Development of a Realistic Simulator for Robotic Intelligent Wheelchairs in a Hospital Environment

Rodrigo A.M. Braga[1,2], Pedro Malheiro[1,2], and Luis Paulo Reis[1,2]

[1] FEUP – Faculty of Engineering of the University of Porto, Rua Dr. Roberto Frias, s/n 4200-465, Porto, Portugal
[2] LIACC - Artificial Intelligence and Computer Science Lab., University of Porto, Portugal
rodrigo.braga@fe.up.pt, pedro.m.malheiro@gmail.com, lpreis@fe.up.pt

Abstract. Nowadays one can witness the increase of the world population carrying some kind of physical disability, affecting locomotion. With the objective of responding to numerous mobility problems, various intelligent wheelchair related projects have been created in the last years. The development of an Intelligent Wheelchair requires a lot of testing due to the complexity of the algorithms used and the obligation of achieving a failproof final product. This paper describes the some need for an Intelligent-Wheelchair specific simulator as well as the requirements of such a simulator. The simulator implementation, based on "Ciber-Mouse" simulator, is also described with emphasis on analyzing the limitations concerning intelligent wheelchair simulation using this adapted simulator. The changes applied on the existing software and the difficulties of robotic simulation development are described in detail. Experimental results are also presented showing that not only the simulator reveals flexible simulation capabilities but, also, enabled to validate the algorithms implemented in the physical intelligent wheelchair controlling agent.

Keywords: Simulation, Robotics, Intelligent Wheelchair.

1 Introduction

Although not obvious, conventional electric wheelchairs have limitations that difficult and may even prevent its normal usage for some people, depending on their disability [1] [2]. Afflictions that affect arm and hand coordination or even vision are sure to benefit from intelligent wheelchairs, which will give some independence back to these patients.

This scenery enables the need for Intelligent Wheelchairs (IWs). An IW will, through computational capabilities, sensors, communications and motor control, automatically (and in an intelligent fashion) provide services for its user [3] [4]. A few examples are: Multiple user interfaces for control and ordering (voice, facial recognition, joysticks, keyboard, etc.), aided driving (in case of miscalculation of the patient's manual drive, the IW will disable human control: e.g.: if a collision is imminent) or even opening a door, as well as many other domotics applications, can be dealt within the IW's communication capabilities.

J. Baltes et al. (Eds.): RoboCup 2009, LNAI 5949, pp. 23–34, 2010.

After applying the hardware add-ons – such as sensors – an IW's capabilities and reliability lie on its control software. The control algorithms must be thoroughly tested. The control software should undertake Multi IW scenario tests with dynamic obstacles, in order to ensure reliability on the developed algorithms. Creating a real life hospital-like situation is not conceivable therefore this is the point where a simulator will be required. It can model not only any kind of map but it can even provide the IW with all sensor information as a real situation would.

In a generic definition [5], simulation is the imitation of a real system, in function of the time. It is used to describe and analyze the behavior of a system and can draw conclusions from "what-if" scenarios. Among the advantages of simulation, we can refer the possibility of testing every aspect of a proposal for modification or addition without endangering real resources; time compression, in order to obtain all the consequences of a modification in a shorter lapse of time; Finding errors and problems on a system becomes simpler. Finding the cause of a certain occurrence is facilitated for it is possible to isolate the simulation to the occurrence and analyze it in detail; Lower costs: typically the cost of a simulation compared to a real test is around 1%; Requirement identification: it is possible to identify the needs in terms of hardware. For example: one can, through simulation, find what will the necessary resolution be for a certain sensor.

Having justified the need for a simulator, the challenge is to find what needs to be simulated and how. An IW-specific robotic simulator must be able to realistically mimic the physics of every aspect of the wheelchair's environment and its own characteristics namely its form, dimensions, motor dynamics, sensors and communications. Moreover, the simulation must take care of all interaction between the wheelchair and the world: detect collisions, return sensor values and calculate IWs' positions.

Additionally, being visualization one of the main objectives of an IW simulation, it is important to have a simulation viewer that can show the main information: IWs' sizes, locations and sensors' values as well as the environment: map with its walls and doors.

2 Related Work

Two projects stood out during literature review of intelligent wheelchair simulation: the Bremen Autonomous Wheelchair [6] (BAW) and the *Vehicle Autonome pour Handicap*és Moteurs [7] (VAHM).

Combined, both BAW and VAHM were motivated by reasons shared with Intellwheels Simulator project: the need to test hardware platform and its performance, without submitting patients to dangerous situations. Despite this, there are conceptual differences in architecture and on the objectives. While these projects intend only to aid the development of a single isolated wheelchair, Intelwheels is multi-agent based, in the sense that a dynamic, more complex environment with multiple intelligent and collaborative objects is intended. Furthermore, BAW simulation segregates completely real and virtual worlds, leaving no room for augmented reality model (which is a critical conceptual design of Intellwheels).

3 Intellwheels Simulator

The original "Ciber-Mouse" Simulator[7][8] is a software developed for a robotic competition, held at University of Aveiro and it was the base of this project. The objective of the participants is to create a robotic agent that would control their "mouse" (a virtual robot) through a maze to find a beacon (the objective).

This software already had various characteristics very usefull for IW simulation such as virtual differential robots (two wheels) and numerous sensors (e.g. compass and proximity sensors). It also contained a 2D simulation viewer, which was specific for the competition[9].

Being based on "Ciber-Mouse" the new simulator has a similar conceptual architecture, consisting of a central simulation server, to which every agent (robotic or viewing agents) must connect to (Figure 1).

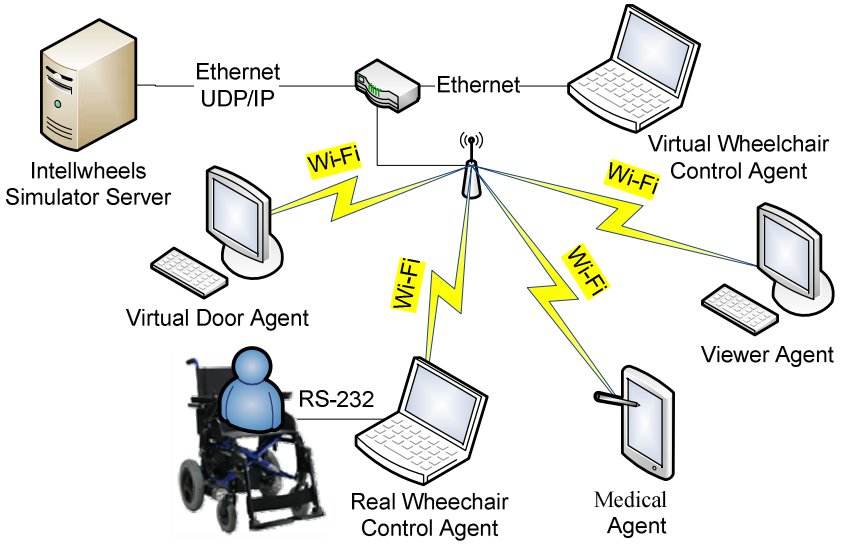


Fig. 1. Intellwheels Simulator Architecture

The most relevant work was made on the Robot Modeling and on the Collision Detection modules which will be detailed further ahead. Evident in the image is the connection of agents for controlling virtual wheelchairs and agents that control real wheelchairs. This possibility of interaction between the real and virtual worlds creates a mixed reality (MR) environment[10]. The definition of MR lies in a mid-stage point between a purely real world and a virtual one.

It is this MR support that stretches this simulator's capabilities beyond single algorithm testing, as it is now possible to see a real IW react to a more dynamic scenario (with moving obstacles, complex maps and other intelligent agents). It all depends on which and how many agents connect, for the simulator itself will not limit their size or movement.

The simulator expects a real wheelchair agent to provide real world data concerning its position, as illustrated in Figure 2.

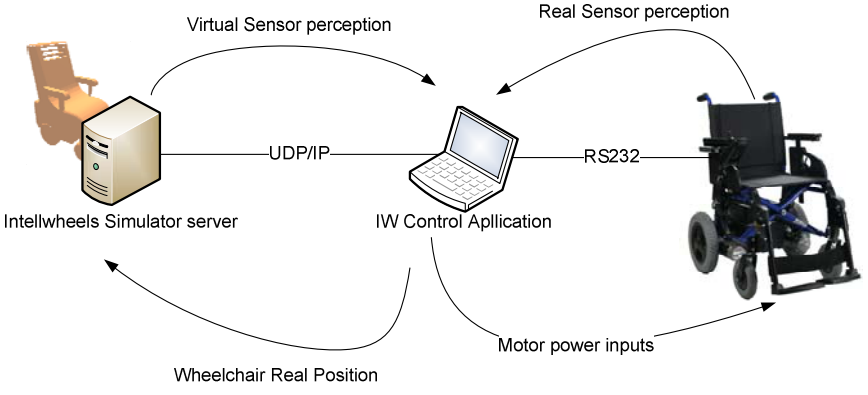


Fig. 2. Mixed Reality information exchange

The agent must, through encoders calculate the real wheelchair's position and then send it to the simulation server. The simulator will then place virtual sensors onto that position and return their perception to the agent. With all the sensor information, the robotic agent will then calculate the motor power inputs which are to be sent only to the real wheelchair.

Having defined the conceptual architecture of the simulator, the starting point for the adaption of the DCOSS version of "Ciber-Mouse" software was the conversion from circular to rectangular body robots. Moreover, the parameters that define the size and the center of movement are now configurable, which allows the modeling of different types of robots (not only wheelchairs).

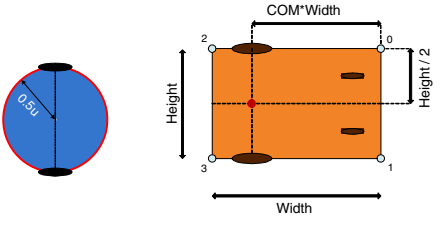


Fig. 3. Circular to rectangle shape modifications

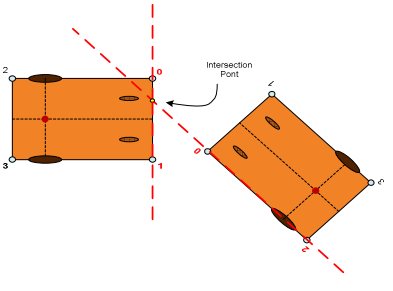


Fig. 4. Robot-Robot collision check

The main usage of the robot's body is in the collision detection verification. A robot's shape has to be defined in mathematical equations that will enable the detection of intersection with other objects. In this simulator there will only be modeled 2 types of objects: walls and robots. Therefore collision checking will only have to be performed with these two types.

Originally, for Robot-Robot collision checking, the "Ciber-Mouse" simulator checked whether the distance that separates the robots' centers (thought X and Y coordinates) was smaller than two times a robot's radius (all robots were circular with the same radius). This simple algorithm is not applicable for different radius robots neither for rectangle shaped robots. The wheelchairs' size and position on the map were now modeled using four parameters: Center of movement point, the wheelchair's orientation angle, the width and height. Through this information all the robots' corners coordinates can be calculated. Using this information the new collision detection algorithm is as follows:

1. Using pairs of corners as line segment defining points, it is calculated an equation for one line segment for each robot.
2. The intersection point of the two lines is calculated. If lines are parallel no point is calculated for there is no intersection.
3. Both X and Y coordinates are checked to find whether they are located within each robot line segment. If so, then there is a collision between the two robots.
4. This process is repeated until the 4 lines of each robot are checked with the lines of every other robot.

In Figure 4 the lines cross at a point that only belongs to one of the wheelchairs, therefore no collision is detected.

An additional modification was made on the dynamic characteristics of the motors' acceleration curves. Since the original software was designed to ensure all robots were equal, every robot connected had to had the same dynamic characteristic. In this IW simulation environment it is expected that different robots connect. Equation (1), show below, was applied to allow the curve definition.

$$output_n = (1 - AccelerationCurve) * input_n + AccelerationCurve * output_{n-1} \quad (1)$$

where $output_n$ is the new motor power output, *AccelerationCurve* is a value between 0.00 and 1.00, defining the slope of the acceleration curve, and $output_{n-1}$ is the power value from the previous period.

Similarly to size and center of movement characteristics, a new robot registration parameter was implemented to allow each robot to define their curve. This value will affect the robot speed calculation consequently affecting the robot's next position calculation by the simulator.

The proximity sensor positioning was the next functionality to be adapted. Originally, the infra red sensor could only be positioned in the perimeter of a circle, with a fixed cone of sight and a fixed direction, radial to the robot. To be true to the rectangular form, the sensor definition was now made by X and Y coordinates, relatively to the robot center, and both the cone of sight and the direction were redefined. All these parameters are now configurable by the agent, at the time of registration with the simulator. Moreover, the sheer modification of a configurable cone enables the agent to register different proximity sensors. A wider cone (around 50-70 degrees) would resemble a sonar proximity sensor whereas as thinner cone (1-10 degrees) would be more similar to an infrared proximity sensor.

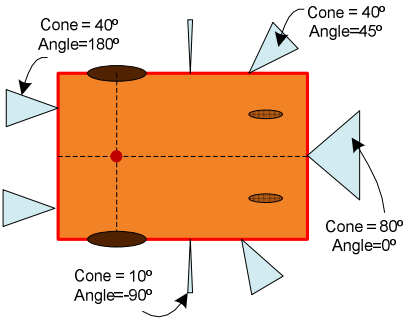


Fig. 5. Sensor definitions

An important objective of this IW simulation was to provide augmented reality scenery. In such an environment, a real IW (connected to the simulator) would be able to interact with virtual objects. Virtual wheelchairs would be able to detect its presence and virtual sensors could be attached to the real chair and provide it with additional information of the world around it. A very simple example of an application for this functionality is the correction of real sensors' errors. A Sonar proximity sensor is likely to have difficulty in distinguishing a table despite being in its normal range of sight. Depending on the shape of the legs, the height of the table and even on its color, acknowledging the presence of an object may fail. If the simulator has the table object drawn on the world, virtual sensors will report information about its location for they are not bound to the physical restrains as real sensors. The identification of a real chair is now made through a new XML tag, defined at the robot registration. A "Type" tag (later stored onto each robot's information within the simulator) will indicate whether the chair is real or simulated. Finaly, in terms of algorithms inside, the main change for mixed reality support is in the section of the code where the simulator calculates and commits the next robot position. Unlike a simulated IW, the real chair's position is not calculated by the simulator (through the left and right power orders given by the agent). Instead, the real IW will provide its current position and orientation, ideally once every simulation cycle, as previously illustrated on Figure 2.

4 Simulation Visual Representation

Visualization is the foundation for human understanding, as we process graphic information in a preconscious, involuntary fashion, similar to breathing**[11]**. In spite of its importance it is critical to ensure quality in a few elements, when developing simulation graphics.

Keeping this in mind, a visualization application was developed – Intellwheels Viewer. On a conceptual sense, it consists of five modules:

- Main Form – Responsible for the aggregation of all the information, allowing it to be transferred between the other modules. It also handles the initial interaction with the user, including configuration parameters of the other modules;

- Communications – This module implements the IP and UDP protocols, for physical connection with the simulator, and XML message parsing, for simulation data extraction;
- Robot – This module will store information concerning each robot: size, center of movement, position, orientation and collision status;
- 2D Visualization – This module will, through the map and robots' information, draw a two dimensional representation of the simulation;
- 3D Visualization – To represent a three dimensional environment, this module uses OpenGL technology **[12]** realistically displays the simulation. It has various camera options, from free camera movement to a 1st person fixed point, similarly to how a real driver of a wheelchair would see the world around.

Every simulator step, new world state data, including robot information, is sent to the viewers and is at that time that the graphical representation is updated. Figure 6 shows a simulation with three wheelchair agents, a table agent and a and door agent, on free camera viewing.

Fig. 6. 3D viewing, on free camera mode

5 Simulation Tests

To validate the performance of the simulator and confirm its importance for intelligent wheelchair development, a series of tests was performed. These tests were based on driving analysis from real, virtual and augmented reality runs, which were compared against each other. The floor of the building were (Deleted for blind review) is

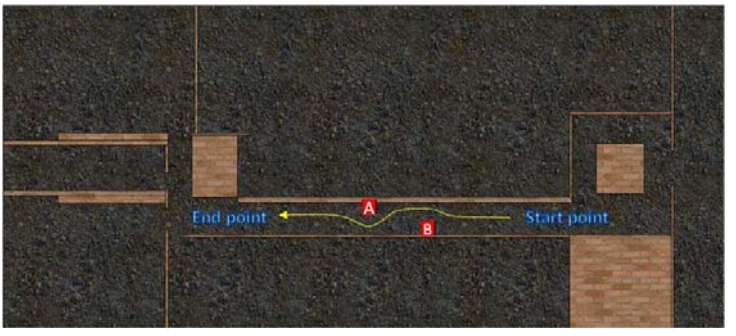


Fig. 7. Intellwheels modeled map of floor with test obstacles, in 2D mode of new viewer

set was modeled into Intelwheels XML map format, to allow a direct comparison between real and virtual tests. Figure 7 shows the simulated scenario (drawn from the CAD file) of the floor where the testes were performed.

An experiment was performed to verify the obstacle avoidance algorithm on the controlling software as well as the augmented reality environment as well (Figure 8). Obstacles (A and B block illustrated in Figure **7**) were placed both on the real and on the virtual environments, in the exact same space, with the same dimensions. The wheelchair should drive 15 meters in the X coordinate while avoiding the obstacles.

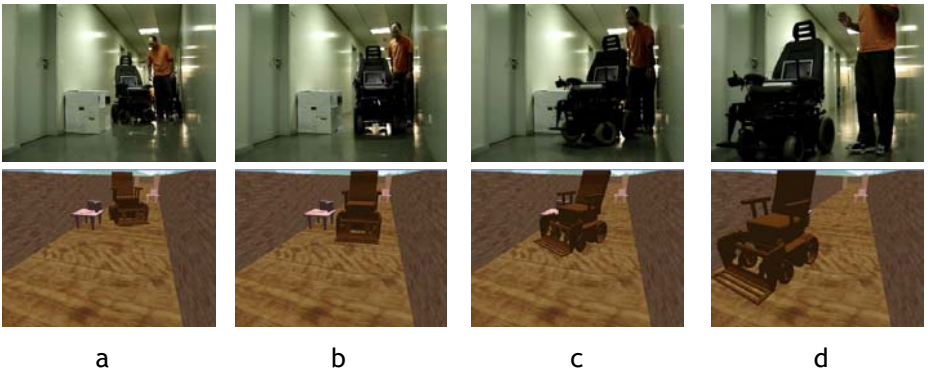


Fig. 8. Augmented Reality test

Using virtual sensors, the real wheelchair (in augmented reality mode) was able to autonomously avoid an obstacle placed in the virtual world. The simulator placed the wheelchair in its true position and the controlling agent correctly used the virtual sensors for motor power orders.

In this same experiment set-up, additional tests were taking, with both real and virtual wheelchairs and using manual and autonomous control. The results of the tests are illustrated in Table 1.

Table 1. Obstacle avoidance test results

Operation mode	Performance	Collision Count
Real/Manual	!	0
Virtual/Manual	✓	0
Real/Autonomous	!	1
Virtual/Autonomous	✓	0
Augmented/Autonomous	✓	0

Performance of these tests was based on whether the wheelchair reached the final point, if it collided and on position logs of the controlling software. The results showed that simulation-aided algorithm testing performed better than purely real tests. The main reason for this was the errors in odometer-based positioning (which accumulates error) and sonar noise which sets the chairs' controlling software into erratic decisions. It is also noticeable that the behavior of the wheelchair is almost equal in virtual and augmented reality modes.

Despite problems with real sensors, this simulation's value was proved with the success of the virtual environment tests. The control algorithm is correctly implemented, which is a conclusion that, without simulated testing, could not have been reached.

In order to test the flexibility of the simulator and of the new viewer and to verify the correct implementation of a developed door agent a new experiment was prepared: ordering a chair to move straight forward, through the virtual door. The door agent itself was configured with height=0.1m, width=1.0m and COM=0.99. Two proximity sensors were defined at each side of the door, to detect approaching objects, as illustrated in Figure 9.

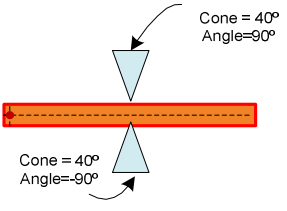
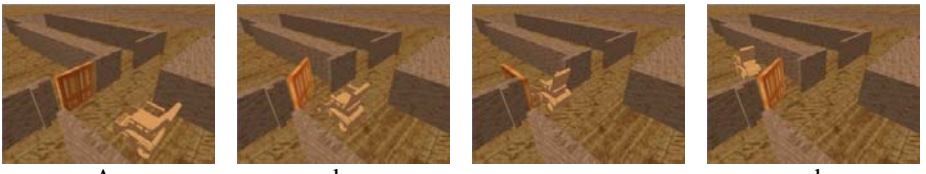


Fig. 9. Representation of the modeled door

Using a robotic agent, a virtual wheelchair was connected to the simulator and the test of door automatic opening was executed. Figure 10 is a series of print screens of the Intellwheels 3D viewer, during this automatic door test. The IW agent ordered the chair to move forward, regardless of what its own proximity sensors detect. On the other hand, the door agent was programmed to open if an object was detected and close only when the sensors stop detecting.



A b c d

Fig. 10. Automatic opening of the door

Having the capability of agent to agent communication, through very simple and effective XML messaging, IWs could autonomously "tell" each other to follow. Figure 11 shows an example where one chair is leading the way. The other IWs receive the first chair's communication of last point and follow the orders.

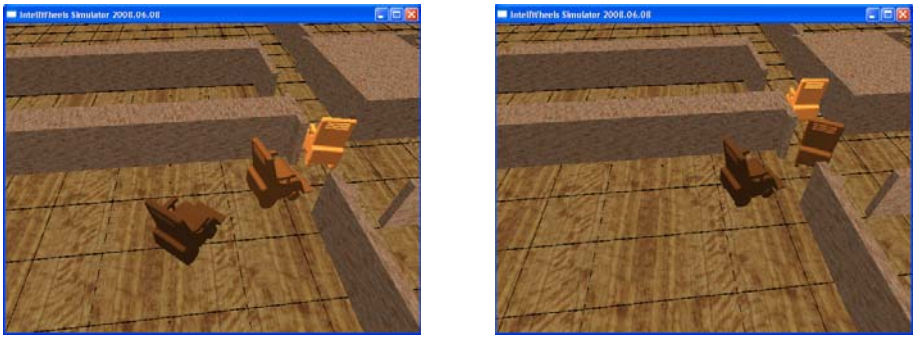


Fig. 11. Intelligent wheelchair communication

The following XML message is exchanged between the chairs:

```
<Actions …..
<Say><![CDATA[msg]]></Say> </Actions>
```

At a later stage of system developing, we have assembled a simple Mixed Reality Rehabilitation Lab. Using this setup, the virtual and augmented reality tests and simulations could be performed with increased realism, as shown in Figure 12.

Fig. 12. Mixed Reality Rehabilitation Laboratory

With such a setup it is now possible to supervise and monitor real and augmented reality tests and also provide a realistic simulation, without subjecting people or the wheelchairs' themselves to stress.

6 Conclusions and Future Work

The final product of this work was a simulation engine and a 2D/3D viewer that is able to test and validate control algorithms for intelligent wheelchairs. What was achieved during experimental testing demonstrated not only that the developed application is capable of simulating maps and wheelchairs but also showed its flexible characteristics. In fact, this simulator is capable of modeling a very wide variety of robots, due to implemented functionalities for configuring the size, center of movement, top speed and acceleration curve of each robotic agent that connects to it.

In this sense, the developed application's capabilities stretch beyond its original intent as a test board for intelligent wheelchairs. Car and pedestrian simulation can be performed, including their physical interaction (collisions). Direct communication between agents is available and, with further developments in agent conception, it could be used for emotional interaction (movement conditioned by attraction or repulsion). Having a distributed architecture, Intellwheels Simulator expects the agents to be external applications that connect through UDP. Because of this attribute, it is able to involve in a single simulation a vast number of intelligent agents, adding the possibility of testing algorithm results in a dynamic, complex environment.

This topic links with the idea of future work, which could be done on development of new robotic agents. The simulator software itself is ready and capable of receiving any kind of control for the virtual motors, therefore the challenge is now to create the various controlling agents themselves.

Concerning the simulator, an interesting addition would be to include other sensors. A good example are encoders, widely used in robot speed and position calculations.

As for the visualization application, the main modification would be to add a "drag and drop" capability. Introducing, in real time, objects and obstacles onto the simulation environment, while seeing where they will be placed is intuitive and allows a more interactive simulation.

Final words to an overview of the contribution of this work on how artificial intelligence and robotic systems can truly aid life of people. Although still a prototype, the Intellwheels project can effectively reduce the limitations and one's dependence on others when faced with physical disabilities.

Acknowledgements

This work was partially supported by FCT Project PTDC/EIA/70695/2006 "ACORD: Adaptative Coordination of Robotic Teams" and LIACC – Artificial Intelligence and Computer Science Lab. of the University of Porto, Portugal. The first author would like to thank for CAPES-Brazil for his PhD scholarship funding.

References

1. Braga, R.A.M., Petry, M., Moreira, A.P., Reis, L.P.: INTELLWHEELS – A Development Platform for Intelligent Wheelchairs for Disabled People. In: 5th International Conference on Informatics in Control, Automation and Robotics, Funchal, Madeira, Portugal, vol. I, pp. 115–121 (2008)
2. Braga, R.A.M., Petry, M.R., Moreira, A.P., Reis, L.P.: Concept and Design of the Intellwheels Development Platform for Intelligent Wheelchairs. In: Cetto, J.A., et al. (eds.) Informatics in Control, Automation and Robotics. LNEE, vol. 37, pp. 191–203. Springer, Heidelberg (2009)
3. Faria, P.M., Braga, R.A.M., Valgôde, E., Reis, L.P.: Interface Framework to Drive an Intelligent Wheelchair using Facial Expressions. In: IEEE International Symposium on Industrial Electronics (ISIE 2007), pp. 1791–1796 (2007)
4. Martins, B., Valgôde, E., Faria, P., Reis, L.P.: Multimedia Interface with an Intelligent Wheelchair. In: Proceedings of CompImage 2006 – Computational Modelling of Objects Represented in Images: Fundamentals Methods and Applications, Coimbra, Portugal, pp. 267–274 (2006)
5. Banks, J.: Introduction to Simulation. In: Proceedings of the 2000 Winter Simulation Conference, Phoenix, Arizona, United States, vol. I, pp. 7–13 (2000)
6. Laue, T., Röfer, T.: SimRobot - Development and Applications. In: Amor, H.B., Boedecker, J., Obst, O. (eds.) The Universe of RoboCup Simulators - Implementations, Challenges and Strategies for Collaboration. Workshop Proceedings of the International Conference on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR 2008). LNCS (LNAI). Springer, Heidelberg (2008)
7. Bourhis, G., Agostini, Y.: The Vahm Robotized Wheelchair: System Architecture and Human-Machine Interaction. Journal of Intelligent and Robotic Systems 22(1), 39–50 (1998)
8. Lau, N., Pereira, A., Melo, A., Neves, A., Figueiredo, J.: Ciber-Rato: Um Ambiente de Simulação de Robots Móveis e Autónomos, Revista do DETUA (2002)
9. Lau, N., Pereira, A., Melo, A., Neves, A., Figueiredo, J.: Ciber-Rato: Uma Competição Robótica num Ambiente Virtual. Revista do DETUA 3(7), 647–650 (2002)
10. Lau, N., Pereira, A., Melo, A., Neves, A., Figueiredo, J.: O Visualizador do Ambiente de Simulação Ciber-Rato. Revista do DETUA 3(7), 651–654 (2002)
11. Milgram, P., Kishino, F.: A Taxonomy of Mixed Reality Visual Displays. IEICE Transactions on Information Systems E77-D, 1321–1329 (1994)
12. Rohrer, M.R.: Seeing is Believing: The Importance Of Visualization in Manufacturing Simulation. In: Winter Simulation Conference, pp. 1211–1216 (2000)
13. Woo, M., Neider, J., Davis, T.: OpenGL Programming Guide, The Official Guide to Learning OpenGL, Version 1.2., 3rd edn. Addison-Wesley, Reading (1999)