A hierarchical design for shared-control wheelchair navigation in dynamic environments

Bingqing Zhang
Computer Science
University College London
United Kingdom
bingqing.zhang.18@ucl.ac.uk

Catherine Holloway
UCL Interaction Centre
University College London
United Kingdom
c.holloway@ucl.ac.uk

Tom Carlson
Aspire CREATe
University College London
United Kingdom
t.carlson@ucl.ac.uk

***Abstract*— For people who have a mobility impairment and find standard wheelchairs unsuitable, a shared-controlled approach could provide a potential mobility solution. However, state-of-the-art research on shared control wheelchairs mainly focus on static environments. In this paper, we present a hierarchical design for our shared-controlled wheelchair using a velocity-based approach together with probabilistic shared control (PSC). By modifying the collision avoidance element and model the robot-pedestrian interaction based on their physical distance, we extended the implementation of PSC to dynamic environments. Our approach was tested in a Unity3D based simulator with human participants. It achieved least number of collisions while obtaining relatively low computational cost and high user agreement comparing with other state-of-the-art methods.**

Keywords—shared control, navigation

I. Introduction

Being able to move around is key to independence and quality of life. While wheelchairs provide a mobility solution, current designs may not be suitable for people with very high level motor impairments, especially when combined with sensory and / or cognitive impairments. To help enable independent mobility for people with such complex needs, research in shared control wheelchairs is being carried.

A shared-controlled wheelchair is a standard powered wheelchair with a collection of sensors. It has the ability to sense its environment and make collision-free actions based on the user's input. In terms of the control strategy, the shared-controlled wheelchairs can mainly be divided into two categories. Escobedo et al. (2013) and Tomari et al. (2012) adopted a hierarchical framework where the user was in charge of high-level tasks such as choosing a desired manoeuvre whilst the wheelchair took control of the low-level tasks [1], [2]. The user could intervene with the autonomous process by switching the operating mode. Similarly, Simpson et.al (1999) proposed an automatic adaption strategy, where the motion planner can automatically select the correct operating mode [3]. However, this strategy only assists in driving in some pre-defined scenarios and may not be suitable to deal with uncertainty.

An alternative control paradigm of shared control is achieved by continuously blending input from both the user and the motion planner [4]–[7] . One of the most popular blending approaches is linear blending, where the final control command is a weighted sum of the user's input and the command computed by the motion planner. As a result, the main focus of this approach lies in the weight negotiation between the user and the planner in order to optimize certain objectives. However, Trautman (2015) demonstrated mathematically that simple linear blending may not guarantee a collision-free trajectory and may not best describe the user's intention. To address these issues, he proposed probabilistic shared control (PSC) [8], which models the human-robot interaction by taking interaction uncertainty into account. This approach adds more flexibility to the user-wheelchair collaboration startegies, while guaranteeing safety mathematically [8]. Following this line of research, Ezeh et al. (2017) applied PSC in a wheelchair navigation task and demonstrated that PSC is safer than liner blending through experiments [4]. However, similar to other state-of-the-art research into the shared-controll of wheelchairs, the application scenario is limited to the static environment only. For a shared-control wheelchair to be useable by people in the real world avoiding dynamic obstacles such as pedestrians will be essential.

In this paper, we first present a hierarchical design for shared-control local navigation, which builds on previous research on PSC and extends its application scenarios to a dynamic environment. We then validate our design in a simulator which is based on Unity3D and compare its performance with other methods.

II. Realted work

In this section, we give a brief overview of related work on velocity-based collision avoidance approaches and probabilistic shared control, which are the building blocks for our design.

A. *Dynamic Window Approach (DWA)*

The Dynamic Window Approach (DWA) which was first proposed by Fox et al. is an obstacle-avoidance method that takes into account the dynamic and kinematic constraints of a mobile robot [9]. The main idea of this approach involves finding feasible and admissible velocities in a velocity search space (v,w) which is pruned by using the idea of a dynamic window. Among all the candidate velocity pairs, the optimal velocity pair is chosen as the one that optimizes certain objective functions [9].

Although this approach demonstrates effective obstacle avoidance with static obstacles, the original version of DWA does not incorporate obstacle velocity information and thus is not suitable to be used in a dynamic environment. An extension of this approach has been proposed in [10], which was designed to overcome the issue of dynamic obstacle avoidance by predicting moving obstacles' future positions using a dynamic occupancy map. However, this approach is computationally expensive.

B. *Generalized Velocity Obsatcle*

Generalized Velocity Obstacle (GVO) is a velocity based collision avoidance approach that was proposed by Wilkie (2009) [11]. It is an extension of the original Velocity Obstacle (VO) algorithm proposed by Fiorini and Shiller in [12], which allows navigating car-like robots among arbitrarily moving obstacles. Different from VO which constructs collision cones geometrically, GVO calculates velocity obstacles algebraically. In detail, for a car-like robot which is assumed to have a constant speed u_s and steering angle u_φ, GVO calculates the future position A of the robot at time t with respect to its current robot frame as

$$A(t,u) = \begin{pmatrix} \frac{L}{tan(u_\varphi)} sin\left(\frac{u_s\, tan(u_\varphi)t}{L}\right) \\ -\frac{L}{tan(u_\varphi)} cos\left(\frac{u_s\, tan(u_\varphi)t}{L}\right) + \frac{1}{tan(u_\varphi)} \end{pmatrix} \quad (1)$$

Where L is the wheelbase length.

Obstacles are considered to have linear motion and their future position B_i at time t can be easily calculated as

$$B_i(t) = \boldsymbol{p}_{B_i} + \boldsymbol{v}_{B_i}t \quad (2)$$

And the velocity obstacle can be constructed for which

$$\|A(t_{min}, u) - B_i(t_{min})\| < r_A + r_B \quad (3)$$

Where r_A and r_B are radius for the robot and the obstacles. As a result, a feasible speed can be found which falls outside the velocity of the obstacles, and the final command is calculated by solving an optimization problem.

C. *Probabilistic Shared Control*

In a method proposed by Trautman (2015), probabilistic shared control assumes both the user's input (interpreted as velocity pairs) and the candidate velocities (computed by the motion planner), follow a probability distribution [8]. The process is implemented by taking the joint probability of the velocity input from both the user and the planner, and the optimal command can be found by maximizing this joint probability. It can be formulated as

$$u_{t+1}^{PSC} = u_{t+1}^{R*} \quad (4)$$

$$u_{t+1}^{PSC} = argmax_{u_{t+1}^R} p(u_{t+1}^h, u_{t+1}^R, u_{t+1}^c | \underline{z}_{1:t}^h, \underline{z}_{1:t}^R, \underline{z}_{1:t}^c) \quad (5)$$

$$p(u_{t+1}^h, u_{t+1}^R, c) = \varphi(u_{t+1}^h, u_{t+1}^R) p(u_{t+1}^h | z_{1:t}^h) p(u_{t+1}^R, u_{t+1}^c | \bar{z}_{1:t}) \quad (6)$$

where $\underline{z}_{1:t}^h$ represents measurements of the user input, $\bar{z}_{1:t} = [\underline{z}_{1:t}^R, \underline{z}_{1:t}^C]$ in which $\underline{z}_{1:t}^R$ defines measurements of the robot trajectory u^R and $\underline{z}_{1:t}^c$ stands for measurements of obstacles trajectory u^c.

$\varphi(u_{t+1}^h, u_{t+1}^R)$ captures how "agreeable" the robot trajectory is with the intention of the user, while $p(u_{t+1}^R, u_{t+1}^c | \bar{z}_{1:t})$ models the autonomy of the robot by taking human-robot interactions into considerations.

Ezeh et al. (2017) implemented PSC for a wheelchair that was able to assist a user navigating in an environment with static obstacles [4]. Mathematically, the implementation ignored the measurements and distribution of the moving obstacles and thus simplifying the problem to

$$u_{t+1}^{PSC} = argmax_{u_{t+1}^R} p\left(u_{t+1}^h, u_{t+1}^R \middle| \underline{z}_{1:t}^h, \underline{z}_{1:t}^R\right) \quad (7)$$

$$p(u_{t+1}^h, u_{t+1}^R, c) = \varphi(u_{t+1}^h, u_{t+1}^R) p(u_{t+1}^h | z_{1:t}^h) p(u_{t+1}^R | \underline{z}_{1:t}^R) \quad (8)$$

III. Design

A. *Our wheelchair*

In order to better understand our design structure, we first describe our wheelchair platform.

As shown in Fig. 1, our wheelchair is built on a standard Salsa M2 electrical wheelchair with a collection of additional sensors.

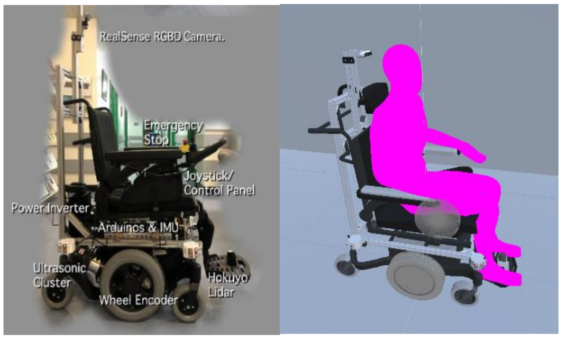


Fig. 1. Left: Our smart wheelchair with sensors including 12 ultrasonic sensors, 1 Hokuyo Lidar, 2 wheel encoders, 1 IMU and 1 RGBD camera. On-board processors such as Arduinos are places under the seat. Right: 1:1 model in Unity3D

In detail, the wheelchair is equipped with two wheel-encoders and one inertial measurement unit (IMU) which are used for dead-reckoning. For perception purposes, ultrasonic sensors, one Lidar and one RGB-D camera have been used. In terms of short-range sensing, 12 ultrasonic sensors (SR08) haven been placed around the bottom frame with 3 of them in each cluster. Each sensor has a sensing angle of 45 degrees up to 1.5m. For long-range sensing, a Hokuyo URG 2D Lidar has been fixed on top of the foot-plate, with a maximum range of 5.6m and sensing angle from -120 °to 120 °. The Realsense RGB-D camera was mounted on top of the wheelchair, which is used for moving obstacle (pedestrian) detection and tracking.

This work is funded by EU2020 project CROWDBOT

B. A hierarchical design

Our proposed design used a hierarchical framework for collision avoidance, which treats static and dynamic obstacles separately. The reasons for this are threefold.

- It allows more flexibility and transparency in control.
- It reduces computational complexity by pruning the subsequent velocity search space for dynamic obstacles avoidance.
- Each sensor only deals with one type of obstacle, which allows it to be associated with a different probability, thus making it easier to be used within probabilistic shared control.

Fig. 2 shows a overview for our collision avoidance and shared control algorithm. In detail, we first search the velocity space for all achievable velocity pairs based on the wheelchair kinematics. Then, the dynamic window approach (DWA) is applied for low-level collision avoidance with static obstacles, which looks into a time horizon $\Delta\mathrm{t}$. In our setting, we found that $\Delta\mathrm{t} = 0.5s$ gave a reasonable result. The resultant admissible and achievable speed is further used as the search space for generalized velocity obstacle (GVO), which filters out any velocity that may result in a collision with moving obstacles. In our implementation, the simulated moving obstacles (pedestrian avatars) each have a radius of $r_B = 0.33\mathrm{m}$. The wheelchair radius is considered as the maximum length on the wheelchair edge from its centre of mass, which is $r_A = 0.6\mathrm{m}$ in our case. Additionally, we assign a safety margin $r_{safe} = 0.15m$. As a result, the final candidate velocity u should satisfy:

$$\|A(t_{min}, u) - B_i(t_{min})\| > r_A + r_B + r_{safe} \quad (9)$$

After this pruning process, a cost function is used to evaluate each final candidate, in terms of heading, clearance and velocity. This cost is interpreted as the probability associated with each candidate velocity, which will be used in the shared control stage.

In terms of the shared control blending strategy, PSC is served as our theoretical basis where we extend it to deal with moving obstacles. We make the following assumptions.

- As no global planning is involved in the navigation, a local goal is defined as 2m ahead of the user's intended trajectory *(see next bullet point)*.
- The user's intended trajectory is only based on the current input at the most recent time step. This allows us to simplify $p(u_{t+1}^h|z_{1:t}^h)$ to $p(u_{t+1}^h|z_t^h)$
- As a starting point of this research, obstacles are currently assumed to have linear motion, and their trajectories are only based on the measurements at the most recent time step. This allows us to reduce $\underline{z}_{1:t}^c$ to $\underline{z}_t^c$.
- The motion of the obstacles is assumed not to be affected by the robot, only a simple interaction function between the moving obstacles and the wheelchair has been modelled.

Mathematically, the problem can be simplified to:

$$u_{t+1}^{PSC} = argmax_{u_{t+1}^R} p\left(u_{t+1}^h, u_{t+1}^R, u_{t+1}^c \middle| \underline{z}_t^h, \underline{z}_{1:t}^R, \underline{z}_t^c\right) \quad (10)$$

$$p(u_{t+1}^h, u_{t+1}^R, c) = \varphi(u_{t+1}^h, u_{t+1}^R) p(u_{t+1}^h|z_t^h) p\left(u_{t+1}^R, u_{t+1}^c \middle| \underline{z}_{1:t}^R, \underline{z}_t^c\right) \quad (11)$$

The agreeability between the user and the planner is modelled as:

$$\varphi(u_{t+1}^h, u_{t+1}^R) = exp(-\frac{1}{2\gamma}(\hat{u}_{1+t}^h - \hat{u}_{1+t}^R)(\hat{u}_{1+t}^h - \hat{u}_{1+t}^R)')\quad (12)$$

where $\hat{\mathrm{u}}_{1+\mathrm{t}}^{\mathrm{h}}$ and $\hat{\mathrm{u}}_{1+\mathrm{t}}^{\mathrm{R}}$ are normalized user and robot trajectories (input). The parameter γ which has a range of $\gamma > 0$ models how strongly $\hat{\mathrm{u}}_{1+\mathrm{t}}^{\mathrm{h}}$ and $\hat{\mathrm{u}}_{1+\mathrm{t}}^{\mathrm{R}}$ are correlated. As a result, the agreeability φ is between 0 and 1, with 1 being the path planner's decision exactly the same as the user's intended trajectory. Through trial and error, we set $\gamma = 100$ in our implementation to achieve an acceptable trade-off.

In addition, the autonomy probability can be written as:

$$p\left(u_{t+1}^R, u_{t+1}^c \middle| \underline{z}_{1:t}^R, \underline{z}_t^c\right) = \varphi(u^R, u^c) p(u_{t+1}^R|z_{1:t}^R) \prod_{i=1}^{n_t} p\left(u_{t+1}^c \middle| \underline{z}_t^c\right) \quad (13)$$

where n_t is the number of moving obstacles been detected by the wheelchair onboard sensors at time t.

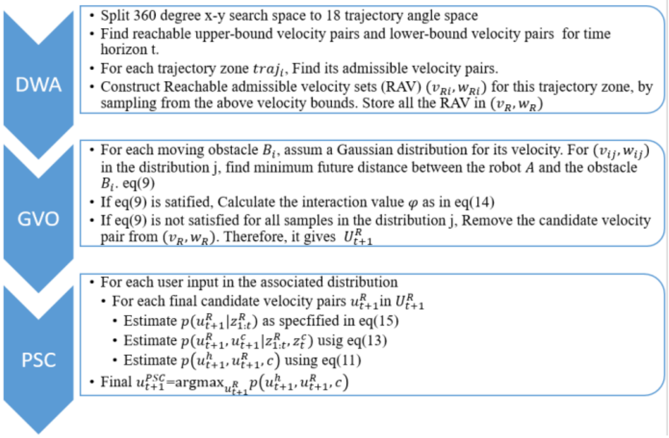
Ideally, the most common scenario for robot navigation that involves moving obstacles is in a human-populated environment. As a result, it is important to consider pedestrian-robot interaction by modelling the interaction function $\varphi(u^R, u^c)$. In this paper, we assume the pedetsrians' short term motion is not affected by the robot, and model the interaction based on robot-pedestrian physical distance.

$$\varphi(u^R, u^c) = \prod_{\mathrm{i}=0}^{\mathrm{n_t}} \left(1 - a * \exp\left(-\frac{1}{2h^2}\left|u_{t+1}^R - u_{t+1}^{c_i}\right|\right)\right) \quad (14)$$

where $\left|u_{t+1}^R - u_{t+1}^{c_i}\right|$ is the Euclidean distance between the robot and each moving obstacle; a is the repulsion force, which can be set between [0,1]; and h is a scaling parameter, which encodes the idea of safety distance implicitly and is set to 0.9. The rationale behind this function is that a low probability is assigned to candidate robot actions which may result in short distance to all moving obstacles that are within the sensor range. $p(u_{t+1}^R|z_{1:t}^R)$ models the motion planner's trajectory distribution based on prior information. This can be considered as the cost that is associated with each candidate velocity pairs that comes from the motion planner. We model the cost function as

$$G(v,w) = \alpha * heading(v,w) + \beta * clearance(v,w) + \gamma * velocity(v,w) \quad (15)$$

where α, β, γ are weights for each term, 'heading' measures the alignment of the robot with the goal direction, 'clearance' represents the distance to the nearest static obstacle on the selected trajectory while 'velocity' equals to the normalized absolute linear candidate velocity. By tuning these parameters, different motion behaviours can be achieved by emphasizing the relative importance of these competing objectives. These parameters together with γ, α and h can be further tuned on a user basis in order to provide the most suitable assistance to the user. Intuitively, we want the wheelchair to run at fast speed if possible and keep certain distance to the obstacle so that the maneveror will not be too aggressive. As a result, in our test, we set α, β, γ to be 0.1,0.3,0.6 respectively which gives a reasonable behaviour through trial and error.



- Split 360 degree x-y search space to 18 trajectory angle space
- Find reachable upper-bound velocity pairs and lower-bound velocity pairs for time horizon t.
- For each trajectory zone $traj_i$, Find its admissible velocity pairs.
- Construct Reachable admissible velocity sets (RAV) (v_{Ri}, w_{Ri}) for this trajectory zone, by sampling from the above velocity bounds. Store all the RAV in (v_R, w_R)

- For each moving obstacle B_i, assum a Gaussian distribution for its velocity. For (v_{ij}, w_{ij}) in the distribution j, find minimum future distance between the robot A and the obstacle B_i. eq(9)
- If eq(9) is satified, Calculate the interaction value φ as in eq(14)
- If eq(9) is not satisfied for all samples in the distribution j, Remove the candidate velocity pair from (v_R, w_R). Therefore, it gives U_{t+1}^R

- For each user input in the associated distribution
  - For each final candidate velocity pairs u_{t+1}^Rin U_{t+1}^R
    - Estimate $p(u_{t+1}^R | z_{1:t}^R)$ as specfified in eq(15)
    - Estimate $p(u_{t+1}^R, u_{t+1}^c | z_{1:t}^R, z_t^c)$ usig eq(13)
    - Estimate $p(u_{t+1}^h, u_{t+1}^R, c)$ using eq(11)
- Final u_{t+1}^{PSC}=argmax$_{u_{t+1}^R} p(u_{t+1}^h, u_{t+1}^R, c)$

Fig. 2. Flowchart of our collision avoidance and shared control algorithm.

C. *Implementation*

Before implementing our strategy on the actual wheelchair, we first implemented and validated it in a simulator which is built on Unity 3D.

In order to achieve an accurate evaluation, all sensors (except the RGB-D camera) characteristics have been modelled in the simulator. Wheelchair dynamics were partly included in the Simulator by using Unity physics components. TABLE 1 shows some key parameters. The center of mass and moment of inertia for the wheelchair body (plus driver), two drive wheels and four caster wheels were calculated automatically from the built 3D models.

TABLE 1. WHEELCHAIR PARAMETERS IN THE SIMULATOR

Parameters	**Value**
Body mass	200kg (Wheelchair + a driver)
Drive wheel mass	2.6kg
Caster wheel mass	1kg
Angular drag	0.05N
Motor maximum torque	400Nm

The strategy was implemented as if all inputs were from actual sensors, which requires minimum modification for it to be transferred to a real-world test. The control strategy was implemented in a ROS framework and the communication between the simulator and ROS is achieved by RosSharp as illustrated in Fig. 3.

To begin with, simple sensor fusion is achieved by implementing an extended Kalman filter in the "robot_localization" package, which takes input from two wheel encoders and the IMU to publish an accurate odometry at 20hz. In terms of perception, 12 ultrasonic sensors and one Lidar senses the surrounding environment, and are used to construct a global occupancy grid map which is published at 10hz. In our implementation, the map has a size of 40m x 40m with a resolution of 0.05m. It is centred at a fixed frame to reduce the computational cost.

Our control strategy takes input from the occupancy grid and implements DWA for static obstacle avoidance. Resultant admissible velocities are used as the search space for the GVO which receives information of the dynamic obstacles from Unity. In the future, this information would be provided by a pedestrian tracker that utilizes the RGB-D camera and Lidar data.

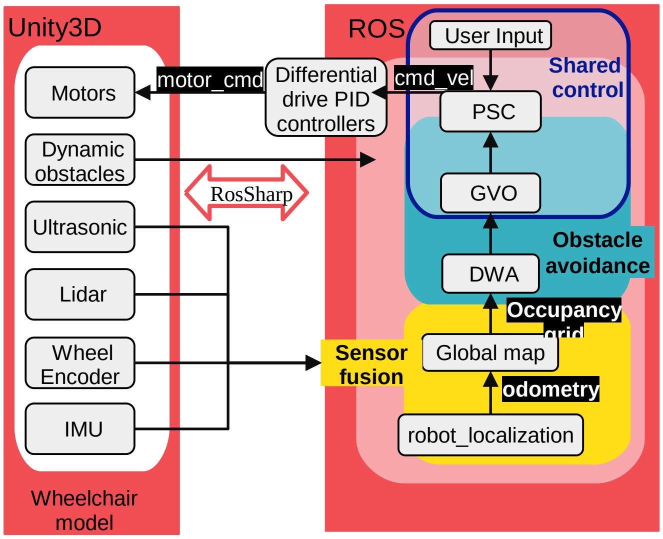


Fig. 3. Structure of the proposed design strategy and its communication with Unity3D

The output from GVO is blended with the user input (which is received from a keyboard) using probabilistic shared control. The final velocity command is published at a rate of 10hz and is subscribed by two differential drive PID controllers running at 50hz, which calculates suitable motor torque for two drive wheels so as to achieve desired linear and angular velocity.

IV. SIMULATION

A. *Simulation Setup*

To evaluate the validity and performance of our proposed strategy, we designed an H-shaped course populated with moving pedestrians. The course was designed to simulate daily wheelchair usage, taking into consideration typical manoeuvres from the Wheelchair Skills Test (WST) [13], specifically we chose three of the most used skills, detailed in TABLE 2.

TABLE 2. WHEELCHAIR SKILLS THAT HAVE BEEN TESTED

Number	Skills from WST
1	Roll forwards
2	Turns while moving forwards (90°)
3	Avoids moving obstacle

Three pedestrian agents were used in our test. These were programmed to move in a simple manner – moving straight in the direction of their original heading. Pedestrians were placed at three corners of the maze with each instructed to move in a different direction at a mean speed of 1.1m/s, this simulates an average human daily walking speed [14]. The initial positions and walking directions were designed to maximize the potential interaction opportunity with the wheelchair. The wheelchair started from the origin (shown in Fig. 4 with a blue star) with a goal of reaching the upper left corner of the H shaped course (shown in Fig. 4 with a red star). The left and right passage of the course has a length of 19m and a width of 3m while the mid passage was 8m long and 3m wide.

During the navigation, all sensor data were published to ROS and a 'global' occupancy grid map constructed which was fixed at the origin to reduce the computational cost.

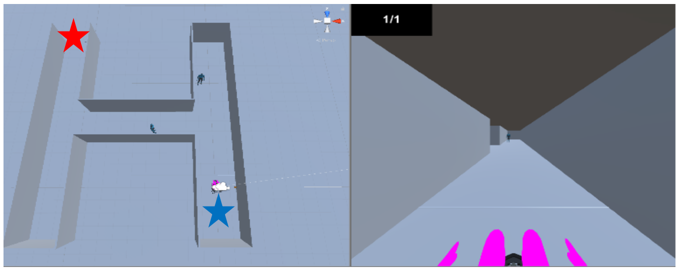


Fig. 4. Left: Screenshot of the simulation (top view) where blue star shows the wheelchair's starting point and the red star represents the goal. Right: First person view

B. Metrics

Three metrics were used to evaluate the safety and assistance for the proposed shared control navigation design. These are defined as:

- C: Number of collisions (with wall or pedestrians)
  This metric is just the count of collisions that occurred in the scenario and was reported by the simulator.

- T_t: Task completion time (the time that user required to reach the goal position from the starting position)
  $$t_c = t_{end} - t_{start} \tag{16}$$

- A: Agreement. We define agreement in terms of the deviation of the direction of the user's command from the direction of the final shared control's command. Mathematically, it is calculated as:
  $$\theta(u) = tan^{-1}(\frac{v}{w}) \tag{17}$$
  $$a_i = 1 - \frac{|\theta(z_h^i) \ominus \theta(u_{SC}^i)|}{\pi} \tag{18}$$

$$agreement = \frac{\sum_{i=0}^{N} a_i \cdot \Delta t_i}{\sum_{i=0}^{N} \Delta t_i} \tag{19}$$

where v and w are the translational and rotational velocities $u \sim [v\ w\ ]$, a_i is the normalised agreement at time step t_i and u_{SC}^i is the final output of the probabilistic shared control. N is the number of samples available in which data from the user measured input z_h^i coincide in time with u_{SC}^i , and Δt_i is the duration of the user's input command z_h^i.

C. Results

Three healthy participants whom are non-wheelchair users were recruited and participated in driving the wheelchair in the simulator. Two had a background in robotics or computer science while 1 participant has an unrelated background. Participants were allowed to perform some trial runs to familiarize themselves with the setup before the actual test. In total, three test trials were conducted where each user drove the wheelchair using keyboard arrow keys (to simulate a head-array interface).

Fig. 5 to Fig. 9 show the results for one trial from one participant (note for these figures, we use a right-hand coordinate frame which means the x-axis is the vertical one and the y-axis is the horizontal one). As shown in Fig. 5 and Fig. 6, the final blended velocity command followed the user's input most of the time, except for when there was a collision risk. In detail, the chosen linear velocity at t=11-16s and t=36-38s, angular velocity at 10-15s and 35-37s are largely different from the user's intended velocity. This is also reflected in Fig. 8 which describes the agreement between the user and the final chosen command. Fig. 9 shows the avoidance trajectory for the wheelchair (plotted in black circles) and the trajectories for three moving pedestrians (plotted in red). The largest disagreements between the user input and the final chosen command are highlighted in Fig. 8 by orange stars.

It can be seen that at t=10-16s, the wheelchair detected two moving pedestrians in proximity while the user's input indicates that the user would like to keep moving forward at the maximum speed. Under this scenario, the control strategy filtered out candidate velocities that may result in collision and found the one that guaranteed safety while best obeying the user's intention. As a result, the wheelchair decreased its linear velocity and made a right turn by choosing negative angular velocities. Afterwards, it stopped for a short time and then moved in the user-intended direction.

Similarly, at t=35-39s, the wheelchair encountered a pedestrian whom moved across. At about t=35-39s, the user kept trying to move straight forward while the controller decided to make a small right turn to avoid future collisions. It then made a big right turn following the user's command.

TABLE 3 provides the test results for each participant. It can be seen that overall the agreement is high for all three participants and no collision occcured. For Participants 3, a slightly longer completion time was observed which could be explained by their unfamiliarity with such system.

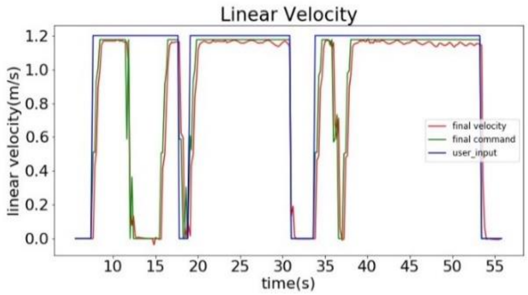


Fig. 5. Linear Velocity

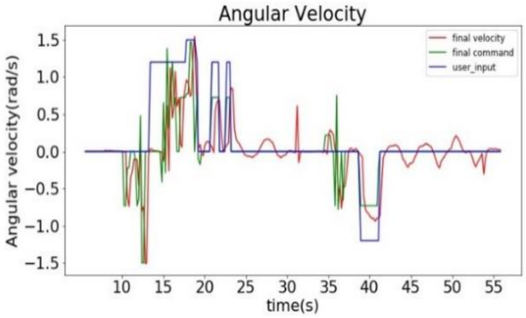


Fig. 6. Angular Velocity

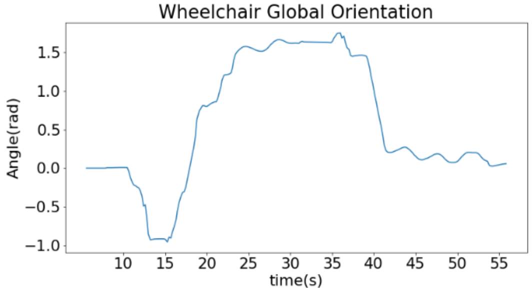


Fig. 7. Wheelchair orientation

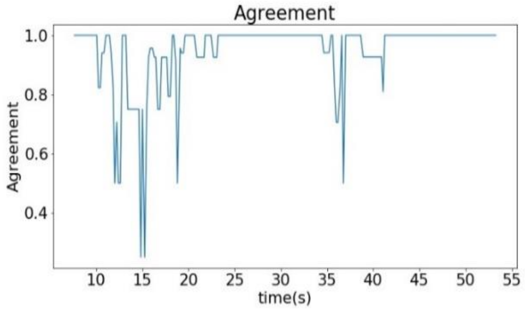


Fig. 8. Agreement between the user's input and the final chosen command

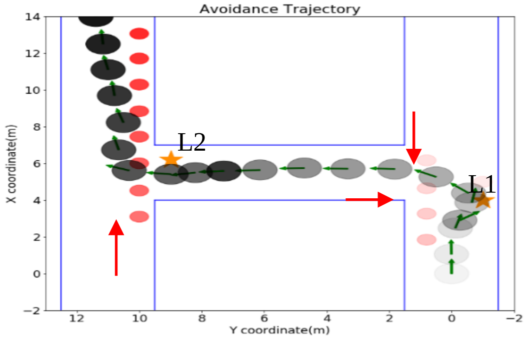


Fig. 9. Wheelchair avoidance trajectory, where the wheelchair detects two pedestrians at L1 and one pedestrian at L2, and make certain motion adaption. The transparency represents the time (alpha from 0-100 corresponds to the time from 0 to 75s). The red arrows represent the moving direction of each pedestrian and the green arrows represent the orientation of the wheelchair).

TABLE 3. TESTING RESULTS FOR THREE PARTICIPANTS

Participant	**C**	$\boldsymbol{T_t}$**(s)**	**A**
1	0	45.15	0.9777
2	0	46.47	0.9469
3	0	50.25	0.9882

In general, we can see that the final command chosen by the controller generated safe and reasonable motions, which allows the wheelchair to avoid static and moving obstacles while following the user's intention. The parameters were chosen in a way that favours high speed while guarantees a safety distance.

D. Comparison

We further evaluated our approach (we will call it GVDWA) by comparing it with related works, namely PSC with naive DWA which was originally implemented by Ezeh et al. [4], and PSC with DWA which takes obstacle velocity into account (we will call it VDWA)[10]. For a valid comparison, we used the same parameters as had been used for our method GVDWA for the cost function in all three methods.

In order to examine its performance in different human density environments, we designed three scenarios as shown in Fig. 10. For each scenario, one user drove the wheelchair three times with each control method, and the performance was evaluated in terms of average number of collisions (C), average computation time (T_c) ,average agreement (A) and average task completion time (T_t). TABLE 4 provides a summary of the result.

It can be seen that in general GVDWA+PSC exceeds other methods in terms of reducing the number of collisions, while its computation time, agreement and task completion time are comparable with PSC+naive DWA. Of the three methods, PSC+naive DWA has the highest number of collisions, which is to be expected as it treats moving obstacles as stati and may result in moving towards the direction of the obstacle. For S1, we further tested the capability of naive DWA+PSC and only observed collision free avoidance when the speed of the avatar was decreased to 0.5m/s.

When obstacles' velocity information is incorporated in DWA, the number of collisions was reduced at the cost of increased computation time. Of the three methods, our proposed method GVDWA and VDWA both predict moving obstacle positions over a short time horizon based on theirvelocity, where VDWA+PSC shows the longest computational time due to the use of a dynamic occupancy map.

In terms of agreement, all three methods result in agreement greater than 0.9, while naive DWA+PSC shows a slightly higher value. This can be explained because in some situations, this method moves towards the user's desired direction despite future collision danger with moving obstacles. Although GVDWA+PSC has slightly lower agreement, the deviation normally occurs when there is a risk of collision. Fig. 11 shows the avoidance trajectory with its corresponding velocity for GVO+DWA+PSC in H_maze with 6 people. Two obvious avoidance can be seen in the two orange star.

As for the task completion time, GVDWA+PSC has the value between VDWA+PSC and DWA+PSC. This result is consistents with the computational time and the agreement, and can be explained as the wheelchair deviated from the user's desired path (potentially the shortest) to avoid obstacles, which took longer time.

TABLE 4. COMPARISON RESULTS SUMMARY(Mean±SD)

	S1（1 pedestrian）				S2 (H_maze + 3 pedestrians)				S3 (H_maze + 6 pedestrians)			
	C	T_c(s)	A	T_t(s)	C	T_c(s)	A	T_t(s)	C	T_c(s)	A	T_t(s)
GVDWA +PSC	0 ± 0	0.0793± 0.0017	0.9589± 0.0158	14.78± 0.5000	0 ± 0	0.0808± 0.0043	0.9540± 0.0050	45.01± 2.9492	0±0	0.0830± 0.0010	0.9356± 0.0393	46.87 ±3.8 192
VDWA+ PSC	0 ± 0	0.0832± 0.0007	0.9679± 0.0030	15.23± 0.3915	0 ± 0	0.1061± 0.0079	0.9469± 0.0048	46.87± 0.5834	0.33± 0.5774	0.1405± 0.0086	0.9502± 0.0154	48.10 ±2.8 744
DWA+ PSC	1 ± 0	0.0773± 0.0049	0.9889± 0.0087	14.96± 0.8908	2 ± 1	0.0768± 0.0045	0.9856± 0.0096	44.6±3 .6128	2.66± 0.5774	0.0818± 0.0053	0.9593± 0.0151	46.65 ±2.2 05

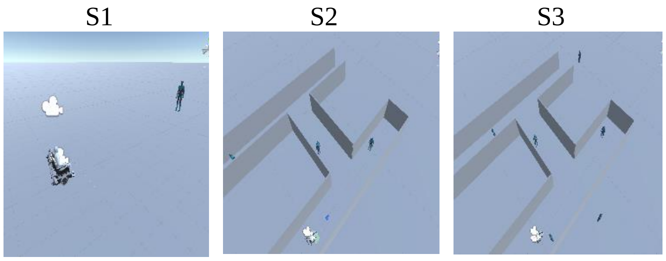


Fig. 10. Three test scenarios. S1 has one pedestrian moving towards the wheelchair, S2 has three pedestrians moving in different directions in the H maze while the number of pedestrians was increased to six in S3. All pedestrians move at an average speed of 1.1m/s

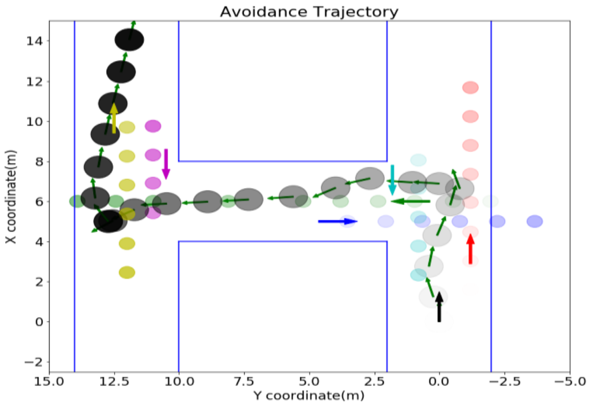


Fig. 11. Avoidance trajectory of the wheelchair in S3 for using our method. Black circle represents the trajectory of the wheelchair and the green arrows represent its orientation (starting from bottom right and move towards top left). Circles with other colors shows the trajectory of different pedestrians when in close proximity with the wheelchair. The wheelchair successfully avoided all pedestrians, and the main avoidance points are shown in orange stars.

E. *Discussion*

The simulation result indicates the validity of our design, which can be seen as a first step towards enabling a shared-controlled wheelchair to navigate in a dynamic environment. Due to the simulator setup and the implementation method, minimum code modification is needed to transfer this approach onto the actual wheelchair. In the future, the test will be carried out in the physical world, where information about the dynamic obstacles will be provided by a people tracker.
However, we would like to point out some limitations of our design and how we intend to address them:

1) *Computational cost*

The Dynamic window approach is computationally expensive. In order to reduce computational time and still allow the design loop to be executed in real-time, we have limited the velocity search space to those that generate trajectories that fall within [-90,90] degrees of the user's intended trajectory. For example, if the user's intention is going forward, the planner will only search for the candidate with $v \geq 0$. This simplification also takes the user's comfort into consideration, as the user may feel uncomfortable if the wheelchair drives in the opposite direction to the way as he or she intended.

In our implementation, we set the horizon for DWA and GVO to be 4s. It allows real-time implementation at a frequency of 10hz on a 2.6GHz Intel Core i7-9750H CPU. However, we expect the computational time to increase when more pedestrians are involved and the crowd density increases.

2) *Latency*

We observed that there is about a 0.25s latency in the system. This is mainly due to the motor latency and the communication delay between ROS and Unity. In order to reduce its effect on the system, when performing the dynamic window search, we assumed the wheelchair keeps moving with its current velocity for two time intervals, and the search starts with the new wheelchair position.

3) Morphology

One main limitation for GVO is that it assumes both the wheelchair and the pedestrians are circular objects. This poor representation of the morphology of the wheelchair may lead to over-cautious behaviour when close to other dynamic obstacles. We would like to address this problem in the future.

4) Simplified obstacles and interaction function

As mentioned in Section III, we assume the obstacles' next movement is only based on its current trajectory measurement. This may not be true when this "obstacle" is a human. In addition, we use a simplified function to model the interaction between the obstacles (human) and the wheelchair which is similar to a cost function that favours velocity which could result in exaggerated clearance. In reality, pedestrians may adapt their trajectory to avoid the robot. Recently, a line of research focuses on investigating human-robot interaction for social navigation using either a modelling [15] or learning [16]–[18] approach. However, to the best of our knowledge, these have not been implemented on a shared-controlled wheelchair as the interaction becomes more complex when a human user is involved.

In the future, we would like to follow this line of research and investigate real-world interactions to better understand this problem.

V. Conclusion

In this paper, we presented a hierarchical design for solving a shared controlled navigation problem in an environment populated with static and dynamic obstacles. We extend the PSC implementation to account for the probability distribution of dynamic obstacles and their influence on robot navigation. The shared control is achieved by maximizing the joint probability between the user, the path planner and the surrounding obstacles. The performance of our method GVDWA has been evaluated in different simulated scenarios and showed the least number of collisions while obtaining relatively low computational cost and high user agreement when comparing with other approaches.

References

[1] A. Escobedo, A. Spalanzani, and C. Laugier, "Multimodal control of a robotic wheelchair: Using contextual information for usability improvement," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Tokyo, Nov. 2013, pp. 4262–4267, doi: 10.1109/IROS.2013.6696967.

[2] M. R. M. Tomari, Y. Kobayashi, and Y. Kuno, "Development of Smart Wheelchair System for a User with Severe Motor Impairment," *Procedia Engineering*, vol. 41, pp. 538–546, 2012, doi: 10.1016/j.proeng.2012.07.209.

[3] R. C. Simpson and S. P. Levine, "Automatic adaptation in the NavChair Assistive Wheelchair Navigation System," *IEEE Transactions on Rehabilitation Engineering*, vol. 7, no. 4, pp. 452–463, Dec. 1999, doi: 10.1109/86.808949.

[4] C. Ezeh, P. Trautman, L. Devigne, V. Bureau, M. Babel, and T. Carlson, "Probabilistic vs linear blending approaches to shared control for wheelchair driving," in *2017 International Conference on Rehabilitation Robotics (ICORR)*, London, Jul. 2017, pp. 835–840, doi: 10.1109/ICORR.2017.8009352.

[5] Qinan Li, Weidong Chen, and Jingchuan Wang, "Dynamic shared control for human-wheelchair cooperation," in *2011 IEEE International Conference on Robotics and Automation*, Shanghai, China, May 2011, pp. 4278–4283, doi: 10.1109/ICRA.2011.5980055.

[6] C. Urdiales, J. M. Peula, M. Fernandez-Carmona, R. Annicchiaricco, F. Sandoval, and C. Caltagirone, "Adaptive collaborative assistance for wheelchair driving via CBR learning," in *2009 IEEE International Conference on Rehabilitation Robotics*, Kyoto, Japan, Jun. 2009, pp. 731–736, doi: 10.1109/ICORR.2009.5209575.

[7] M. Fernandez-Carmona, B. Fernandez-Espejo, J. M. Peula, C. Urdiales, and F. Sandoval, "Efficiency based collaborative control modulated by biometrics for wheelchair assisted navigation," in *2009 IEEE International Conference on Rehabilitation Robotics*, Kyoto, Japan, Jun. 2009, pp. 737–742, doi: 10.1109/ICORR.2009.5209573.

[8] P. Trautman, "Assistive Planning in Complex, Dynamic Environments: A Probabilistic Approach," in *2015 IEEE International Conference on Systems, Man, and Cybernetics*, Kowloon Tong, Hong Kong, Oct. 2015, pp. 3072–3078, doi: 10.1109/SMC.2015.534.

[9] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, Mar. 1997, doi: 10.1109/100.580977.

[10] M. Seder and I. Petrovic, "Dynamic window based approach to mobile robot motion control in the presence of moving obstacles," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, Rome, Italy, Apr. 2007, pp. 1986–1991, doi: 10.1109/ROBOT.2007.363613.

[11] D. Wilkie, J. van den Berg, and D. Manocha, "Generalized velocity obstacles," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louis, MO, USA, Oct. 2009, pp. 5573–5578, doi: 10.1109/IROS.2009.5354175.

[12] P. Fiorini and Z. Shiller, "Motion Planning in Dynamic Environments Using Velocity Obstacles," *The International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, Jul. 1998, doi: 10.1177/027836499801700706.

[13] R. L. Kirby, J. Swuste, D. J. Dupuis, D. A. MacLeod, and R. Monroe, "The Wheelchair Skills Test: A pilot study of a new outcome measure," *Archives of Physical Medicine and Rehabilitation*, vol. 83, no. 1, pp. 10–18, Jan. 2002, doi: 10.1053/apmr.2002.26823.

[14] R. V. Levine and A. Norenzayan, "The Pace of Life in 31 Countries," *Journal of Cross-Cultural Psychology*, vol. 30, no. 2, pp. 178–205, Mar. 1999, doi: 10.1177/0022022199030002003.

[15] P. Trautman, J. Ma, R. M. Murray, and A. Krause, "Robot navigation in dense human crowds: Statistical models and experimental studies of human–robot cooperation," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 335–356, Mar. 2015, doi: 10.1177/0278364914557874.

[16] H. Bai, S. Cai, N. Ye, D. Hsu, and W. S. Lee, "Intention-aware online POMDP planning for autonomous driving in a crowd," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, USA, May 2015, pp. 454–460, doi: 10.1109/ICRA.2015.7139219.

[17] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-Robot Interaction: Crowd-aware Robot Navigation with Attention-based Deep Reinforcement Learning," *arXiv:1809.08835 [cs]*, Sep. 2018, Accessed: May 27, 2019. [Online]. Available: http://arxiv.org/abs/1809.08835.

[18] A. Vega, L. J. Manso, R. Cintas, and P. Núñez, "Planning Human-Robot Interaction for Social Navigation in Crowded Environments," in *Advances in Physical Agents*, vol. 855, R. Fuentetaja Piz án, Á. Garc ía Olaya, M. P. Sesmero Lorente, J. A. Iglesias Mart ínez, and A. Ledezma Espino, Eds. Cham: Springer International Publishing, 2019, pp. 195–208.