JiaoLong- an Intelligent Wheelchair for Handicapped and Elderly

Jingchuan Wang*, Weidong Chen, Zhixuan Wei, Wenlong Liao and Wenxi Zhang

Department of Automation, Shanghai Jiao Tong University, 200240, China
jchwang@sjtu.edu.cn

Abstract—**An intelligent wheelchair JiaoLong with multi-mode is developed for the handicapped and the elderly. Based on the electric wheelchair, JiaoLong is equipped with a multi-sensor system including encoder, laser, and microphone. It is designed of two manipulate modes according to the user's disability and the environment for use. The dynamic shared control mode adapts to the users who are in some high handicap level and have some defects in powered wheelchair control, such as weak eyesight and trembling hand. The autonomous navigation mode adapts to the users who have difficulty and even no ability in powered wheelchair control. Besides, based on the dynamic localizability matrix, an improved particle filter localization algorithm is also proposed in this paper. A comprehensive platform for both driver training and algorithm design of wheelchair is presented also. It can be used for intelligent wheelchair algorithm design.**

I. Introduction

INTELLIGENT wheelchairs have been developed in a number of research labs over the past years. Conventional wheelchairs require the presence of healthcare stuff or family members who are burdened with actually operating the wheelchairs. In order to release the burden of staff in facilities such as hospitals and nursing rooms, fundamental functions such as: Avoid Obstacle, Follow Wall, and Pass Doorway have been proposed[1]. According to the past researches, safety and comfort have also been important aspects for intelligent wheelchair. On top of that, enhancement of independent mobility by a smart wheelchair can also help in rebuilding people's confidence of social skills [2].

However, a more important feature of smart wheelchair is functionality, to make the systems adaptable to the particular needs of each user according to the type and degree of handicap involved. In this aspect, the control system of a smart wheelchair should change the degree of autonomy according to the user's control ability. This user adapted idea is very useful especially when people are having rehabilitation training [3,4,5]. In this paper we introduce an intelligent wheelchair: JiaoLong with multi-mode, the aim of which is to provide an aid to mobility for different disabled and elderly people.

Based on the commercial powered wheelchair, JiaoLong is equipped with encoder and Laser Range Finder (LRF) [6], has been designed to meet a wide range users through multi-mode control system design. This paper introduces the control system of JiaoLong, the realization of two modes aiming to meet the demands of users of different level of handicap, an improved particle filter localization algorithm we proposed to ensure safety travel, as well as the results of the evaluation experiments carried out in welfare institute environments.

* corresponding author

Integrating kinematics, dynamics, sensors, and the 3D models of wheelchair and world, a comprehensive platform for both driver training and algorithm design of wheelchair is presented also. It realistically mimics the characteristics of wheelchair, and takes care of all interactions between the wheelchair and the world.

Fig.1 Prototype of JiaoLong

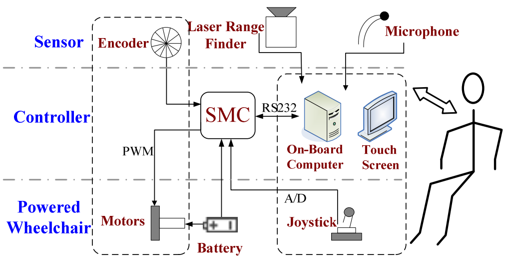


Fig.2 Hardware structure of the JiaoLong wheelchair

II. Control Mode

The JiaoLong wheelchair prototype is based on an ordinary powered wheelchair. Fig.2 provides an overall view of the JiaoLong system architecture[6]. A multi-mode control method for intelligent wheelchair is proposed in JiaoLong system. According to the type and degree of different users' handicap, two control modes of JiaoLong are provided: **Dynamic Shared-Control Mode** and **Autonomous Navigation Mode**. These modes could be chosen, changed and confirmed though the human-friendly interface of OBC via touch screen or voice. Control method based on multi-mode makes JiaoLong adaptable to the needs of different users in different environments[7].

A. Dynamic Shared-Control Mode

Humans, especially old or disabled ones, are less precise in maneuver, do not preserve curvature well and sometimes have difficulty in perceiving their surroundings. Pure reactive controller has problems like fall in local traps, oscillation and unsmooth trajectory. However, human is always good at high level planning, and machine is precise in detecting environmental information and executing motion control. It is significant to find out a method that can combine human control ability and machine control ability effectively[7]. Machine assistance should be adaptable to the difference of the users' control abilities. A Dynamic Shared Control (DSC) algorithm rooted in this idea is proposed in this paper[8].

Fig.3 depicts the functional diagram of this algorithm. The parameters are depicted

v_{user}, ω_{user}: the speed from user.

ω_{mach} ,$\omega_{optimal}$: the desired rotational speed generated by the reactive controller and weight optimizer.

v_{final} ,ω_{final}: the desired speed to be sent to SMC.

Obs_Info: the obstacle distribution map.

v_0 ,ω_0: the current speed of the wheelchair.

k_1 , k_2: the control weights of the user and the reactive controller.

There are two key parts in this architecture: the reactive controller and the user-adapted control weight assignment. The reactive controller provides basic machine help (ω_{mach}) using MVFH&VFF methods. The user-adapted control weight assignment assigns control weights to user and the reactive controller. The minimum vector field histogram (MVFH) method and vector force field (VFF) method which the reactive controller adopts are developed by the University of Michigan[9,10].

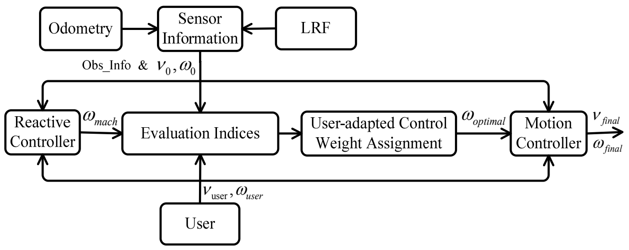


Fig.3 Architecture of the dynamic shared control system

The algorithm of dynamic shared control system includes below steps:

1) Gather LRF, odometer data and the user's input (v_{user}, ω_{user}).

2) Update obstacle distribution map (Obs_Info).

3) If the distance between JiaoLong and obstacles is in dangerous distance, then go to step 6).

4) The reactive controller calculates the machine output ω_{mach} by using MVFH&VFF.

5) The control weights of the user and the reactive controller (k_1 , k_2) are calculated by user-adapted control weight assignment. Go to step 7).

6) Set k_1=0, k_2=0.

7) The motion controller calculates v_{final} and ω_{final} according to v_{user} and $\omega_{optimal}$ and sends v_{final} and ω_{final} to SMC via RS232.

Three indices are proposed to evaluate the controller's performance: *safety*, *comfort* and *obedience.* The performance in maneuver is measured by evaluation indices and the weight optimizer calculates the user's control weight according to this performance. User-adapted control weight assignment is the center of dynamic shared-control mode. Through the weights dynamic assignment of the user and the reactive controller, the maximum indices is obtained. As shown in Fig.4, the Final Command is obtained in Potential Area. The key of weights dynamic assignment is how to get k_1 , k_2 through the optimized model which describes the information of obstacles, user's input and input of reactive controller.

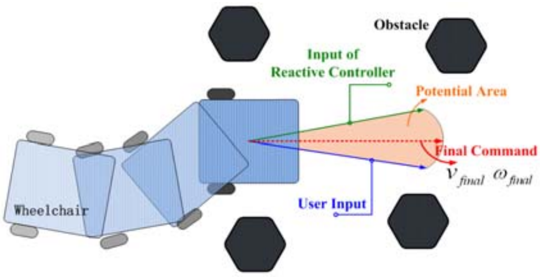


Fig.4 Instruction of control weight assigning

The shared control problem can be written in a standard multi-objective optimization problem form (Eq. (1)).

$$\begin{cases} \max\limits_{\kappa_1,\kappa_2}(safety(v(t),\omega(t),Obs\_Info)) \\ \max\limits_{\kappa_1,\kappa_2}(comfort(v(t),\omega(t),Obs\_Info)) \\ \max\limits_{k_1,k_2}(obedience(v(t),\omega(t),Obs\_Info)) \\ s.t. \\ v(t)=v_{user}(t) \\ \omega(t)=\kappa_1\omega_{user}(t)+\kappa_2\omega_{mach}(t) \\ \kappa_1+\kappa_2=1 \\ \kappa_1,\kappa_2\geq 0 \end{cases} \tag{1}$$

Where, *v(t)* represents the speed to be sent to the SMC. To avoid violent change in motion command, we define ω as a linear combination of ω_{use} and ω_{mach}. This equation means finding the user's control weight is equivalent to finding the proper k_1 and k_2 to maximize the three max(·) items under the restrictions stated after s.t..

The *safety* index needs to be able to reflect the possibility of a wheelchair colliding with obstacles. *safety* in this paper is defined as:

$$safety = 1-\exp(-\alpha \cdot dis) \tag{2}$$

Where, α is a constant. *dis* measured in millimeter represents the distance between the wheelchair and the nearest obstacle in its path.

Therefore, we decide to use a function of velocity change to evaluate *comfort* index. Like the definition of *safety*, we adopt the negative exponential function to normalize *comfort*.

$$comfort = \exp(-\beta|\omega - \omega_0|) \tag{3}$$

Where, β is a constant. Since the wheelchair's translational speed is given by user directly and it won't be changed unless a collision is about to happen, there is no component reflects translational speed change in Eq. (3).

The *obedience* index is used to evaluate the proximity between the user's control intention and the final motion command. *obedience* is calculated as:

$$obedience = \exp(-\gamma\left|\xi - \xi^*\right|) \tag{4}$$

Where, ξ^* is the orientation calculated from the user's input v_{user} and ω_{user}; ξ is the orientation determined by v and ω; γ is a constant. This index can make the wheelchair moving under the user's intention as long as he is able to maintain *safety* and *comfort*.

These three indices are usually contradictory to each other under normal circumstances. For example when a wheelchair is traveling through a crowd, it will be required to turn a big round for *safety*, but for *comfort* it is not allowed to do that. Therefore, there is no absolute optimum solution for Eq. (1). An evaluation function of this problem is needed to achieve an effective solution.

It was found that increasing a certain index which is already above a certain value will make the other two indices drop drastically. For example, enforcing *safety* to increase when it is already above 0.9 will make the wheelchair always choose the most spacious path, and the user will feel the wheelchair is not moving under his control at all. Therefore, a principle we proposed of solving this problem is: always improve the smallest index among the three. In accordance with this principle we choose the *minimax* method to simplify Eq. (1) to a single objective problem (Eq. (5)).

$$\begin{cases} \max\limits_{v,\omega}\left(\min(safety, comfort, obedience)\right) \\ s.t. \\ \omega \in [\omega_{user}, \omega_{mach}] \\ v = v_{user} \end{cases} \tag{5}$$

The algorithm's adaptation could be improved by using this *minimax* method. The precedence relation among indices will change naturally when facing different situations.

B. Autonomous Navigation Mode

In some structured environment, JiaoLong can carry users to navigate automatically. The map of environment is built through the human-guided method. The positions which are important to the users are marked through user's voice input during guidance. During navigation, through user's voice or touch screen input, JiaoLong can navigate to the destination automatically and avoid the dynamic obstacles[11]. In localization, the fundamental particle filtering method[12] is used.

The autonomous navigation mode adapt to the users who have difficulty and even no ability in powered wheelchair control. JiaoLong can carry them to navigate in hospital or welfare institute according to the user's or nurse's instructional input. It will lighten the burdens of nurse and the families of patients. In path planning, the classical A* algorithm[13] is used, and in trajectory following, the lane curvature method (LCM)[14] is used.

Fig.5 shows the process and the program frameworks of JiaoLong's autonomous navigation system. The program frameworks are consisted of three parts: Human-Robot Interface, Localization & Navigation, and Driver.

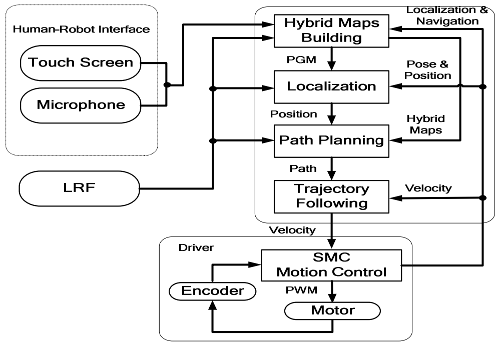


Fig.5 Frameworks of "JiaoLong" autonomous navigation system

1) Hybrid maps building

The area of application environment which the wheelchair is used will be very large. So, the navigation method[11] based on hybrid map is proposed. The hybrid map is built through the human guided method, it consisted of many small probabilistic grid maps (PGMs) and a global topological map (GTM). On every small PGM there are grid markers (GM) which are as the topological nodes. The grid markers are memorized through user's voice input during the human guided. The connection between neighbor PGMs is established through the GMs that stand for the same place. Then all the GMs and their connections are composed the GTM. The PGMs, GMs and their connections are all generated through human guidance (voice or screen-touch).

Human-Robot Interface unit is in Windows OS using human-guided method. Localization & Navigation unit is the controlling kernel in Linux OS. Driver unite is motor controlling system, which controls the motors through PWM and feeds the encoder data.

Using human-guided method, the algorithm of hybrid maps building is as Fig.6. Z_{Pose} is the orientation angle and $Z_{Position}$ is the coordinates supplied by encoder-based odometer. Z_L is environment information scanned by the LRF. The PGM is built using incremental maximum likelihood method (IMLM)[15]. The PGM's composition is shown as below:

$$PGM = \{\ SN,\ Map,\ GM,\ GM_p\ \} \tag{6}$$

Where, the GM and GM_p composition is:

$$GM,\ GM_p = \{\ SN,\ Position(x,\ \mathrm{y}),\ Pose(\theta)\ \} \tag{7}$$

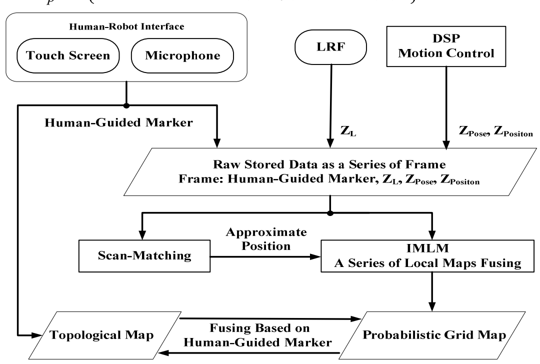


Fig.6 Hybrid map building algorithm

2) Improved Particle Filter Localization Algorithm

We proposed an improved particle filter localization algorithm which is based on the dynamic localizability matrix. This algorithm estimates the belief of laser range finder observations using the localizability matrix of observation model and the belief of the odometer data using the covariance matrix of prediction model. In order to increase robot localization precision, the proportion between of them is adjusted dynamically. During localization period, every particle's local map according to current LRF data is built and then compared with current PGM in order to obtain every particle's weight. Then the position could be achieved through summarizing of all the particles' weights based on Bayesian filter algorithm.

Framework of the improved particle filter localization based on localizability is shown on Fig7.

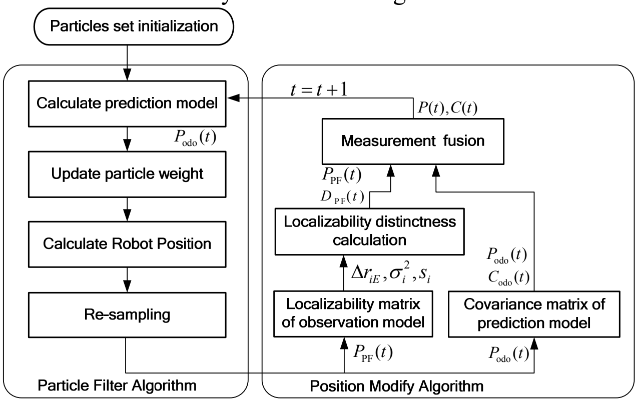


Fig.7 Framework of improved particle filter localization

On one hand, this algorithm estimates the belief of laser range finder observations using the localizability matrix of observation model. On the other hand, it estimates the belief of the odometer data using the covariance matrix of prediction model. Then based on these two indicators, the predicted robot position is modified according to the observed information.

- Localizability Matrix Estimation

The dynamic localizability indicator was introduced in paper[16]. The structure of map, noisy of sensor and the effect of the unknown obstacles are considered comprehensively. As show as equations (8)

$$\boldsymbol{D}(\boldsymbol{P})=\sum_i^n\left(\frac{1-s_i}{\sigma_i^2}\begin{bmatrix}\frac{\Delta r_{iE}^2}{\Delta x^2} & \frac{\Delta r_{iE}^2}{\Delta x\Delta y} & \frac{\Delta r_{iE}^2}{\Delta x\Delta\theta}\\ \frac{\Delta r_{iE}^2}{\Delta x\Delta y} & \frac{\Delta r_{iE}^2}{\Delta y^2} & \frac{\Delta r_{iE}^2}{\Delta y\Delta\theta}\\ \frac{\Delta r_{iE}^2}{\Delta x\Delta\theta} & \frac{\Delta r_{iE}^2}{\Delta y\Delta\theta} & \frac{\Delta r_{iE}^2}{\Delta\theta^2}\end{bmatrix}\right) \tag{8}$$

s_i: Effecter of unknown obstacles;

σ_i^2: Variance of LRF measurement;

$\frac{\Delta r_{iE}}{\Delta x}$, $\frac{\Delta r_{iE}}{\Delta y}$, $\frac{\Delta r_{iE}}{\Delta z}$: Measurement change of LRF after robot's movement in Δx, Δy and Δz.

The localizability matrix $\boldsymbol{D}_{PF}(\boldsymbol{P})$ of observation model for particle filter algorithm can be defined as:

$$\boldsymbol{D}_{\mathrm{PF}}(\boldsymbol{P}) = k_{\mathrm{P}}\boldsymbol{D}(\boldsymbol{P}) \qquad k_{\mathrm{P}}<1 \tag{9}$$

$\boldsymbol{P}$: robot position (assume to the result of particle filter);

k_P is defined as:

$$k_{\mathrm{P}} = 1-\frac{1}{\log n_{\mathrm{P}}}$$

- Covariance Estimation of Prediction Model

The kinematic model of differential-drive mobile robots[17] is:

$$\begin{bmatrix}v\\ \omega\end{bmatrix}=\boldsymbol{C}\begin{bmatrix}\omega_R\\ \omega_L\end{bmatrix};\quad \boldsymbol{C}=\begin{bmatrix}L_R/2 & L_L/2\\ L_R/b & -L_L/b\end{bmatrix} \tag{10}$$

v, ω: robot velocity and angular velocity;

ω_R, ω_L: angular velocities of right and left wheels

L_R, L_L: the radio of the right and left wheels;

b: the distance of right and left wheels.

Then the odometer of robot can be calculated as:

$$\begin{pmatrix}x(t)\\ y(t)\\ \theta(t)\end{pmatrix}=\begin{pmatrix}x(t-1)+\Delta t v(t-1)\cos\left(\theta(t-1)+\Delta t\frac{\omega(t-1)}{2}\right)\\ y(t-1)+\Delta t v(t-1)\sin\left(\theta(t-1)+\Delta t\frac{\omega(t-1)}{2}\right)\\ \theta(t-1)+\Delta t\omega(t-1)\end{pmatrix}$$

$$=\begin{pmatrix}x(t-1)+\frac{L_{\mathrm{R}}+L_{\mathrm{L}}}{2}\cos\left(\theta(t)+\frac{L_{\mathrm{R}}-L_{\mathrm{L}}}{2b}\right)\\ y(t-1)+\frac{L_{\mathrm{R}}+L_{\mathrm{L}}}{2}\sin\left(\theta(t)+\frac{L_{\mathrm{R}}-L_{\mathrm{L}}}{2b}\right)\\ \theta(t-1)+\frac{L_{\mathrm{R}}-L_{\mathrm{L}}}{b}\end{pmatrix} \tag{11}$$

At time t, the input of odometer can be assumed as:

$$u(t)\equiv\begin{bmatrix}L_{\mathrm{R}}(t) & L_{\mathrm{L}}(t) & b\end{bmatrix}^{\mathrm{T}} \tag{12}$$

The error of mobile robot odometer prediction model can be caused by many reasons: payload, skidding, encoder error and etc.

There is an assumption that all of the input variables are independent and their errors are Gaussian distribution. Then the covariance matrix $\boldsymbol{Q}(t)$ of $u(t)$ is:

$$\boldsymbol{Q}(t)=\begin{bmatrix}\sigma_{\mathrm{R}}^2(t) & 0 & 0\\ 0 & \sigma_{\mathrm{L}}^2(t) & 0\\ 0 & 0 & \sigma_{\mathrm{b}}^2(t)\end{bmatrix} \tag{13}$$

The increment of odometer is:

$$\Delta\boldsymbol{P}(t)=\begin{bmatrix}\Delta x(t)\\ \Delta y(t)\\ \Delta\theta(t)\end{bmatrix}=\begin{bmatrix}x(t)-x(t-1)\\ y(t)-y(t-1)\\ \theta(t)-\theta(t-1)\end{bmatrix}$$

$$=\begin{bmatrix}\frac{L_{\mathrm{R}}+L_{\mathrm{L}}}{2}\cos\left(\theta(t)+\frac{L_{\mathrm{R}}-L_{\mathrm{L}}}{2b}\right)\\ \frac{L_{\mathrm{R}}+L_{\mathrm{L}}}{2}\sin\left(\theta(t)+\frac{L_{\mathrm{R}}-L_{\mathrm{L}}}{2b}\right)\\ \frac{L_{\mathrm{R}}-L_{\mathrm{L}}}{b}\end{bmatrix} \tag{14}$$

The covariance matrix of increment of odometer can be calculated as follows[18]:

$$\mathrm{cov}\left(\Delta\boldsymbol{P}(t)\right) = \frac{\partial\Delta\boldsymbol{P}(t)}{\partial u(t)} Q(t) \frac{\partial\Delta\boldsymbol{P}(t)}{\partial u(t)}^{\mathrm{T}} \tag{15}$$

The derivation [16] of equation (15) is:

$$\frac{\partial\Delta\boldsymbol{P}(t)}{\partial u(t)} = \begin{bmatrix} \frac{\cos\theta}{2}-\frac{L\sin\theta}{2} & \frac{\cos\theta}{2}+\frac{L\sin\theta}{2} \\ \frac{\sin\theta}{2}+\frac{L\cos\theta}{2} & \frac{\sin\theta}{2}-\frac{L\cos\theta}{2} \\ \frac{1}{b} & -\frac{1}{b} \end{bmatrix} \tag{16}$$

$$L=\frac{L_{\mathrm{R}}+L_{\mathrm{L}}}{2},\quad D=\frac{L_{\mathrm{R}}-L_{\mathrm{L}}}{2},\quad \theta=\theta_k+\frac{L_{\mathrm{R}}-L_{\mathrm{L}}}{2b}$$

Then, the covariance matrix of prediction model is:

$$\boldsymbol{C}_{\mathrm{odo}}(t) = \boldsymbol{C}(t-1) + \mathrm{cov}\left(\Delta\boldsymbol{P}(t)\right) \tag{17}$$

- Position Modify Algorithm

At time t, based on prediction model robot position can be calculated as:

$$\boldsymbol{P}_{\mathrm{odo}}(t) = \boldsymbol{P}(t-1) + \boldsymbol{f}_{\mathrm{odo}}\left(\Delta u(t)\right) \tag{18}$$

$\boldsymbol{P}(t\text{-}1)$: robot position at time t-1;

$\Delta u(t)$: input of odometer at time t-1;

f_{odo}: kinematic model;

$\boldsymbol{C}_{\mathrm{odo}}(t)$: covariance matrix of prediction model.

Based on prediction model, the new particle set $\{\boldsymbol{P}_{\mathrm{pf}}(t), w(t)\}$ will be update.

$\boldsymbol{P}_{\mathrm{pf}}(t)$: position of particles;

$w(t)$: weight of particles。

The robot position can be calculated through new particle set:

$$\boldsymbol{P}_{\mathrm{PF}}(t) = \sum_i w^i\left(\boldsymbol{z}(t),M\right)\boldsymbol{p}^i_{\mathrm{PF}}(t) \tag{19}$$

$\boldsymbol{z}(t)$: observation of LRF;

$\boldsymbol{M}$: map information.

The predictive position is modified based on observation, it can be calculated：

$$\Delta\boldsymbol{P}_{\mathrm{obs}}(t) = \boldsymbol{P}_{\mathrm{PF}}(t) - \boldsymbol{P}_{\mathrm{odo}}(t) \tag{20}$$

At time t, the predictive position is modified based on the localizability matrix $\boldsymbol{D}_{PF}(\boldsymbol{P})$ of observation model and the covariance matrix of prediction model [19]:

$$\boldsymbol{P}(t) = \boldsymbol{P}_{\mathrm{odo}}(t) + \frac{\boldsymbol{D}_{\mathrm{PF}}(t)}{\boldsymbol{D}_{\mathrm{PF}}(t)+k_1\boldsymbol{C}^{-1}_{\mathrm{odo}}(t)}\Delta\boldsymbol{P}_{\mathrm{obs}}(t) \tag{21}$$

k_1: proportion between the localizability of observation model and the covariance matrix of prediction model.

After the re-sampling period, the particle set is modified in the differencing direction of robot position and predictive position:

$$\{\boldsymbol{p}'(t)\} = \{\boldsymbol{p}(t)\} + P(t) - P_{\mathrm{PF}}(t) \tag{22}$$

According to equation (21), in crowed environment, the localizability is lower. Robot localization relies mainly on the predictive information of odometer. In loose environment, the particle set is modified according to the predictive position of observation model. The accumulative error is eliminated. So, the localizability is higher.

III. Simulation Platform

A. Platform Structure

Fig.8 shows the software structure of the proposed development and simulation platform. This platform combines a real joystick, virtual wheelchair and virtual world, and achieves a real-time simulation. In a loosely coupling way, client program accesses to the platform through a topic-based pub/sub communication, including getting sensors' data and controlling the speed of wheelchair.

The implementation of this platform makes use of Gazebo and ROS: the calculations of dynamics and sensors are vastly based on the physical engine ODE and rendering engine OGRE in Gazebo; communication between client programs is implemented in ROS. The modules excluding GUI are dynamically loaded, and can be modified or replaced conveniently according to the specified wheelchair simulated (especially sensors). Since the intelligent wheelchairs are ever-changing, a platform whose code is not available may not works in such situations. Following the principles established by Gazebo and ROS, this platform is completely open source, and maintains a simple API for developer and a friendly GUI for end user.

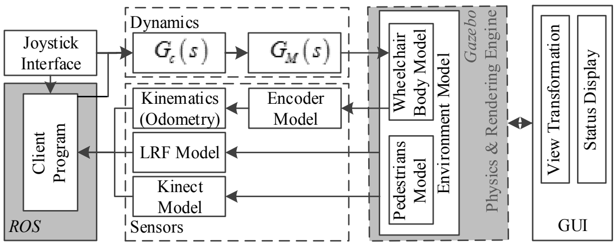


Fig.8 Software structure of the development and simulation platform

B. Platform Model

The 3D model of wheelchair was draw in SolidWorks, which is shown in Fig.9. In order to control the six joints of the wheelchair, the body, shafts, front and rear wheels are drawn separately.

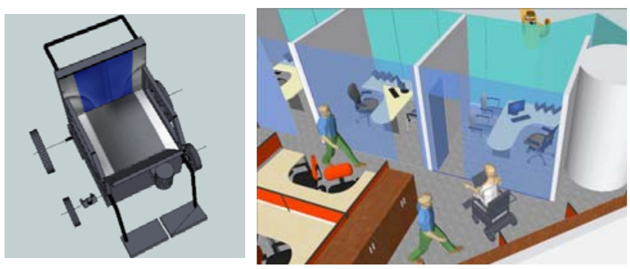
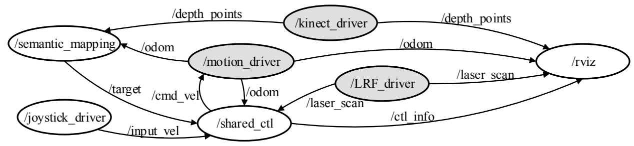


Fig.9 3D wheelchair model and world model

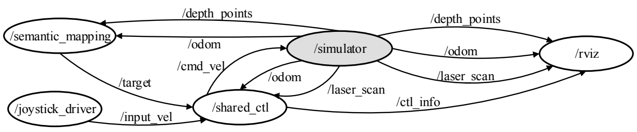
In order to simulate the actual scene, both the static and dynamic characteristics of world are modeled. The static characteristics refer to stationary objects such as furniture, doors and walls. Described in the COLLADA format, these objects are composed of complex shapes and materials with transparency and texture. Some popular 3D modeling software including Google SketchUp and 3D Studio Max are used to synthesize them in a same scene. As the dynamic characteristics, random pedestrians are modeled in the scene, and their behavior is roaming with obstacle avoidance.

C. Development of Client Program

The design and optimization of new algorithms for smart wheelchair is somehow difficult with the limitation of test environments and available users. In this situation, the proposed platform can shorten the development cycle and reduce the experimental costs. This part took the semantic map based shared controller[20], which was tested and verified on the platform, as an example.



a) The software structure in a real scene.



b) The software structure in a virtual scene.

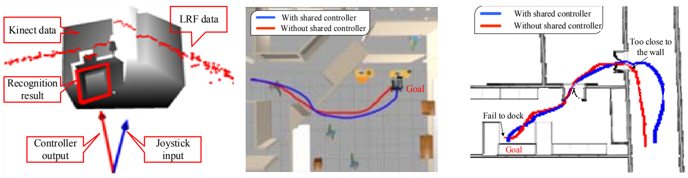
Fig.10 Software structures of the system combined with the shared controller

The software structures of the system in real and virtual scene can be found in Fig.10. Obviously, the only difference between them is that the simulator replaces the hardware related modules in the real scene. In this sense, the steps to develop a client program can be described as follows:

a) If the drivers for hardware exist, remove them;

b) Implement the program in ROS (If already has implemented, skip this step);

c) Change the topic name of the program if necessary.



a) Schematic diagram b) Simulated trajectories c) Actual trajectories

Fig.11 Results of the wheelchair with and without controller

Fig.11-a illustrates how the controller works. It builds a local 3D semantic map which provides the semantic and geometrical information through Kinect. Based on this map, the controller chooses different behaviors according to user intention and achieves an object related navigation with obstacle avoidance through LRF. As shown in Fig.11-b, the user tried to drive and dock the wheelchair at the destination desk from the initial position. Both of the trajectories of the wheelchair with and without the application of the controller can be found in Fig.11-b. The wheelchair with the controller combining the semantic information and the position of obstacles can pass through the doorway vertically and dock precisely. On the opposite, the wheelchair without the controller is too close to the wall to pass, and cannot approach the table and fails to dock.

Compared with the trajectories in a real scene shown in Fig.11-c, the characteristics that has been described above are similar. Some differences between them should be pointed out as well: a) the speed of the simulation is slower than a real-time one due to the large amount of Kinect's data; b) the results of the simulation is seemly better with less noise of sensors (for example, the slip of wheels is ignored in the virtual scene). Despite these drawbacks, the platform can be a useful tool for algorithm design with sensors and dynamics implemented in the platform.

IV. CONCLUSION

This paper shows the prototype, hardware architecture, improved algorithm in localization, experiments and tests results of the intelligent wheelchair called JiaoLong. Fundamental ideas and methods used in the dynamic shared control mode and the autonomous navigation mode are also proposed. In dynamic shared control mode, three indices including safety, comfort and obedience are designed to evaluate the wheelchair's performance. Through the weights dynamic assignment of the user and the reactive controller by user-adapted control weight assignment, the maximum indices is obtained. In autonomous navigation mode, the hybrid map is easy to be built using human-guided method. Besides, our improved particle filter localization algorithm based on the dynamic localizability matrix shows effectiveness in localization. On top of that, the path planning and trajectory following algorithms ensure that JiaoLong can navigate in the large scale environment. On the end, a development and simulation platform for algorithm design is designed and implemented. The application for algorithm design is presented. Experiments show the effectiveness of the proposed platform, and it can be used for intelligent wheelchair algorithm design.

REFERENCES

[1] E. Prassler, J. Scholz and P Fiorini. A robotics wheelchair for crowded public environment. IEEE Robotics Automation Magazine. Vol:8(1), pp.38-45, 2001.

[2] G.Bourhis, P.Pino. Mobile Robotics and Mobility Assistance for People with Motor Impairments: Rational Justification for the VAHM Project. IEEE Trans. Rehab. Eng, 1996, Vol.4, pp.7-11.

[3] M. A. Gignac, C. Cotta, E.M. Badley. Adaptation to chronic illness and disability and its relationship to perceptions of independence and dependence. J Gerontol B Psychol Sci Soc Sci. 2000, Vol.55 (6), pp.362-372.

[4] A. Lankenau, T. Rofer. A Versatile and Safe Mobility Assistant. IEEE Robotics & Automation Magazine, 2001, Vol.8 (1), pp.29-37.

[5] G. Bourhis, O. Horn, O. Habert, A. Pruski. An Autonomous Vehicle for People with Motor Disabilities. IEEE Robotics & Automation Magazine, 2001, Vol.88 (1), pp.20-28.

[6] J. C. Wang, W. D. Chen. Design and Implementation of a Multi-Sensor Based Autonomous Wheelchair, Jounal of Nanjing University of Science and Technology (Natural Science), Vol.33 Sup, pp.104-409, Aug. 2009.

[7] T. Gomi, A. Griffith. Developing Intelligent Wheelchairs for the Handicapped Wheelchair. Assistive Technology and Artificial Intelligence, 1998, pp.151-178.

[8] Q Li, W. D. Chen and J. C. Wang. Dynamic shared control for human-wheelchair cooperation, in Proc. 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, May 9-13, 2011.

[9] S. P. Levine, D. A. Bell, L. A. Jaros, and etal. The NavChair assistive wheelchair navigation system, IEEE Transactions on Rehabilitation Engineering, vol.7, no. 4, 1999, pp. 443-451.

[10] J. Borenstein, Y. Koren. The vector field histogram-fast obstacle avoidance for mobile robots, IEEE Transactions on Robotics and Automation, vol. 7, no. 3, 1991, pp. 278-288.

[11] Y. Wang, W. D. Chen and J. C. Wang. Hybrid map-based navigation for intelligent wheelchair, in Proc. 2011 IEEE International Conference on Robotics and Automation , Shanghai, China, May 9-13, 2011.

[12] F. Dellaert, W. Burgard, D. Fox, et al., Monte Carlo Localization for Mobile Robots, in Proceedings of the IEEE International Conference on Robotics and Automation, vol 2, pp. 1322–1328, Detroit, MI, May 1999.

[13] P. E. Hart, N. J. Nilsson and B. Raphael, A Formal Basis for the Heuristic Determination of Minimum Cost Paths, IEEE Transactions of Systems Science and Cybernetics, Vol: ssc-4(2), pp. 100-107, July 1968.

[14] N. Y. Ko, R. G. Simmons. The Lane-curvature Method for Local Obstacle Avoidance, in Proc. 1998 IEEE International Conference on Intelligent Robotics and Systems, Victoria, BC, Canada, 1998.

[15] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust Monte Carlo Localization for Mobile Robots, Artificial Intelligence, Summer 2001.

[16] Wang W, Chen W D, Wang Y. Localizability Estimation for Mobile Robot with Use of Probabilistic Grid Map [J]. Robot, 2012, 34 (4): 485-491,512.

[17] Antonelli G, Chiaverini S, Fusco G A. Calibration Method for Odometry of Mobile Robots Based on the Least-squares Technique: Theory and Experimental Validation [J]. IEEE Transactions on Robotics, 21(5), 2005: 994-1004.

[18] Kleeman L. Advanced Sonar and Odometry Error Modeling for Simultaneous Localization and Map Building [C]//IEEE International Conference on Intelligent Robots and Systems, Las Vegas, NV, USA: IEEE, 2003: 699-704.

[19] Roecker J A, McGillem C D. Comparison of Two-sensor Tracking Methods Based on State Vector Fusion and Measurement Fusion [J]. IEEE Transactions on Aerospace and Electronic Systems, 1988(24): 447-449.

[20] Wei, Z.X., Chen, W.D., Wang, J.J.: 3D semantic map-based shared control for smart wheelchair. In: Su, C.-Y., Rakheja, S., Liu. H.H. (eds.) Intelligent Robotics and Applications. LNCS, vol. 7507, pp. 41-51. Springer, Heidelberg (2012)