Human-Aware Navigation Planner for Diverse Human-Robot Interaction Contexts

Phani Teja Singamaneni[1], Anthony Favier[1,2], Rachid Alami[1,2]

***Abstract*— As more robots are being deployed into human environments, a human-aware navigation planner needs to handle multiple contexts that occur in indoor and outdoor environments. In this paper, we propose a tunable human-aware robot navigation planner that can handle a variety of human-robot contexts. We present the architecture of the system and discuss the features along with some implementation details. Then we present a detailed analysis of various simulated human-robot contexts using the proposed planner. Further, we show that our system performs better when compared with an exiting human-aware planner in various contexts. Finally, we show the results in a real-world scenario after deploying our system on a real robot.**

I. Introduction

In the recent decade, more and more robots are entering into human environments. From the robotic vacuum cleaners to the human assisting robots in shops, malls [1], [2] and airports [3], all of these robots are working in environments with humans moving around. To navigate in these places, the robot needs to be aware of the humans in the environment, and treating humans simply as obstacles may not be enough. Besides, the robot's motion should be safe, legible and acceptable to humans rather than being optimal from the sole point of view of the robot (time, energy etc.). Therefore, a new field of robotic navigation called human-aware (or social) navigation has emerged, which studies various human motion and social aspects for developing more human acceptable robotic navigation.

Depending on the shape of the local environment (large area, small rooms, narrow corridors or passage) and the density and activity of the humans (individuals, crowds, domestic or public space motion activity) in that environment, human-aware robot navigation has to address various types of human-robot interaction contexts. These contexts can differ from the situations where the current position of the human is enough to where a good estimate of the goal is necessary or even where path negotiation has to take place. For instance, if the robot is in the middle of a dense crowd, it should better be purely reactive and compliant to the overall motion flows than in a corridor where less reactive and more cooperative motion with path negotiation is preferable. Therefore, different navigation planners are developed for different types of environments with shared human spaces like malls [2], streets [4], warehouses [5], offices [6], labs, homes [7] etc. All these different planners emerged as there is no single algorithm that can cover all environments and situations. In order to address these issues, we propose a highly tunable human-aware navigation system with multiple modes of planning that can be employed in a variety of human-robot contexts, with a small number of pertinent parameters that can be adjusted to treat various contexts. Our main contributions in this work are threefold:

1) We propose a tunable human-aware navigation planner with different planning modes that can handle very complex indoor scenarios as well as crowded scenarios called the Cooperative Human-Aware Navigation (Co-HAN) Planner.
2) We extend our previous work, Human-Aware Timed Elastic Band (HATEB) [8] local planner, to effectively handle large numbers of people and to offer more legible and acceptable navigation.
3) We evaluate the proposed planner in several simulated human-robot scenarios and present both qualitative and quantitative analysis. Further, we also present the tests conducted on the real robot at our lab.

The rest of the paper is organized as follows. In section II, we discuss the related works. The planner's architecture is presented in section III along with explanations of various modules and features. Following this, section IV presents the evaluation of our planner in various simulated human contexts. We also present a comparison with one of the existing human-aware navigation planners. In section V, we talk about the tests conducted on the real robot. Finally, section VI presents some discussion and conclude.

II. Related Work

There are a variety of human-aware navigation planners designed for different human-robot contexts. In the context of a crowd or robot navigation in the street, Ferrer. et al [4] presents a potential field based navigation using the social force model. The authors of [9] extended this to human-object and human group interactions by proposing the proactive social motion model. The work by Repiso et. al [10] shows the context of a robot accompanying a human. The authors of [11] address this crowd navigation problem by using reinforcement learning and, the works [3], [12] address the same with inverse reinforcement learning. Coming to other contexts, the works presented in [13], [6] and [7] show some interesting costmap based approaches for planning paths in complex indoor scenarios that can occur at homes or offices. In this paper, we use a similar costmap based approach to handle static humans. Fernandez Carmona et. al [5] compares the performance of the existing navigation

[1]Authors are with LAAS-CNRS, Universite de Toulouse, CNRS, Toulouse, France, {ptsingaman, anthony.favier, rachid.alami}@laas.fr

[2]This work has been partially funded by the Agence Nationale de la Recherche through the ANITI ANR-19-PI3A-0004 grant

planners in a warehouse context and proposes an architecture to include humans in planning. The work of Güldenring et. al [14] addresses the same context using reinforcement learning. Some other works like [15], [16] use inverse reinforcement learning for confined and public space navigation contexts. Khambhaita and Alami [17] addressed the context of human-robot co-navigation based on an optimization-based approach. Note that none of the above planners was designed to handle multiple human contexts together. A multi-context human-aware navigation planning is a very new field, and not many works exist. Lu et. al [18] proposed a layered costmap based approach for handling different navigation contexts. A more recent work by Banisetty et. al [19] shows some promising results using a deep learning-based context classification and multi-objective optimization based navigation planner [20]. However, these results are validated only in indoor scenarios and, the authors do not present any results in a crowd, unlike the proposed system.

In order to handle the dynamic humans in our navigation planner and plan a socially acceptable trajectory for the robot, a human motion prediction system is required. One of the classic approaches of human motion prediction is based on the social force model [21]. Ferrer et. al [22] uses this social force model both to predict human motions and to move the robot among the crowds. Kollmitz et. al [7] uses a simple linear prediction based on current human velocity. Instead of predicting the trajectory, a possible human goal can also be predicted using certain reasoning over a probable set of goals [23]. Our proposed navigation system uses one such goal prediction system [24] as a part of the human path prediction module. Apart from this, our system offers three other human path prediction methods to handle different situations. In a recent work by Fisac et. al [25], the authors suggest a probabilistic human model with confidence to handle the uncertainties in a system.

One of the key elements of the proposed system is the context-based shifting between different planning modes. This kind of modality shifting is discussed in the works of Qian et. al [26], and Mehta et. al [27] based on Partially Observable Markov Decision Processes. Unlike these, our system uses situation assessment based modality shifting. In our previous work [8], we introduced this modality shifting with three different modes of planning. In the current work, we extend this to handle a large number of humans and also introduce some elements, including a new planning mode. This modified *HATEB local planner* is integrated into the proposed framework as the local planner.

III. Planner Architecture and Features

In this section, we present the overall architecture of the human-aware navigation planning system and explain its features that allow us to deal with various kinds of human-robot contexts. Our system is developed over the ROS [28] navigation stack, and the architecture of the proposed system is shown in Fig. 1. The red blocks shown in the Fig. 1 are the modifications we introduced into the standard ROS navigation stack and are the major contributions of this

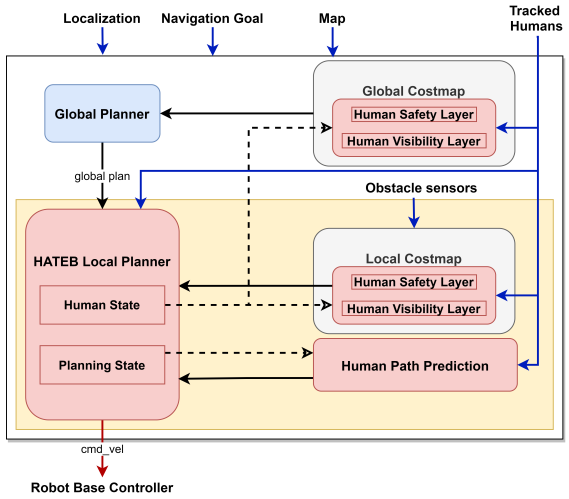


Fig. 1: Software architecture of the proposed planner.

work. As shown in the figure, we introduce *Human Safety* and *Human Visibility* costmaps layers into both global and local costmaps. The *Human Safety* layer is modelled as a 2D Gaussian around the human, and the *Human Visibility* layer as a 2D half Gaussian on the backside of the human. Both these layers have a cutoff radius of $3m$ [13] beyond which the cost is zero. These layers are implemented using a costmap plugin that we developed called the *human_layers*. The addition of these layers is controlled by the *Human State* of the *HATEB local planner*. The *Human Path Prediction* module predicts the possible paths for the requested humans using the selected prediction method. The *HATEB local planner* module accesses different human-robot scenarios and determines the *Human State* and the *Planning State* shown in the figure. Both these states together decide the planning mode of the system and also control the transition between different modes. Based on the *Planning State*, the appropriate path prediction method is selected for the humans. After accepting a navigation goal, the system continuously accesses the human-robot scenario and appropriately chooses a planning mode that decides the command velocity sent to the robot's base controller. Note that the planning mode need not be constant and can shift depending on the context. Further, our system is completely tunable, and the transition between different modes can be tuned (or changed) by changing the mode transition parameters [8] as per the requirement. We call this entire system together with all modules as CoHAN planner. The system is publicly available on github at `https://github.com/sphanit/CoHAN_Planner`.

A. Types of Humans and Costmap layers

In our system, we deal with different types of humans while navigating the robot to the goal. Fig. 2 shows all these types of humans along with the robot's visibility and the planning radius, R. While the robot is moving in the environment, the system considers only the humans within this planning radius that are in the visible region. Among these humans, it checks for the static and dynamic

humans and updates the *Human State* for all the humans. The *human_layers* plugin checks the *Human State* of all the observable humans and adds the *Human Safety* and *Human Visibility* costmap layers around the static humans. We chose to add these costmap layers only around static humans because the response time for static humans is usually slow, and the robot should therefore maintain a larger safety distance as well as avoid surprise appearances from behind [13]. Besides, no elastic band is added to the static

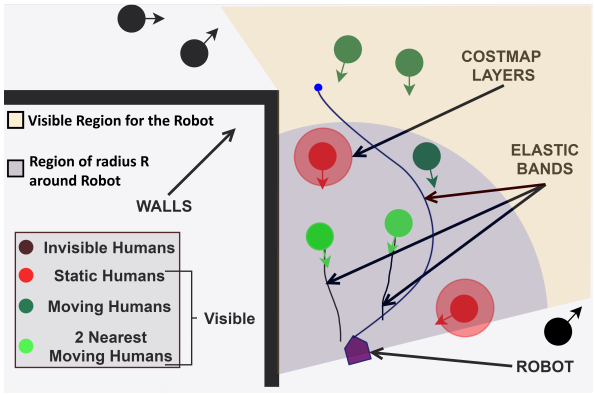


Fig. 2: Different types of humans considered in our system. A sample trajectory of the robot is shown among different humans.

humans, and the addition of these layers is necessary to plan a safe path. For the dynamic humans within the planning radius, the system adds elastic bands to the two nearest humans and plan their paths and trajectories until they move behind the robot or out of the planning radius.

B. Human Path Prediction

The *Human Path Prediction* module deals with different kinds of human goal predictions and building the global plans for required humans. Our system currently offers four types of human goal prediction and path planning methods.

1) *PredictBehind*: This method predicts that the human goal is behind the robot. The position of the robot when the human enters the visible planning radius is used for this. This goal is used to predict the path.
2) *PredictGoal*: This method predicts the most probable goal among the set of goals provided to the system using the approach described in [24]. The predicted goal is then used for path prediction.
3) *PredictVelObs*: This method builds a path by extrapolating the current human velocity over a fixed duration and does not predict any goal. Currently, the duration is set to $5s$. This is the default prediction service in *VelObs* mode.
4) *PredictExternal*: This service accepts a goal from an external system and adds a global path prediction based on the provided goal.

These services provide global plans for the humans that are used by the *HATEB local planner* for planning local plans (or trajectories).

C. HATEB local planner and planning modes

It is the core module of the proposed human-aware navigation planning system. *HATEB local planner* is based on the human-aware extension of Timed Elastic Band (TEB) [29] local planner by Khambhaita and Alami [17]. This module plans the robot's trajectory and the possible human trajectories for the two nearest humans in the visible planning radius based on the predicted human paths. It continuously assesses the current human-robot context and sets the *Planning State* and the *Human State*. Depending on the value of these states, it shifts between different planning modes. This is needed in the intricate human-robot contexts that cannot be solved using a single planning mode.

Modes of Planning: *HATEB local planner* provides four different modes of planning at the control level and intelligently shifts between them based on the situation.

1. *SingleBand mode*: This is the mode in which the planning system starts and has an elastic band only for the robot. The system stays in this mode as long as there are no humans within the visible planning radius. The default planning radius is taken as $10m$.
2. *DualBand mode*: In this mode, elastic bands are added to the two nearest dynamic humans in visible planning radius and trajectories are planned for humans along with the robot. This kind of human planning allows the robot to proactively plan its trajectory and adapt to the changing human plans. On top of being useful as predictions, these planned trajectories also offer a possible solution for the human-robot context, which, if followed, will resolve any conflict that exists.
3. *VelObs mode*: This mode uses all the human-aware criteria while planning but adds bands to humans only if they have some velocity. This mode is useful in crowded human scenarios or when the robot cannot move due to entanglement issues [8] of the *DualBand* mode.

The situation assessment and mode shifting scheme for the above-mentioned planning modes is described clearly in our previous work [8]. In this work, we extended this by adding a *Backoff-recovery* mode.

4. *Backoff-recovery mode*: The *Backoff-recovery* mode is required when there is no solution to the planning problem unless one of the agents completely clears the way for the other. This kind of situation commonly occurs in a very narrow corridor where only the person (or robot) can navigate at a time. If a human and robot face each other in a very narrow corridor or another situation where one of them has to clear the way for the other, our system gives priority to the human and makes the robot clear way for the human. This is done by making the robot move back slowly until it can go either left or right to clear the way (by querying the costmaps on all three sides and setting a temporary goal in the possible direction). Once the robot clears the way, it waits for the corresponding human to complete its navigation or a timeout and then proceeds to its goal. This mode is activated when the robot is in *VelObs* mode in the near vicinity of the human ($< 2.5m$), and it is stuck without progressing towards the goal for more than 10 seconds. It can also accept new

goals in the waiting state and navigate to them, discarding the existing goal.

HATEB uses several human-aware constraints in its optimization scheme for proactive and legible planning around humans in the environment. Several of these constraints are listed in our previous works. In this work, we have added two new constraints along with the previous constraints. These new constraints, called the *Visibility* and *Relative Velocity* are explained briefly below:

Visibility: It adds cost to the optimization when the robot is behind the human and it plans to cross or go in front of the human. This constraint tries to avoid the emergence of the robot suddenly from behind and makes the robot enter the human's field of view from a larger distance. It is implemented by adding a 2D half Gaussian behind the human.

Relative Velocity: This constraint adds cost to the optimization based on the relative velocity between humans and the robot and their distance. The main effect of this constraint is low robot velocity in the close vicinity [30] of the human if it cannot find a path to maintain a greater distance. If the robot can find a path with a greater distance from a human, it chooses that path with a normal velocity profile. Another effect of this constraint is early intention show of the robot similar to *TTC* or *TTCplus* constraint given in [8]. The cost added is shown below.

$$cost_{rel\_vel} = \frac{((max(\overrightarrow{V_{rel}} \cdot \overrightarrow{P_r P_h}), 0) + \|\overrightarrow{V_r}\| + 1)}{\|\overrightarrow{P_r P_h}\|} \quad (1)$$

where $\overrightarrow{P_r}, \overrightarrow{V_r}$ are the position and velocity of the robot, $\overrightarrow{P_h}, \overrightarrow{V_h}$ are the position and velocity of the human and $\overrightarrow{V_{rel}} = \overrightarrow{V_r} - \overrightarrow{V_h}$. Since this constraint has similar effects as *TTC* or *TTCplus*, we activate this constraint alone and deactivate *TTC* and *TTCplus* constraints in all the experiments presented in this paper. *HATEB* takes all the activated human-robot constraints and other necessary kinodynamic constraints and plans the trajectories of the robot and the humans. Since the local planner runs a computationally expensive optimization in each control loop, extending the planning beyond two humans does not yield real-time control of the robot. Hence we restricted our human planning to the two nearest humans.

One more extension in this work is the addition of the field of view of the robot into the planning system. This is done using ray-tracing from the robot's position on the environmental map. Since human tracking is provided by an external system, it is important to restrict the system to consider the humans present in its field of view. This is more natural and makes it easier to use our system with vision-based human tracking. The proposed system also offers a variety of parameter settings that can choose prediction mode, the human-aware constraints to be used and tuning over these costs. Even the planning can be restricted to only one of the three planning modes (except *Backoff-recovery*). Hence, it is possible to extend it to many kinds of human-robot contexts by properly choosing the parameters and with simple tuning. With the addition of the costmap layers around the static humans, this framework can handle all the scenarios presented in [19]. We present our results in different scenarios and environments in the next section.

IV. Results in various Simulated Scenarios

To validate our system, we applied it to various kinds of human-robot contexts that can occur in day-to-day life. These situations are generated in a simulated environment based on MORSE [31]. The humans in these simulations are controlled in three different ways to test the robustness of the system: (1) Joystick based control by a human operator, (2) Using an improved human motion simulator we have developed in our lab and (3) Using the human trajectories generated by *HATEB local planner* (an ideal situation where the human moves as expected by the robot). We present in detail some of these intricate scenarios in this section, along with some quantitative results. Further, we also present some details about the extension of our system to crowded scenarios using PedSim ROS[1]. In all figures shown below, the trajectories of the robot and humans (if shown) are shown as coloured dots. These are the poses planned by *HATEB local planner*, and the colour visualizes the time. If the colour of the predicted human pose dot is the same as the colour of the robot pose dot, they will both be at that location at the same time.

A. Door Crossing Scenario

Door crossing is a common situation in many human environments. If two humans try to pass through the same door, one of them has to compromise and clear the way for the other. We have placed the robot running our planning system in the door crossing situation shown in Fig. 3. The goal of the robot is beside the second human standing in the room, and the system uses *PredictBehind* human path prediction. The left part of the figure shows the simulated scenario and the corresponding trajectories planned for the human and the robot. The simulated human crossing the door was controlled using a joystick and, hence does not move as the planning system expects. The system quickly adapts to these changes and makes the robot clear the way for the human by waiting on the side, as shown in the right part of Fig. 3. The robot continues to its goal after the human crosses the robot. The planning mode is *DualBand* until the human crosses the robot, and then it switches to *SingleBand*. As soon as the robot crosses the door, it faces one more human, but this human is just standing at the same place and does not move. Since the human is static, our system adds the *human_layers* to the costmaps and re-plans its path. The same scenario is repeated with the second human placed in two different orientations and as shown in Fig. 4. In both scenarios, there is enough space between the wall and the human for the robot to reach its goal, maintaining a safe distance from the human. On the top left scenario of the figure, the human can see the robot, and so the planner makes the robot pass through this space. However, in the

[1] https://github.com/srl-freiburg/pedsim_ros

Fig. 3: Door crossing scenario in the simulated environment. The human moves towards the door. The robot sees the human and waits on the side of the door (right) until the human crosses.

second scenario, the human cannot see the robot. Therefore, our planner completely re-plans the path as shown (top right) and makes the robot reach its goal by taking a longer path making the robot visible to the human. It is due to the added *Human Visibility* layer. Fig. 4 also shows the plots

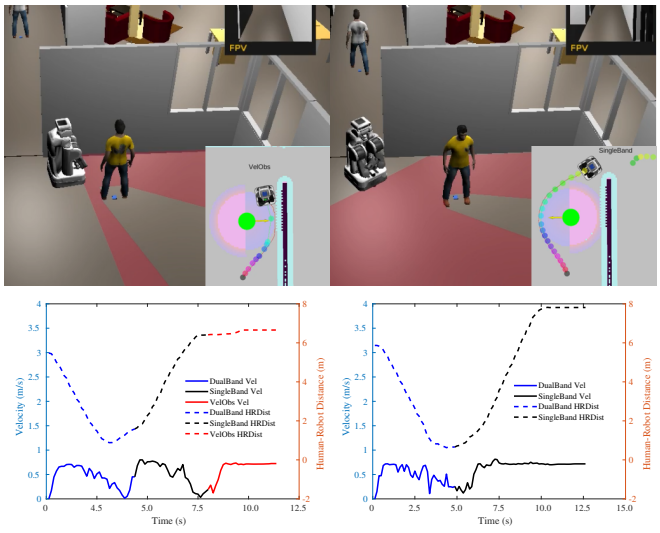


Fig. 4: Door crossing scenario in the simulated environment with the static human in two different orientations. Top two pictures show the scenarios in simulation and the planned trajectory of the robot. The bottom two figures show the robot velocity and human-robot distance graphs over time.

corresponding to robot velocity (on the left y-axis) and the distance between the moving human and the robot (human-robot distance) (on the right y-axis) with respect to the time (on the x-axis). Different colours in different portions of the plots correspond to different planning modes of the system, as indicated in the plots. The solid line represents the robot's velocity (Vel), while the dashed line shows the human-robot distance (HRDist). The same conventions are followed across this paper. From both the graphs (Fig. 4 bottom left and right), it can be observed that the robot's velocity decreases as the human-robot distance decreases. It is a combined effect of several human-aware constraints of our system. However, the *Relative Velocity* constraint plays a major role here. Secondly, it can be seen in the graph of the first scenario (bottom left) that the robot's velocity decreases one more time before the planner changes to *VelObs* mode. It is because the robot is trying to navigate a narrow space between the human and the wall. This causes the planner to slow down its velocity and checks the state of the human. Since the human is static, it shifts to *VelObs* mode that has little reduced constraints and continues its navigation.

B. Narrow Corridor Scenario

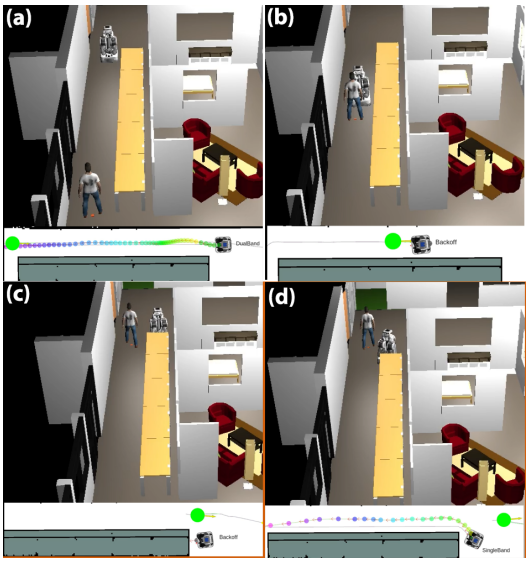


Fig. 5: Narrow corridor scenario simulated in MORSE. (a) The initial planned trajectory of the robot in *DualBand* mode. (b) The robot's way is blocked by the human and the system shits to the *Backoff-recovery* mode. (c) The robot clears the way for the human and waits on the side until human crosses the robot. (d) The robot continues to its goal in *SingleBand* mode.

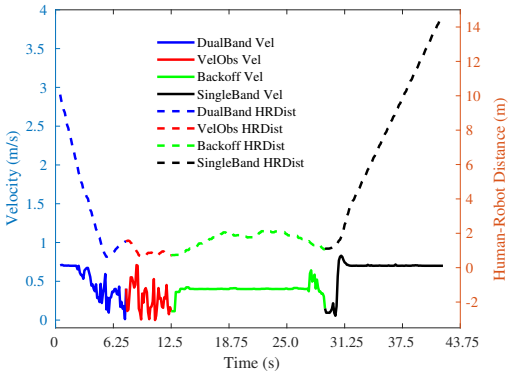
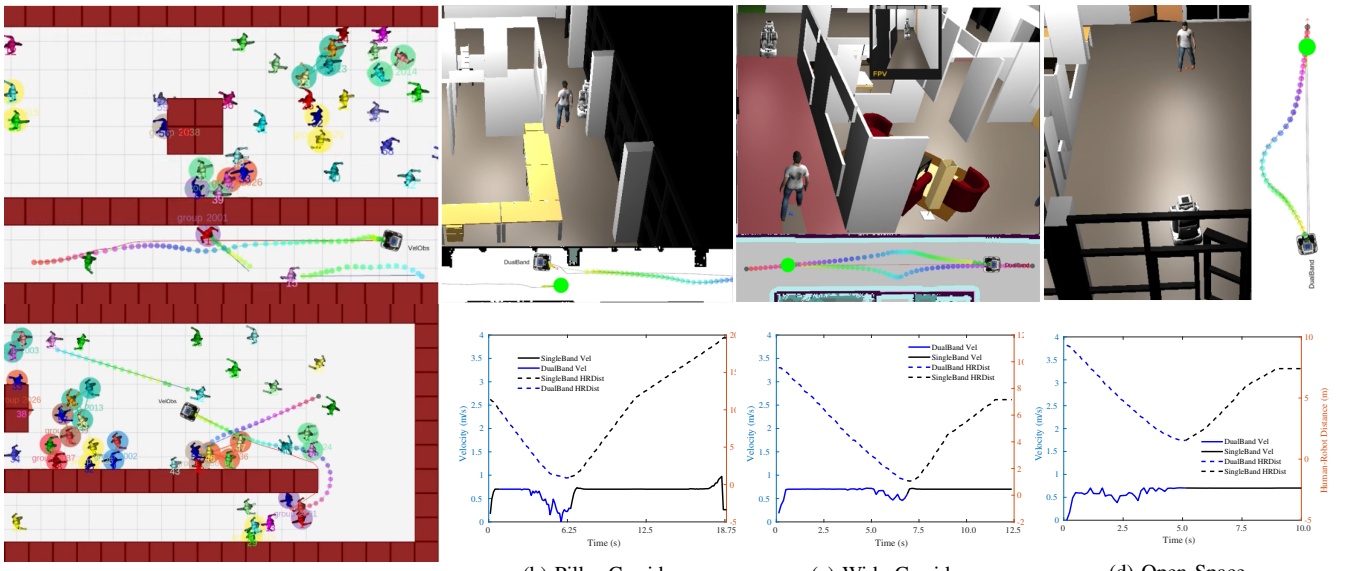


Fig. 6: Plots of velocity and human-robot distance over time in the *Narrow Corridor* scenario.

This scenario occurs when a long corridor has to be traversed by two humans in opposite directions, and the corridor is wide enough only for a single human. In this case, one of them has to go back and wait for the other to cross. When one of the agents in this scenario is a robot, it becomes a little more complicated as the robot should back-off giving priority to the human while taking legible actions. Most of the existing planners either re-plan a long deviation to reach the goal or fail in this complicated situation. A more natural way to handle this would be to clear the way for



(a) PedSim (b) Pillar Corridor (c) Wide Corridor (d) Open Space

Fig. 7: (a) The robot running the proposed navigation system in the PedSim ROS pedestrian simulator. The robots planned trajectory and the predicted trajectories of the two nearest humans in *VelObs* mode are shown. (b) A corridor with pillars, wide enough for only one agent at the side of the pillar. (c) A wide corridor where the two agents have enough space to cross each other maintaining safe distances. (d) An open space scenario where the robot has enough space to avoid and show its intention to the human well in advance. In (b), (c) and (d), the plots of robot's velocity and human-robot distance over time are shown below the scenario.

the human and wait until the human crosses the robot to resume its goal. The *Backoff-recovery* mode of our system does exactly this. To make the actions more legible, the robot moves back slowly without showing its back until it can go either left or right to clear the way. The snapshots from the simulated version of this scenario are shown in Fig. 5. Each picture also shows the planned trajectory of the robot in each setting with the planning mode shown behind the robot. This scenario uses the *PredictGoal* human path prediction, and the goal of the robot is on the other side of the corridor. Fig. 5 (a) shows the initial situation when the two agents are entering the narrow corridor. As the robot can see the human is moving, it operates in *DualBand* mode until the human blocks it's way completely. The human agent in this setting is controlled by the human simulator mentioned earlier. As soon as the robot finds itself blocked, it switches to *VelObs* mode and checks for a possible solution. However, when it cannot find the solution after repeated checks, it switches to *Backoff-recovery* mode after few seconds ($> 10s$) as shown in Fig. 5 (b). Fig. 5 (c) shows the robot waiting for the human to cross the corridor before it can resume its goal. The robot switches to *SingleBand* mode and resumes its navigation to the goal as in Fig. 5 (d).

The plots of robot velocity and human-robot distance with respect to time for this scenario is shown in Fig. 6. As the human-robot distance decreases after a certain threshold, the velocity of the robot decreases like in the door crossing scenario (blue part). When the robot switches its mode from *DualBand* to *VelObs*, the robot tries to move in different directions causing the oscillations seen in the plot (red part). In the *Backoff-recovery* mode, it maintains a constant velocity (green part) and stops waiting for the human. The human agent of the human simulator starts moving towards the robot as soon as it starts moving back. This explains a near-constant human-robot distance trend in green. Once the human passes the robot, it resumes its navigation in *SingleBand* mode (black part).

C. Results in the Crowd context and other scenarios

We further tested our system in various scenarios, including crowds. For the simulation of crowds, we have used the PedSim ROS simulator and use the system in *VelObs* mode. Fig. 7 (a) shows two snapshots from the tests. The robot adds elastic bands to two of the nearest humans in the environment and successfully navigates the crowd generated by the simulator (shown in video). Further, it is seen that the robot proactively clears the way for PedSim agents while navigating in the corridor shown, Fig. 7 (a).

We have simulated three other scenarios in MORSE: (1) Pillar Corridor, (2) Wide Corridor and (3) Open Space, and used our system to navigate them. These scenarios are shown in Fig. 7 (b), (c) and (d). In all three scenarios, human and robot goals are behind the other agent. In the Wide Corridor scenario, the robot predicts that the human's goal is behind its initial position and plans the human's trajectory. We use this planned human trajectory to control the human in this case, and hence it represents the ideal scenario for the planner. The scenario and its corresponding velocity and human-robot distance plots are shown in Fig. 7 (c). For the other two scenarios, the human agent was controlled by the human simulator, and the system uses *PredictGoal* human path prediction. The velocity and human-robot distance plots for the Open Space (7 (b)) scenario are similar to the Wide Corridor one (7 (d)), however, in the Pillar Corridor (7 (c)) scenario, the velocity almost reaches zero when the human

Experiment	HATEB			TPF		
	Path Length (m)	*Total Time* (s)	*Min HRDist* (m)	*Path Length* (m)	*Total Time* (s)	*Min HRDist* (m)
Open Space	9.2329	**16.0117**	**1.2927**	**8.7892**	24.7683	0.9508
Narrow Passage	9.7279	**17.8017**	0.7061	**9.4017**	27.1633	**0.935**
Pillar Corridor	19.1127	**31.46**	**0.8948**	**18.4489**	52.9417	0.7589
Narrow Corridor	**23.5369**	**48.3850**	**0.6636**	-	-	-

TABLE I: Mean values of the metrics over 10 repetitions in four different contexts. TPF failed in the Narrow Corridor case.

is at the closest. This occurs as the robot has to wait behind the pillar for the human to cross.

D. Quantitative Results

In order to check the repeatability of our system and evaluate its performance with respect to the existing human-aware navigation planners, we have selected four different simulated scenarios and repeated the same experiment 10 times in each of the scenarios. The scenarios we considered here include Open Space, Narrow Passage (similar to the one in [8]), Pillar Corridor and Narrow Corridor. The human in these scenarios is controlled by the improved human simulator mentioned earlier. In all these scenarios, our system produced consistent results over the repetitions with similar paths. Further, we compared it with the human-aware navigation planner presented in [7] that was designed for indoor home context. This system uses *Timed Path Follower* (TPF) as its local planner, and its trajectory is highly dependant on the path produced by its global planner, *Lattice Planner*. Note that our comparison was limited to only one planner as not many human-aware planners are available openly for indoor contexts. This planner was also made to run repeatedly 10 times in the above four scenarios. However, in the Narrow Corridor case, this planner failed to complete the navigation, and the robot got stuck in front of the human as the *Lattice Planner* could not find a path.

We used three different metrics to present the comparisons between these two planners, the total length of the path taken, the total time to complete the scenario and the minimum human to robot distance that the planner encountered while executing each scenario. The average over the 10 runs is taken and presented in Table. I. Note that the total time taken by the TPF is greater as its linear velocity was limited by the planner, even though the same velocity and acceleration limits of the robot were provided to both the planners. In terms of the total path length, TPF always followed a lesser distance compared to *HATEB*. This is because *HATEB* took the larger deviations to either show intentions (in Open Space) or a clear path for the human (Pillar Corridor and Narrow Passage). However, in the Narrow Passage case, TPF produced better behaviour by waiting for the human to cross, while *HATEB* blocked the way for a bit before clearing the way for the human. This can also be seen by comparing the minimum human-robot distance in this case. Finally, in terms of the minimum human-robot distances, *HATEB* varies widely, as it handles each case differently. If the space is available, it takes a greater distance than TPF otherwise, it slows down the robot's velocity and approaches a little closer to the human. In TPF, this metric produces similar results in two of the three scenarios. In the Pillar Corridor, this metric has a lesser value compared to *HATEB* as the robot goes towards the wall opposite to the pillar and waits instead of going behind the pillar.

V. Experiments with Real Humans

We deployed our system on a real robot platform, Pepper[2] for some real-world experiments in our lab. For the human detection and tracking, we used the OptiTrack[3] motion capture system that publishes the positions and velocities of the tracked humans at 10 Hz. We present results from two experiments, one in open space and the other in the narrow corridor. Fig. 8 shows the instances from these experiments.

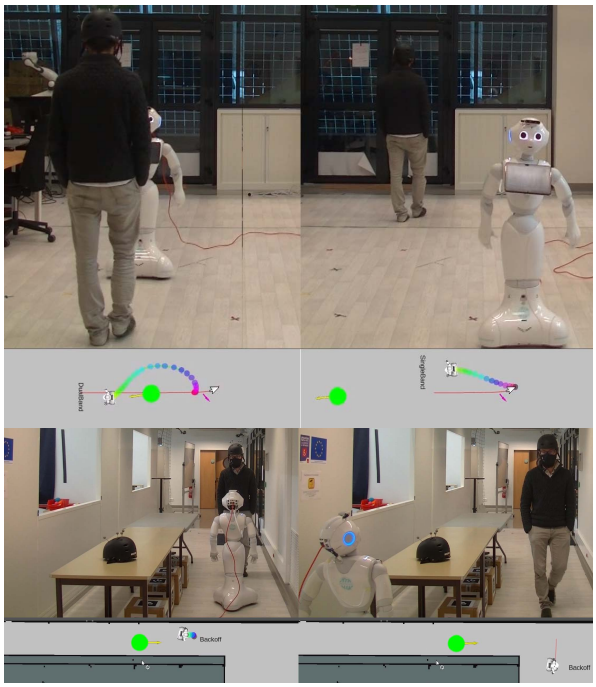


Fig. 8: Our system deployed and tested on a real robot, Pepper. The top pictures show the open space scenario, and the bottom pictures show the test of the *Backoff-recovery* mode.

The pictures on the top show the open space scenario with the planned trajectory of the robot shown at the bottom of each instance. In this scenario, the human approaches very close to the robot at the crossing point, and this makes the robot slow down, back off a little and then re-plan its trajectory (please see the video[4] attachment). The bottom part of Fig. 8 shows instants from the test of *Backoff-recovery* mode in

[2]https://www.ald.softbankrobotics.com/en/pepper
[3]http://www.optitrack.com/
[4]Video Link: https://youtu.be/DB_8HpjngJ4

our system. The human stands in the corridor blocking the robot's way. The planner starts in *DualBand* mode and finally switches to *Backoff-recovery* mode as there is no solution. The robot slowly backs off and clears the way for the human by moving to the left side of the corridor, as seen in Fig. 8.

VI. Discussion and Conclusion

In this work, we proposed a new human-aware navigation planner that can handle a variety of human-robot contexts. It was able to handle both outdoor crowd scenarios and indoor intricate scenarios, thanks to the different planning modes and tunable parameters in the system. These planning modes are at the control level and hence differs from higher-level modes as used in [27]. Consider the *Backoff-recovery* mode for example, instead of going into the corridor, the robot could have stopped (or back-off) as soon as it sees a person, or if the robot has progressed more than the human, the human can go back and let the robot go. By employing a higher level planner over our system, this case can be handled much more efficiently, but our focus is on providing this feature at the control level. We introduced *Human Safety* and *Human Visibility* layers into the system through costmaps to address the static human scenarios. For handling the dynamic humans, we have used a variety of human-aware constraints in *HATEB* along with visibility and planning radius. The proposed system also provided different types of human path prediction methods. We introduced two new human-aware constraints in addition to the previous ones present in *HATEB* to offer a more legible trajectory. Further, we evaluated our system in a variety of simulated scenarios and presented both qualitative and quantitative results. Finally, the real-world tests on a robotic platform were presented.

Limitations and Future Work: One of the major limitations of our system could be computational complexity as it performs optimization in each control loop. However, it does not affect the real-time performance of the robot in *SingleBand* and *Backoff-recovery* modes (10Hz). In the other modes, it may lead to a little reduced control rate (8-9 Hz), however still in real-time. One of the immediate future works is to develop a higher-level planner on top of our system to handle the contexts more efficiently and to include more contexts like following or accompanying a human. Currently, the system is not designed for handling groups of people differently, and we plan to include it in the future version of the system. One more possible future path would be to include human detection and tracking system along with the navigation system.

References

[1] T. Kanda, M. Shiomi, *et al.*, "An affective guide robot in a shopping mall," in *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, 2009.

[2] M. E. Foster, B. Craenen, *et al.*, "Mummer: Socially intelligent human-robot interaction in public spaces," *arXiv preprint arXiv:1909.06749*, 2019.

[3] R. Triebel, K. Arras, R. Alami, *et al.*, "Spencer: A socially aware service robot for passenger guidance and help in busy airports," in *Field and service robotics*, Springer, 2016.

[4] G. Ferrer, A. Garrell, *et al.*, "Social-aware robot navigation in urban environments," in *IEEE ECMR*, 2013.

[5] M. F. Carmona, T. Parekh, *et al.*, "Making the case for human-aware navigation in warehouses," in *Annual Conference Towards Autonomous Robotic Systems*, Springer, 2019.

[6] X.-T. Truong, V. N. Yoong, *et al.*, "Dynamic social zone for human safety in human-robot shared workspaces," in *11th International Conference on Ubiquitous Robots and Ambient Intelligence*, 2014.

[7] M. Kollmitz, K. Hsiao, *et al.*, "Time dependent planning on a layered social cost map for human-aware robot navigation," in *IEEE ECMR*, 2015.

[8] P.-T. Singamaneni and R. Alami, "Hateb-2: Reactive planning and decision making in human-robot co-navigation," in *International Conference on Robot & Human Interactive Communication*, 2020.

[9] X.-T. Truong and T. D. Ngo, "Toward Socially Aware Robot Navigation in Dynamic and Crowded Environments: A Proactive Social Motion Model," *IEEE Transactions on Automation Science and Engineering*, 2017.

[10] E. Repiso, G. Ferrer, and A. Sanfeliu, "On-line adaptive side-by-side human robot companion in dynamic urban environments," in *IEEE/RSJ IROS*, 2017.

[11] Y. F. Chen, M. Everett, *et al.*, "Socially aware motion planning with deep reinforcement learning," in *IEEE/RSJ IROS*, 2017.

[12] B. Okal and K. O. Arras, "Learning socially normative robot navigation behaviors with bayesian inverse reinforcement learning," in *IEEE ICRA*, 2016.

[13] E. A. Sisbot, L. F. Marin-Urias, *et al.*, "A human aware mobile robot motion planner," *IEEE Transactions on Robotics*, 2007.

[14] R. Guldenring, M. Görner, N. Hendrich, *et al.*, "Learning local planners for human-aware navigation in indoor environments," in *IEEE/RSJ IROS*, 2020.

[15] N. Pérez-Higueras, R. Ramón-Vigo, *et al.*, "Robot local navigation with learned social cost functions," in *11th International Conference on Informatics in Control, Automation and Robotics*, 2014.

[16] N. Pérez-Higueras, F. Caballero, *et al.*, "Teaching robot navigation behaviors to optimal rrt planners," *International Journal of Social Robotics*, 2018.

[17] H. Khambhaita and R. Alami, "Viewing robot navigation in human environment as a cooperative activity," in *International Symposium on Robotics Research*, 2017.

[18] D. V. Lu, D. Hershberger, *et al.*, "Layered costmaps for context-sensitive navigation," in *IEEE/RSJ IROS*, 2014.

[19] S. B. Banisetty, V. Rajamohan, *et al.*, "A deep learning approach to multi-context socially-aware navigation," in *IEEE/RSJ IROS*, 2020.

[20] S. B. Banisetty, S. Forer, *et al.*, "Socially-aware navigation: A non-linear multi-objective optimization approach," *arXiv preprint arXiv:1911.04037*, 2019.

[21] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, no. 5, p. 4282, 1995.

[22] G. Ferrer and A. Sanfeliu, "Multi-objective cost-to-go functions on robot navigation in dynamic environments," in *IEEE/RSJ IROS*, 2015.

[23] A. Bordallo, F. Previtali, *et al.*, "Counterfactual reasoning about intent for interactive navigation in dynamic environments," in *IEEE/RSJ IROS*, 2015.

[24] G. Ferrer and A. Sanfeliu, "Bayesian human motion intentionality prediction in urban environments," *Pattern Recognition Letters*, 2014.

[25] J. F. Fisac, A. Bajcsy, *et al.*, "Probabilistically safe robot planning with confidence-based human predictions," *arXiv preprint arXiv:1806.00109*, 2018.

[26] K. Qian, X. Ma, *et al.*, "Decision-Theoretical Navigation of Service Robots Using POMDPs with Human-Robot Co-Occurrence Prediction," *International Journal of Advanced Robotic Systems*, 2013.

[27] D. Mehta, G. Ferrer, and E. Olson, "Autonomous navigation in dynamic social environments using Multi-Policy Decision Making," in *IEEE/RSJ IROS*, 2016.

[28] M. Quigley, K. Conley, *et al.*, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, 2009.

[29] C. Rösmann, W. Feiten, *et al.*, "Efficient trajectory optimization using a sparse model," in *IEEE ECMR*, 2013.

[30] T. Kruse, A. Kirsch, *et al.*, "Evaluating directional cost models in navigation," in *ACM/IEEE international conference on Human-robot interaction*, 2014.

[31] G. Echeverria, N. Lassabe, *et al.*, "Modular open robots simulation engine: Morse," in *IEEE ICRA*, 2011.