Human Aware Path Crossing

Zhihao Liao
Sibley School of Mechanical and Aerospace Engineering
Cornell University
Ithaca, New York 14850
Email: zl673@cornell.edu

Yifan Tang
Sibley School of Mechanical and Aerospace Engineering
Cornell University
Ithaca, New York 14850
Email: yt542@cornell.edu

***Abstract*—Robots are entering our everyday life and they tend to work with humans. So, robots are required to move considering human comfort and safety. We implement a time-dependent planning algorithm for robot navigation. It uses a social cost function with predicted human motion to make a collision free and human like path. Also, it uses time dependent path planning to generate the paths considering motion constraints of robots. We compare the performance of robot in path crossing scenario using this new algorithm with that using A* path planning algorithm. The result shows that the time-dependent planning algorithm enables the robot to wait for human when human crosses robots' path.**

I. Introduction

Robot are entering our life and it is becoming increasingly common that robots share the workspace with human. So, social navigation is important for robots. The basic requirement of social navigation is collision avoidance. The existing methods generally consider humans as obstacles and make the plan assuming that humans are static. However, humans are moving continuously. Using these algorithm, robots need to replan when human's position changes. Taking path crossing scenario as an example. According to Fig 1, when humans cross robots' path, robots may first detour in front of humans before making another path behind humans, which makes the robots too close to human and makes robots' behavior confusing. In this case, humans may feel unsafe and uncomfortable when working with robots using this static planning methods. A better method is planning with prediction of human trajectories. Marina Kollmitz et al. [1] use predicted human trajectories and a novel social cost model to plan collision-free paths that consider human comfort. This approach generates a more human like path that increases the legibility of robots' behaviors and improves human's sense of safety. In this project we build a system on Sphero SPRK robot to implement this algorithm. We use Vicon motion capture system to obtain the position of human and robot. Then, we use ROS Navigation stack to generate costmap and realize navigation of the Sphero robot. Finally, we compare the performance of robot in path crossing scenario using time dependent planning algorithm and A* algorithm without replanning function.

II. Related Work

Prediction of human trajectories provides the basis for social navigation. Kuderer et al. [4] provide a method to predict human trajectories by learning a set of features that are relevant to characteristics of human trajectories in crowded environment. Marina Kollmitz et al. [1] predict human to move at constant velocity given human's current velocity and direction. Though this model is not accurate enough in a larger time frame, it is simple for testing time dependent planning algorithm. To generate human aware path, robot also needs to make a model of human. A common way is to treat humans as moving obstacles and change the costmap around humans. Thibault Kruse et al. [3] add directional cost to the costmap around human based on the compatibility of motion directions. Jorg Muller et al. [5] use a Gaussian kernel to give higher cost to people classified as obstacles. Marina Kollmitz et al. [1] also increase the cost around human based on a Gaussian kernel.

(a) Robot starts to deviate from straight line in front of human (b) Robot deviates more, becomes stalled (c) Robot replanned path by going behind human

Fig. 1: Effects of replanning with Static cost model in simulation. The robot is planning paths going from left to right, a simulated person walks from top to bottom without stopping for the robot. Taken from [6]

III. Algorithm

According to Marina Kollmitz's work, they applied time dependent, deterministic planning (A*) method to determine the relative spacial relationship in different time. In their approach, time was discretized by a constant intervals Δt. For each time expansion, the robot's current pose (x, y, ϕ) and its velocity (v, ω) was used to describe its state transition by using the state space C= (x, y, ϕ, v, ω, t). Then, a search graph is constructed according to a discrete time expansion model and the a set of executable set of actions u_i, which is a combination of the forward and angular acceleration in the action set:

$$U = \{(e_x a_x, e_\phi a_\phi) | e_x, e_\phi \in \{-1, 0, 1\}\}$$

A. Social Cost Model

The social cost model in Marina Kollmitz's paper uses the Gaussian distribution function to describe cost around human. Specifically, a two dimension Gaussian with amplitude *A* and different standard deviations σ_x and σ_y along a person's front and side. Moreover, the Gaussian function is displaced by δ_x and δ_y, considering the fact that people are more sensitive to area in front of them and people have preferences for others to pass on one side. Besides, Marina Kollmitz et al. consider a non-traversable area within a radius of r_0 for safety issues. The path through this region will be pruned during planning. After describing the static cost, the social cost model also considers the time effect. According to Marina Kollmitz's article, they use decay factors for the amplitude d_A, the variance d_σ and the forbidden obstacle radius $d_{r,0}$ for prediction uncertainties. This means the Gaussian distribution function will flatten and lower the costs of occupying the space of human as the prediction time step increases.

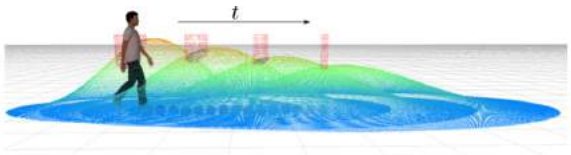

Fig. 2: Time dependent decay functions for social cost model[1]

B. Dynamic Cost Map

The cost map is consisted of a static layer and multiple dynamic layers to represent the environment constraint and time independent social constraint respectively. For time step *i*, by combining the static cost map and the *i*-th dynamic layer, we can get the corresponding cost map of predicted human trajectory between the time step *i* and *i+1*. As a result, the planning procedure is separated into a query of time steps. For each time step, a dynamic cost map is generated for the motion planning. This allows the robot to efficiently represent the time dependent navigation effort because the costmap can be looked up in a constant array. [1]

C. Optimization

Marina Kollmitz's article uses A* search algorithm to optimize the costs along the path. They consider factors including path execution time, path distance, static environment constraints and social constraints, costs of which can be obtained from the dynamic cost map. Each of these factors has a cost value c_i. All these cost lead to the resulting cost function $C = \sum w_i \cdot c_i$. The weighting factors w_i defines the priority of the factor and decide the trade off between detour or waiting time and closer proximity. This cost function C is used as $g(n)$ in A* algorithm, which will be explained later.

D. A* heuristic

The A* algorithm computes $f(n) = g(n) + h(n)$ at every node, where $g(n)$ is the cost from start point to the node and $h(n)$ is the estimated cost from the node to the goal. The A* algorithm always visits the neighbors with the smallest $f(n)$ value. To calculate $f(n)$, the heuristic function $h(n)$ is very important because it directs the exploration of searching. Marina Kollmitz's group uses a Dijkstra search algorithm to find the shortest path to the goal on the static layer of cost map, assuming that no people are present in the environment [1]. Based on this initial search result, they estimate the heuristic function for every point. Finally, they use the estimated heuristic function in A* search algorithm.

IV. System

Fig 3 shows how the whole system works and Fig 4 shows the ROS framework in detail. We use Sphero SPRK robot to do the experiment. The robot is a spherical, differential drive robot, which can be controlled by computer using ROS through a bluetooth connection. The connection between Sphero and ROS is done by using the open-source package written by Melonee Wise. We use Vicon optical motion capture system to capture human's position and velocity. Also, location of Sphero is obtained from Vicon system and sent to navigation stack. During navigation, the costmap would be updated based on the information from Vicon and be sent to navigation stack. In navigation stack, a planner will keep generating a global path from the robot's current position to the goal. The path is then sent to a path follower which generates several waypoints on the path. Waypoints are stored in a list and they divide the global path into several smaller parts for the robot to follow. Each time the robot would be guided towards the nearest waypoint in the list until it finally reaches the goal. If the nearest waypoint is still far away, the planner would start replanning. The path follower uses PD controller to generate velocity commands for the Sphero. The Velocity messages are published on topic "/cmd_vel" and Sphero subscribes to this topic. After receiving messages, Sphero would follow these commands and start moving.

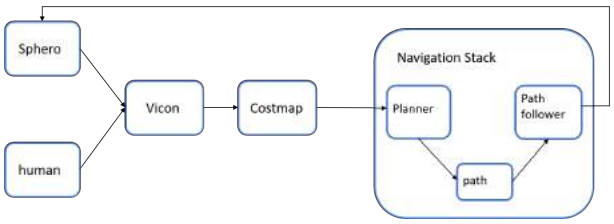

Fig. 3: System Framework

In our experiment, the Vicon system is used to get the position and the movement of the Sphero robot and human. From Fig 6, we can see that Vicon system consists of multi-cameras, each camera can capture the reflected light from reflective markers and all the cameras collaborate to generate the precise position of an object in the 3D coordinates. To make Vicon accurately recognize the robot and the human,

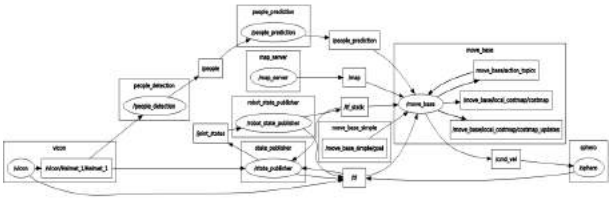

Fig. 4: ROS framework

Fig. 5: Sphero SPRK robot

we need to set a unique pattern of reflective markers for human and robot respectively. For robot, as Sphero would rotate its body while moving, it's hard for Vicon to capture the marker pattern if we put markers directly on its body. So we used a cage on the Sphero, and put the markers on the top of the cage. Vicon captures human's movement through a helmet with markers on it. Once Vicon gets the position information of the object, it would publish the information to ROS topic "Vicon/object_name/object_name". We write a subscriber to get the information and send it to tf node, which is responsible for transforming the position in world coordinate to the corresponding position in map coordinate in Rviz. We also write a state publisher which can publish the transform between the map coordinate and the robot's local coordinate, base_link. By doing this, we can synchronize the pose of the robot model in Rviz and the real Sphero.

V. Evaluation Method

We do path crossing experiment to evaluate the novel algorithm. Fig 9 shows the experiment setup. Participant and robot are supposed to go from one cone to the other on the opposite corner at the same time. Participant should begin to move when the robot begins to move. We do the experiment using time dependent planning algorithm 10 times and using A* planning algorithm without replanning function 10 times. During the experiment, we record the distance between human and robot. Then, we analyze the minimum distance in each experiment. The hypothesis is that the robot with time dependent planning algorithm would stop and wait for human to across, thus the minimum distance in experiments should be larger than experiments without replanning, in which case the robot would keep moving. As during crossing, if the distance is too small, the robot is more likely to collide with human

Fig. 6: Vicon System

(a)

(b)

Fig. 7: Cage and helmet with unique marker patterns

(a)

(b)

Fig. 8: Marker pattern on the cage and in Vicon

or make human uncomfortable. So we assume that a larger distance between human and robot during path crossing would contribute to the sense of safety of human.

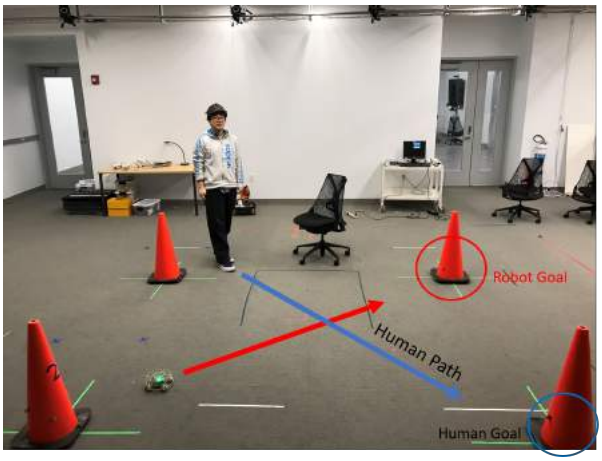

Fig. 9: Experiment Setup showing participant at start position with helmet, robot at start position in the cave, the robot path and human path crossing at 90 degree

VI. Result

The video of experiments is available at https://youtu.be/Nue245JfY4A. We can see from the video that the replanning algorithm enables Sphero to stop for a second while the human is passing, starts to replan and drives the Sphero to its goal after human crosses its path. By comparison, without replanning algorithm, Sphero goes directly to its goal and gets too close to human. It would even run into the human, which would be very dangerous if it is a large robot. Also, through Rviz, we can see that robot is replanning its path when human crosses. Since the human is in the way, the planner could not find a good path to the goal. Consequently, the robot would stay in the same place and keep replanning. Once the human passed, the planner would then update a new plan for Sphero and the robot would start to move. This shows that our algorithm successfully works.

Then, we analyze the minimum distance between human and robot in experiments. The data is displayed in table I and Fig 10. We do independent T-test on these two cases. On average, replanning algorithm increases minimum distance between human and robot($M = 1.02$,$SE = 0.0126$) , than from A* algorithm without replanning ($M = 0.314$,$SE = 0.0282$). This difference was extremely signicant, t(10) = 7.62, $p \leq 0.01$. According to the data, we can find that with replanning algorithm, the minimum distance is explicitly larger than that without replanning. This result further agrees with our hypothesis and proves that our algorithm works. Besides, the variance of non-replanning algorithm (0.072) is larger than that of replanning algorithm(0.014). This is because the robot would stop when it finds that human is crossing the path, so the minimum distance is more stable in replanning algorithm. While without replanning, the robot would go straight towards the goal and won't stop even the human are in the way. As in each experiment the velocities of human and robot are different, the minimum distance would also vary, which explains the larger variance of the data.

Without Replanning / m	With Replanning / m
0.144614	0.913548
0.276781	0.879347
0.641188	0.924658
0.211542	1.206415
0.12311	1.058704
0.166227	0.966038
0.046614	1.043437
0.047888	1.179883
0.750295	1.162561
0.736804	0.886849

TABLE I: Minimum Human-Robot Distances by using and not using replanning algorithm

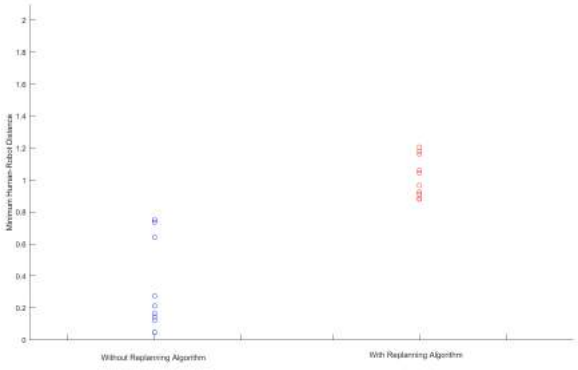

Fig. 10: Minimum Human-Robot Distances by using and not using replanning algorithm

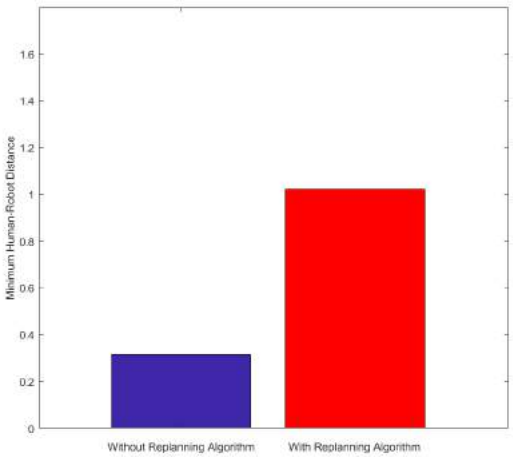

Fig. 11: The mean of Minimum Human-Robot Distances by using and not using replanning algorithm

VII. Conclusion

We have built a system to test time dependent algorithm proposed by Marina Kollmitz et al. [1] To test the algorithm, we did path crossing experiments using the time dependent planning algorithm in Marina Kollmitz's work, compared with A* planning algorithm without replanning function. By comparing the minimum distance between human and robot, we find that time dependent algorithm generates a larger average distance while the robot are closer to human and sometimes even collide with human during path crossing using A* algorithm without replanning function. The experiment proves that time dependent algorithm is better and produces a more human like path considering human comfort and safety.

There are some limitations of our work. Due to controller accuracy of Sphero, it is hard to tune a good path follower by changing the proportional gain and differential gain of PD controller. Therefore, in experiment, Sphero does not follow the generated path perfectly. Also, tuning the threshold of replanning criteria is hard. Without an excellent replanning criteria, the robot will try replan frequently. So, it takes long time for the robot to obtain a new plausible plan and the robot will stay on the same place during replanning time. Therefore, it is difficult to know whether the robot choose to stay and wait for human or the robot cannot find a path. The performance of the algorithm can be more obvious with better parameters and better robot platform. In the future, the prediction of human

trajectories could be improved by taking human's orientation and goal into consideration. Finally, we hope that we can ask a group of Cornell students to participate in the experiment and do a within subject study to understand humans' sense of safety for robots with replanning algorithm and A* algorithm without replanning.

ACKNOWLEDGMENT

We'd like to acknowledge and thank Professor Hoffman for his guidance in this project, and thank Professor Kress-Gazit and her PhD student Ji Chen for the use of Autonomous Systems Lab resources and support for Vikon system to conduct our experiment.

REFERENCES

[1] M. Kollmitz, K. Hsiao, J. Gaa et al, "Time dependent planning on a layered social cost map for human-aware robot navigation," *2015 European Conference on Mobile Robots (ECMR)*, Lincoln, 2015, pp. 1-6.

[2] T. Kruse, P. Basili, S. Glasauer et al, "Legible robot navigation in the proximity of moving humans," *2012 IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO)*, Munich, 2012, pp. 83-88.

[3] T. Kruse, A. Kirsch, H. Khambhaita et al, "Evaluating Directional Cost Models in Navigation," *2014 9th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, Bielefeld, Germany, 2014, pp. 350-357.

[4] Kuderer, M., Kretzschmar, H., Sprunk, C., Burgard, W. "Feature-Based Prediction of Trajectories for Socially Compliant Navigation," *Robotics: science and systems*, 2012.

[5] Mller, J., Stachniss, C., Arras, K. O., Burgard, W. "Socially inspired motion planning for mobile robots in populated environments," *Proc. of International Conference on Cognitive Systems*, 2008.

[6] Kruse, T., Basili, P., Glasauer, S., Kirsch, A. (2012, May). Legible robot navigation in the proximity of moving humans. In Advanced Robotics and its Social Impacts (ARSO), 2012 IEEE Workshop on (pp. 83-88). IEEE.