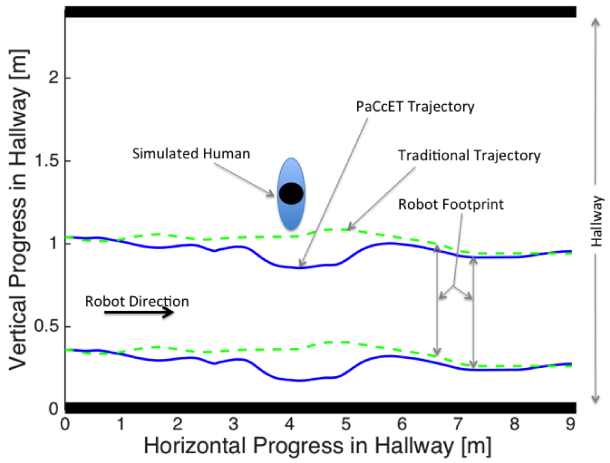

Fig. 3. *Experiment 1* - Robot path and simulated human position for the robot closely passing by a static simulated human. The dashed green lines represents the robot's trajectory footprint when using the traditional trajectory planner and the blue solid lines represent the robot's trajectory footprint when using the PaCcET trajectory planner.

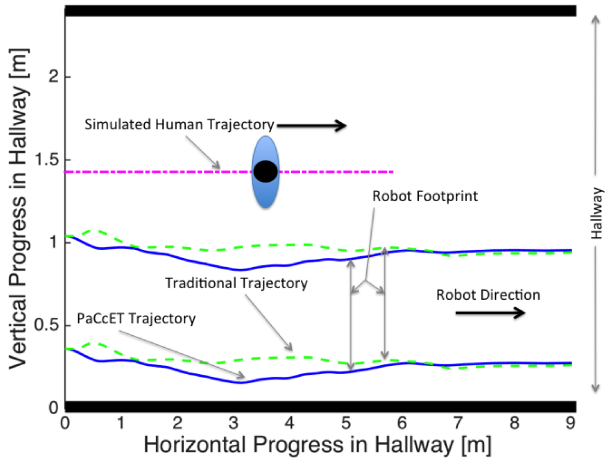

Fig. 4. *Experiment 2* - Robot and simulated human path for the robot closely passing by simulated human walking slowly in the same direction. The dashed green lines represents the robot's trajectory footprint using the traditional trajectory planner, the blue solid lines represents the robot's trajectory footprint using the PaCcET trajectory planner, and the dashed magenta line represents the simulated human's trajectory.

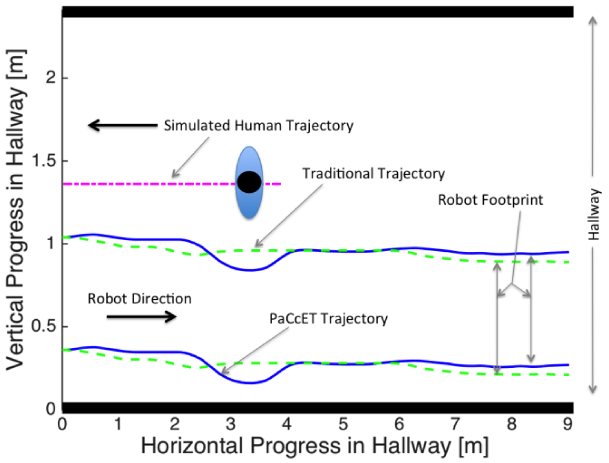

Fig. 5. *Experiment 3* - Robot and simulated human path for the robot closely passing by simulated human walking in the opposite direction. The dashed green lines represents the robot's trajectory footprint using the traditional trajectory planner, the blue solid lines represents the robot's trajectory footprint using the PaCcET trajectory planner, and the dashed magenta line represents the simulated human's trajectory.

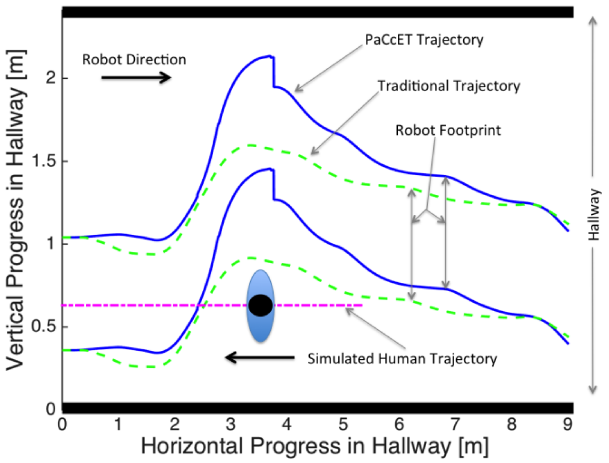

Fig. 6. *Experiment 4* - Robot and simulated human path for the robot avoiding a head on collision with a simulated human walking in the opposite direction. The dashed green lines represents the robots footprint using the traditional trajectory planner, the blue solid lines represents the robots footprint using the PaCcET trajectory planner, and the dashed magenta line represents the simulated human's trajectory.

other objective is being properly incorporated when selecting a local trajectory? To test if this approach was working, four experiments were conducted in simulation to test the quality and robustness of our method in static and dynamic environments. In the four distinct scenarios we show that not only is PaCcET assisting in selecting efficient trajectories based on the original cost function, but it can also include interpersonal distance into its evaluation process without any need for tuning parameters. In each experiment the path of the robot using the traditional local planner and the PaCcET based local planner are shown along with the simulated human's position or trajectory.

The simulated environment for each experiment was second floor hallway of the Scrugham Engineering and Mines building at the University of Nevada, Reno. The map of the building used in simulation was built using gmapping on the actual PR2. The simulated PR2 is identical to the real-world PR2 in terms of sensing capabilities and is using AMCL for localization on the map. For example, the simulated PR2 uses a 30 meter range laser scanner which is identical to the real PR2 robot's laser scanner.

A. *Experiment 1*

In the first experiment, the robot was tasked with getting to a goal while passing closely to a static simulated human. **Figure 3** shows that when using the traditional planner the

robot made sure to avoid a collision with the simulated human however did not consider any social distance. This 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 loner an issue the robot only needed to minimize the first objective therefore returning to a straight-line path. It's 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.

B. Experiment 2

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 4** 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 clearly deviated from its global trajectory in order to consider the interpersonal distance objective, then return to the global trajectory once the once threshold for the interpersonal distance was no loner an issue.

C. Experiment 3

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 5** 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.

D. Experiment 4

The previous experiments show that when using a PaCcET-based trajectory planner interpersonal distance can be considered when selecting a local trajectory in static and dynamic conditions when a collision is not imminent; however, the case of a collision that will 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 not paying attention or unwilling to change their course and walking directly towards the robot. **Figure 6** shows that the traditional trajectory planner was successful at avoiding the collision as expected however, did so along with 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 for a short time used its holonomic movement as a way to quickly minimize the heading difference portion of the original cost function objective.

VI. Discussion and Future Work

From the results in the previous section, it is clear that our proposed planner behaved more consistently with an agent that considers the social factor specified, when compared to a traditional planner. The PaCcET navigation planning attributes, such as path length and time to reach the goal are slightly greater than that of the traditional planner. This means that the proposed approach generated trajectories that were sub-optimal in the sense of efficient path planning however, more optimal from a social aspect. The PaCcET navigation planner generated trajectories that are not only safe (for both the agents and environment) but also considered the personal space for the simulated human partner.

By utilizing egocentric sensing, our proposed planner achieved socially acceptable trajectories that are optimized for various objectives, which also includes interpersonal distance. All this was tested in simulation using a 2D simulator called Stage on a machine with an Intel 6th-generation i7 processor @3.4 GHz, 32 GB of RAM. The PR2 robot in the simulation is identical to the real PR2 in terms of navigation capabilities and sensing. Implementing it on the PR2 in a real-world setting such as hallways, open spaces, etc, and measuring both qualitative and quantitative metrics is our next step. In future work, we plan on collecting data with regards to the performance of the planner, i.e. time, path length, distance maintained from a human, etc, and compare this to the traditional planner [23] or an openly available IRL SAN planner [17]. We will also compare the social aspects of the PaCcET planner to the traditional planner or an openly available IRL SAN planner.

VII. Conclusion

We presented a novel approach to socially-aware navigation and showed in simulation that the trajectories generated by this approach are better in comparison with a traditional planner. This new planner optimizes not only for shortest distance and other traditional planning performance metrics, but also includes a social factor, interpersonal distance. As more socially assistive robots are deployed in human environments, navigation planners should account for the uncertain human environments that demand social norms to be followed. Unlike other approaches, our proposed approach doesn't need any training data or an expert that can teach the robot how to navigate in a socially appropriate way. It is also important to note that our approach utilizes a non-linear optimization tool, PaCcET, over multiple cardinal objectives to come up with a trajectory that is best suited for the considered objectives.

ACKNOWLEDGEMENTS

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 Research (ONR, #N00014-16-1-2312, #N00014-14-1-0776). We would like to acknowledge the help of Gaetano Evangelista, Andrew Palmer, and Roya Salek Shahrezaie.

REFERENCES

[1] "Samrt luggage robot." https://thepointsguy.com/2018/01/autonmous-smart-luggage-premiere-ces/, January 2018.

[2] C. Shi, S. Satake, T. Kanda, and H. Ishiguro, "A robot that distributes flyers to pedestrians in a shopping mall," *International Journal of Social Robotics*, Nov 2017.

[3] D. Feil-Seifer and M. Matarić, "Defining socially assistive robotics," in *International Conference on Rehabilitation Robotics (ICORR)*, (Chicago, IL), pp. 465–468, June 2005.

[4] B. Mutlu and J. Forlizzi, "Robots in organizations: the role of workflow, social, and environmental factors in human-robot interaction," in *Proceedings of the International Conference on Human-Robot Interaction (HRI)*, (Amsterdam, The Netherlands), pp. 287–294, ACM, 2008.

[5] D. Feil-Seifer and M. Matarić, "Distance-based computational models for facilitating robot interaction with children," *Journal of Human-Robot Interaction*, vol. 1, pp. 55–77, July 2012.

[6] E. T. Hall, *The hidden dimension*. Doubleday & Co, 1966.

[7] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch, "Human-aware robot navigation: A survey," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1726–1743, 2013.

[8] S. B. Banisetty, M. Sebastian, and D. Feil-Seifer, "Socially-aware navigation: Action discrimination to select appropriate behavior," in *AAAI Fall Symposium Series: AI-HRI*, November 2016.

[9] M. Sebastian, S. B. Banisetty, and D. Feil-Seifer, "Socially-aware navigation planner using models of human-human interaction," in *International Symposium on Robot and Human Interactive Communication (RO-MAN)*, (Lisbon, Portugal), pp. 405–410, August 2017.

[10] G. Ferrer, A. Garrell, and A. Sanfeliu, "Robot companion: A social-force based approach with human awareness-navigation in crowded environments," in *Intelligent robots and systems (IROS), 2013 IEEE/RSJ international conference on*, pp. 1688–1694, IEEE, 2013.

[11] C. Dondrup and M. Hanheide, "Qualitative constraints for human-aware robot navigation using velocity costmaps," in *Robot and Human Interactive Communication (RO-MAN), 2016 25th IEEE International Symposium on*, pp. 586–592, IEEE, 2016.

[12] T. Kruse, A. Kirsch, H. Khambhaita, and R. Alami, "Evaluating directional cost models in navigation," in *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*, pp. 350–357, ACM, 2014.

[13] D. V. Lu, D. Hershberger, and W. D. Smart, "Layered costmaps for context-sensitive navigation," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pp. 709–715, IEEE, 2014.

[14] O. A. I. Ramírez, H. Khambhaita, R. Chatila, M. Chetouani, and R. Alami, "Robots learning how and where to approach people," in *Robot and Human Interactive Communication (RO-MAN), 2016 25th IEEE International Symposium on*, pp. 347–353, IEEE, 2016.

[15] M. Kuderer, H. Kretzschmar, and W. Burgard, "Teaching mobile robots to cooperatively navigate in populated environments," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pp. 3138–3143, IEEE, 2013.

[16] R. Ramon-Vigo, N. Perez-Higueras, F. Caballero, and L. Merino, "Transferring human navigation behaviors into a robot local planner," in *Robot and Human Interactive Communication, 2014 RO-MAN: The 23rd IEEE International Symposium on*, pp. 774–779, IEEE, 2014.

[17] B. Kim and J. Pineau, "Socially adaptive path planning in human environments using inverse reinforcement learning," *International Journal of Social Robotics*, vol. 8, no. 1, pp. 51–66, 2016.

[18] C. A. C. Coello, "A comprehensive survey of evolutionary-based multiobjective optimization techniques," *Knowledge and Information systems*, vol. 1, no. 3, pp. 269–308, 1999.

[19] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and multidisciplinary optimization*, vol. 26, no. 6, pp. 369–395, 2004.

[20] A. Messac and P. D. Hattis, "Physical programming design optimization for high speed civil transport," *Journal of aircraft*, vol. 33, no. 2, pp. 446–449, 1996.

[21] R. J. Balling, J. T. Taber, M. R. Brown, and K. Day, "Multiobjective urban planning using genetic algorithm," *Journal of urban planning and development*, vol. 125, no. 2, pp. 86–99, 1999.

[22] C. Coello and A. D. Christiansen, "Multiobjective optimization of trusses using genetic algorithms," *Computers & Structures*, vol. 75, no. 6, pp. 647–660, 2000.

[23] E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey, and K. Konolige, "The office marathon: Robust navigation in an indoor office environment," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 300–307, IEEE, 2010.

[24] L. Yliniemi and K. Tumer, "Paccet: An objective space transformation to iteratively convexify the pareto front," in *Asia-Pacific Conference on Simulated Evolution and Learning*, pp. 204–215, Springer, 2014.

[25] L. Yliniemi and K. Tumer, "Complete coverage in the multi-objective paccet framework," in *Genetic and Evolutionary Computation Conference*, 2015.

[26] M. Colby, L. Yliniemi, P. Pezzini, D. Tucker, K. M. Bryden, and K. Tumer, "Multiobjective neuroevolutionary control for a fuel cell turbine hybrid energy system," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, pp. 877–884, ACM, 2016.

[27] V. Sarfi and H. Livani, "A novel multi-objective security-constrained power management for isolated microgrids in all-electric ships," in *2017 IEEE Electric Ship Technologies Symposium (ESTS)*, pp. 148–155, Aug 2017.

[28] L. Yliniemi, "Considerations for multiagent multi-objective systems," in *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pp. 1719–1720, International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[29] V. Sarfi, H. Livani, and L. Yliniemi, "A novel multi-objective security-constrained power management for isolated microgrids in all-electric ships," in *Electric Ship Technologies Symposium (ESTS), 2017 IEEE*, pp. 148–155, IEEE, 2017.