ABCM Symposium Series in Mechatronics - Vol. 2 - pp.563-570

HYBRID DELIBERATIVE/REACTIVE ARCHITECTURE FOR HUMAN-ROBOT INTERACTION

Valdir Grassi Junior
Escola Politécnica, University of São Paulo, Brazil
vgrassi@usp.br

Sarangi P. Parikh
Systems Engineering, United States Naval Academy, USA
sparikh@usna.edu

Jun Okamoto Junior
Escola Politécnica, University of São Paulo, Brazil
jokamoto@usp.br

Abstract. *In this paper we present a hybrid architecture that combines deliberative motion plan, local reactive behaviors and human-initiated inputs. The deliberative motion plan is described by an approximated navigational function created over a map of the environment. This function has a unique global minimum at a desired goal destination, and by using this function the robot can navigate autonomously to the goal. While navigating the robot senses the world and detects unpredicted obstacles along the way, and by using local reactive behaviors the robot avoids obstacles without disregarding the motion plan. At any point, the user can give inputs that are incorporated into the system. We combine these three distinct and at times conflicting control inputs in a proper way. We implemented this architecture on a robotic wheelchair and we show some experimental results. In our experiments the user uses a joystick to provide inputs to the system.*

Keywords: *mobile robot, robot architecture, human-robot interaction, robot navigation*

1. Introduction

Mobile robots can help humans on a variety of tasks, such as, material handling, transport, surveillance, demining, assistance to people with disabilities, housekeeping, etc. In order to perform such tasks the robot should have some degree of autonomy. But at the same time, a completely autonomous system is not necessarily suitable for all application domains. In fact, some applications, such as, robotic wheelchairs and manned robotic vehicles, may require intensive human-robot interaction. Whatever is the case, the robot should have an intelligent control system allowing the robot to operate in dynamic and unpredictable environments. This system is responsible for taking decisions based on previously acquired global knowledge of the environment and/or on-the-spot sensed local information in order to successfully accomplish the task. If the application requires interaction with the user, this intelligent control system should integrate the user's input in a proper way.

Mobile robot architectures provide a guideline to build intelligent control systems for robots. An architecture describes what components or building blocks the system should have, how these components are organized to form the system, and how they interact with each other giving the control system its functionality.

Mobile robot architecture can be classified according to the relationship between sensing, planning and acting components inside the architecture. Based on this, there are three types of architectures: deliberative, reactive and hybrid (deliberative/reactive).

In deliberative architectures, there is a planning step in between sensing and acting. This planning is based on a map of the environment acquired by the sensors, and no action can be took without planning. Reactive architectures are composed by a set of reactive behaviors and there is no planning based on a global map or model of the environment. In these reactive behaviors sensing is directly associated with acting. Hybrid architectures combine elements from the other two architectures, i.e., they combine deliberative motion plans with reactive behaviors.

When we compare deliberative architectures with reactive architectures we observe that deliberative architectures work in a more predictable way, have a high dependency of a precise and complete model of the world, and they can generate optimized trajectories for the robot. On the other end, reactive architectures have a faster response to dynamic changes in the environment, can work without a model of the world, and are computationally much simpler. Finally, hybrid architectures try to present the best characteristics of the other two architectures.

In most assistive devices, such as robotic wheelchairs, the operator and the robot must share the control of the system. On this type of application, the robotic system is able to increase precision of the movements, assure the user's safety, and help to decrease the user's fatigue. But also the user should be able to interact with the robotic system at any time to express his or her will.

Wheelchair researchers have taken different approaches to incorporate human inputs into the control loop. One strategy

is to allow the user to command directions to the chair directly and use the autonomous system for ensuring safety by avoiding obstacles (Miller and Slack, 1995). Another is to have the wheelchair perform specified behaviors, such as following a person or tracking a line (Simpson and Levine, 1999), (Borgolte et al., 1998). At an even higher level, it is beneficial to be able to automatically navigate to locations on a map (Bourhis and Agostini, 1998).At this level, landmarks or known targets are used to navigate to the desired location (Gomi and Griffith, 1998), (Crisman and Cleary, 1998).

In previous work we presented a control framework developed to combine the user's input with a deliberative motion plan and reactive obstacle avoidance behavior (Parikh et al., 2004). We also presented an usability study conducted with a robotic wheelchair running the developed control framework (Parikh et al., 2005). Here our main contribution is to define a hybrid (deliberative/reactive) mobile robot architecture that allows the user to interact with the system at the different levels mentioned before. This architecture was developed to be applied on a robotic wheelchair but we believe it can also be applied to other mobile robotic applications where human-robot interaction is required. We describe in this paper each one of the components of the developed architecture and how they are integrated in the whole system.

In section 2, we describe our architecture. We discuss the motion planner, reactive behaviors, human inputs and how the inputs provided by these three components are coordinated. In section 3 we present some experimental results. Finally, in section 4 we conclude discussing some aspects of the developed architecture.

2. Hybrid Deliberative/Reactive Architecture

The developed hybrid architecture for human-robot interaction is composed by the following main components: sensing modules, mapping and localization modules, motion planner, reactive behaviors, input coordinator, and actuator modules. The user interact with the system through the available interfaces. The mapping module generates a map of the environment that can be made available to the user. This map is also used by the motion planner to generate a global deliberative motion plan when the user gives a destination goal for the robot. Figure 1 shows a representation of this architecture.

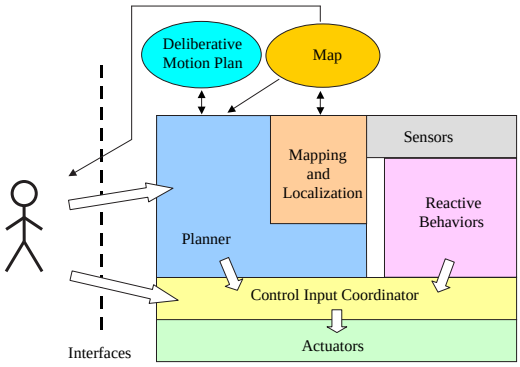

Figure 1. Diagram of the developed architecture.

In this architecture three sources of control input are coordinated and brought together resulting on a single control input that is sent to the actuators. These three control inputs comes from the deliberative motion planner, u_ϕ, the reactive behavior, u_g, and the user's inputs, u_h. Each of these is a 2×1 vector function, $u(t)$, which appears in Eq. 3. These three sources are independent and parallel from each other and they do not necessarily occur together all the time. For example, if the motion planner is not activated, the robot can be driven using only the user's input together with the reactive behaviors. The available control inputs goes to the input coordinator module that applies the heuristic that is presented in this paper to find a suitable control input for the actuators. We will see that at every instant, each vector will define a half-space in R^2, which we will denote by U_ϕ or U_g. We will compute a feasible set, F, as the intersection of the appropriate half spaces. The input coordinator will make use of this feasible set to find the control input to be sent to the robot's actuators.

In the remaining of this section, we present the model used for the robot, each one of the components of the architecture, and more details of how the three inputs are coordinated.

2.1 System Model

Here we model the robot as a two-wheeled, nonholonomic cart-like robot. The governing equations are well-known (Ma, Košecká and Sastry, 1999):

$$\begin{aligned} \dot{x} &= v\cos(\theta) \\ \dot{y} &= v\sin(\theta) \\ \dot{\theta} &= \omega, \end{aligned} \tag{1}$$

where $\nu = [v, \omega]^T$, consists of the forward velocity, v, and angular velocity, ω, while (x, y) are the coordinates of the center of the wheel axle in an inertial frame. θ is the angle that the robot coordinate system forms with the inertial frame. As seen in Fig. 2, generally, we describe features (obstacles, targets, etc.) by (f_i, z_i), where f_i is the angle of the feature relative to the robot and z_i is the corresponding range.

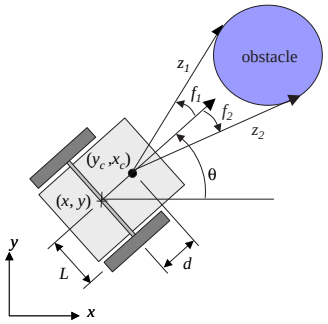

Figure 2. Model of the system. (x, y) is the coordinate of the midpoint of the axle while (x_c, y_c) is the point that is being controlled.

In what follows, we consider a simplified two-dimensional configuration space given by $q = [x_c, y_c]^T$, the coordinates of a reference point on the wheelchair. From Eq. (1), we can write:

$$\dot{q} = \begin{pmatrix} \dot{x_c} \\ \dot{y_c} \end{pmatrix} = \begin{pmatrix} \cos\theta & -d\sin\theta \\ \sin\theta & d\cos\theta \end{pmatrix} \begin{pmatrix} v \\ \omega \end{pmatrix} = J\nu. \tag{2}$$

Because J is always invertible if d is nonzero, we can consider the linear decoupled model:

$$\dot{q} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} u. \tag{3}$$

Any input vector u can be mapped to ν.

We approximate the robot's shape as being a circle of radius ρ centered at the control point (x_c, y_c). When necessary, we grow obstacles in the environment by ρ to plan safe trajectories. Our goal in the next section is to synthesize feedback controllers, $u(q, s(t), h(t))$, where $s(t) = [f_1, z_1, f_2, z_2, \ldots, f_p, z_p]$ is information about features in the environment while $h(t)$ is a 2×1 input vector that is obtained from the operator's joystick. We will assume that the robot has a perfect estimator for the state of the vehicle and therefore of q.

2.2 Motion Planning

One way the user can interact with the system is by selecting on a map a position to where the user wants the robot navigates. By selecting this goal, a deliberative motion plan for the robot is created.

In order to obtain this deliberative motion plan, we would like to use a global navigation function as described by Rimon and Koditschek (1992). However, it is difficult to compute this function instantaneously, so instead, we compute an approximate navigation function by using dynamic programming on an occupancy grid. The occupancy grid is based on known obstacles, such as walls and tables, from a map of the environment. Since this function does not satisfy all the properties of a navigation function, it is called a potential function. However, it does have a unique global minimum at the desired goal. The scalar potential field or navigation function is denoted by $\phi(q)$. Because the construction of this function assumes a nominal map of the environment, we call this resulting controller a deliberative behavior. This contrasts to other works where potential field controllers are called reactive because they do not assume a map of the environment. These reactive potential field controllers are simply based on the sum of repulsive and attractive potential fields and can present many local minimum.

To reach the desired goal, the robot must follow a motion plan that satisfies the constraint $\dot{\phi}(q) < 0$. This constraint is satisfied when the robot moves along the negative gradient of the potential, $-\nabla\phi(q)$. Thus, the preferred controller used in this method is:

$$\dot{q} = u_\phi = -\nabla\phi(q). \tag{4}$$

In fact, $-\nabla\phi(q)$ is not the only control input that moves the robot to the goal. Any control input in the same half space of $-\nabla\phi(q)$ is a feasible controller (Esposito and Kumar, 2002). So, as seen in Fig. 3 (left), U_ϕ is the half space which contains all velocity vectors that the robot can follow that satisfy "the potential field constraint." This is exploited in the following sections to combine reactive behaviors and human inputs with a deliberative motion plan.

2.3 Reactive Behaviors

The set of reactive behaviors that should be implemented in an intelligent control system depends on the robot's application domain. Some examples of reactive behaviors are: follow a person, keep a desired formation with other robots, navigate on the middle of a hallway, navigate through a doorway, dock by a table or desk, avoid obstacles, stop when the robot is too close to an obstacle, etc.

Here we are mainly interested on navigation tasks. For this reason in what follow we show how to combine the deliberative motion plan with local reactive obstacle avoidance behavior. We assume that the robot has a map of the environment that includes static obstacles. However, the robot has no *a priori* knowledge of unmodeled or dynamic obstacles that it may encounter while heading towards the goal.

If obstacles are detected within a specified minimum distance around the robot, the obstacle avoidance algorithm may need to be activated. The detection of objects can be done by any of the available sensors. The distance between the robot and an object i measured by the sensor is z_i and the angle of the object relative to the robot is f_i. If δ is a minimum distance that is required between the robot and an obstacle, and if $|z_i| < \delta$, the robot will switch to an obstacle avoidance behavior. In this case, the robot moves away from the obstacle while driving towards the desired target given by the deliberative plan. The obstacle acts as a constraint to our deliberative plan, which is represented by $g_i = g_i(z_i, f_i, q)$. We define g_i so that, as z_i increases, $\dot{g}_i < 0$. In order not to violate the constraint, we need an input that makes $\dot{g}_i \leq 0$, where

$$\dot{g}_i(q(t)) = \frac{\partial g_i}{\partial q} \cdot \dot{q}. \tag{5}$$

The simple control $u_g = -\nabla g(q)$ guarantees that $\dot{g}_i < 0$. Accordingly we define a half space, U_g, that contains a set of robot configurations which will not collide into the obstacle. The feasible set is the intersection of the half plane given by the potential field, U_ϕ, and the half plane given by the obstacle constraint, U_g. As seen in Fig. 3, $F = U_\phi \cap U_g$ and the goal is to select an input that makes $\dot{\phi} \leq 0$ (decrease the distance to the goal) as well as $\dot{g}_i \leq 0$ (increase the distance to the obstacle). Any u which exists in the set F is a feasible controller. So whenever u_ϕ is not in the feasible space F, we project u_ϕ onto the boundary of the feasible space F.

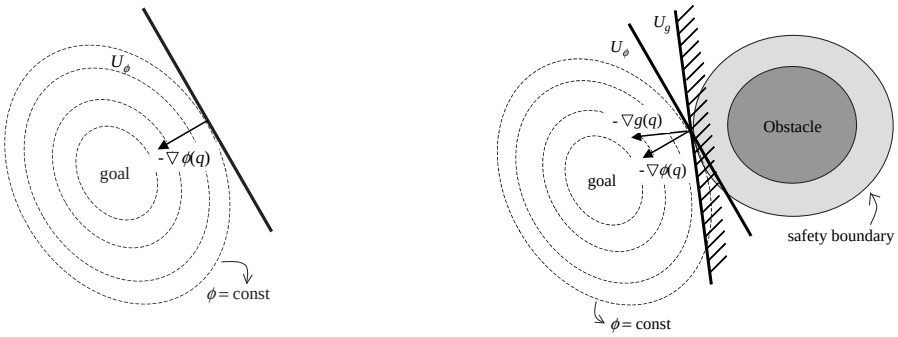

Figure 3. U_ϕ is the half plane consistent with the deliberative motion plan (left). On the right, the feasible region is the intersection of U_ϕ and the half-space given by the obstacle-free configuration space.

2.4 Human Inputs

As mentioned before, the user can interact with the system at a high level by specifying a goal destination on a map. Independently if a goal destination is set or not, at any moment the user can also give a low level input to the system. These low level inputs can be manual velocity commands, and the user could use a joystick like interface to input these type of commands.

We define a hierarchical, prioritized framework to combine human inputs with deliberative plans and reactive behaviors. In this framework, safety is obviously the highest priority. So we would like the human user to maintain control of the system, while keeping our first priority of safety. Only in safe situations when the human user is not providing any input information can autonomous behaviors be invoked.

When the human user manually inputs a command, we have various methods for handling the input. In the simplest case, when there is not a desired goal select, we can allow the user to have complete control of the wheelchair. In this situation, u_h is the human input vector and the controller is simply defined as $u = u_h$.

However, since safety is one of our top priorities, it is important to combine user-initiated inputs with local obstacle avoidance. If the user is manually driving the wheelchair and comes across an obstacle, there are two methods that will allow us to avoid a collision: either drive around it or stop. Before either of these are done, the first step is to check if the user's input, u_h, will result in a collision. As seen in Fig. 4 (left), if the human input is in the feasible half plane, U_g, then the controller input is simply, $u = u_h$. Since the user is not trying to enter a constrained region, we allow him

or her complete control similar to the case without any obstacles. However, if the user's input is located in the obstacle constrained region, then we need to either stop moving or allow the user partial control in a manner that is consistent with the constraint. In Fig. 4 on the right, the user's input is located in the constrained region. In the constrained region, there is a threshold that denotes the stopping region. If the user input is in the constrained region, but outside of the stopping region, then the user is allowed partial control of the motion. Partial control is given by projecting the human input onto the boundary of the obstacle constraint. This allows the chair to nominally move in the direction specified by the user while also avoiding the obstacle. Thus, Eq. (6) permits the user to keep limited control of the wheelchair without a collision.

$$u = (u_h \cdot \hat{t}_g)\hat{t}_g = ||u_h|| \cos(\beta)\hat{t}_g \tag{6}$$

where

$$\hat{t}_g = \frac{(\nabla g_i)^\perp}{||(\nabla g_i)^\perp||}. \tag{7}$$

However, if the input u_h is within the stopping region, the human input cannot be modified to make it consistent with the constraint and is therefore considered unsafe, resulting in the chair being stopped.

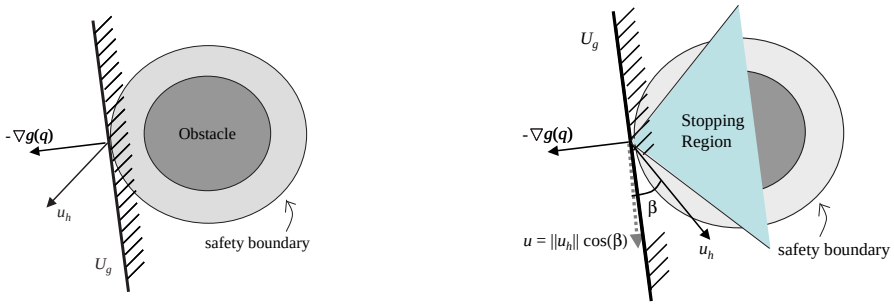

Figure 4. Human input is consistent with half plane given by the obstacle-free configuration space (left). On the right, although the user's input is within the constrained region, by projecting the input into the feasible region, we allow the user to maintain limited control of the wheelchair without a collision.

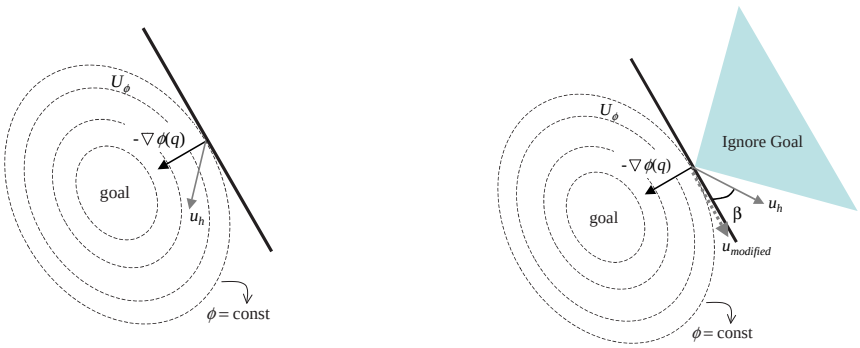

Figure 5. The feasible region is the half plane containing $-\nabla\phi(q)$ (left). On the right, the human input occurs outside of the feasible region and is modified to conform with the goal constraint.

Along with combining human inputs with local behaviors, we are also interested in combining user inputs with a deliberative plan. In this case, the user has selected a desired target that he or she wants to reach. While using a deliberative plan to reach the goal, the user may decide to deviate from the path to perform a subtask. If the user's input is consistent with the goal, i.e. $u_h \in U_\phi$, then we allow the user to maintain complete control of the wheelchair, $u = u_h$, as seen in Fig. 5. If the user's input is not consistent with the goal, then, similar to the obstacle constraint, there are two options available. The first is to modify the user's input to conform with the goal. Figure 5 (right) illustrates the case where the human input is within the goal constrained region and is projected to the boundary of the feasible half plane. This allows the user to have limited control of the wheelchair while conforming with the potential field controller. The controller:

$$u = (u_h \cdot \hat{t}_\phi)\hat{t}_\phi \tag{8}$$

where

$$\hat{t}_\phi = \frac{(\nabla\phi)^\perp}{||(\nabla\phi)^\perp||}, \tag{9}$$

modifies the human input, u_h, by projecting it to the tangent of the potential field line. This keeps the robot motion consistent with $\dot{\phi} \leq 0$, which means the goal has not been abandoned. If the user is persistent, i.e. the input is beyond a threshold, then the goal is disregarded and the human is given complete manual control of the chair. The threshold corresponds to a cone in the space of velocities as seen in Fig. 5 (right). Note that the plane delineating the half-space U_ϕ is not a hard constraint as is the case with an obstacle. Thus, the user has greater flexibility in this case.

2.5 Input Coordinator

The input coordinator module in the architecture is responsible to apply the rules described so far in order to combine all the three control inputs (deliberative motion plan, reactive behaviors, and human inputs).

In summary, as long as the user's input is contained in the feasible space we allow the human to intervene and take control of the motion. In other words, $u = u_h$ if $u_h \in U_\phi \cap U_g$. However when the user's input is not consistent with either the goal-oriented deliberative motion plan or the reactive obstacle avoidance, we apply the rules previously described in this section. If a goal exists, we compare the user's input with the goal-oriented motion plan. Then we either modify the user's input or we drop the goal constraint preserving the user's input. After that, the resultant input is compared with the obstacle avoidance constraint. If necessary, the input is again modified to preserve the safety of the user. This allows the user maximum control while preventing collisions.

Figure 6 shows the algorithm (pseudo-code) used to combine the three control inputs.

$\overline{u} := 0$
if goal is specified **then**
  $u_\phi := -\nabla\phi(q)$
  if user input, u_h **then**
    $\varphi :=$ angle between vector u_h and vector u_ϕ
    if $\varphi > \pi/2$ and $\varphi < \varphi_{limit}$ **then**
      $\overline{u} :=$ projection of u_h into the half-space defined by u_ϕ
    else
      $\overline{u} := u_h$
    end if
  else
    $\overline{u} := u_\phi$
  end if
else if user input **then**
  $\overline{u} := u_h$
end if
if obstacle close and $\overline{u} \neq 0$ **then**
  $u_g := -\nabla g(q)$
  $\gamma :=$ angle between vector $\overline{u}$ and vector u_g
  if $\gamma > \pi/2$ and $\gamma < \gamma_{limit}$ **then**
    $\overline{u} :=$ projection of $\overline{u}$ into the half-space defined by u_g
  else if $\gamma > \gamma_{limit}$ **then**
    $\overline{u} := 0$
  end if
end if
Use $\overline{u}$ as control input for the wheelchair

Figure 6. Algorithm for combining deliberative plans, reactive behaviors and human inputs.

3. Results

We implemented the described control system on a robotic wheelchair equipped with onboard processing and a suite of sensors. The chair has a omni-directional camera, mounted over the user's head, that allows the user to view 360 degrees around the wheelchair. Along with the vision system, there is a laser scanner, which is mounted in the front of the wheelchair, under the user's feet. The laser measures distances at every half degree through a 180 degree scan. Similarly, IR Proximity sensors are placed on the back of the chair to detect any obstacles located behind the wheelchair. Lastly, encoders on the motors provide a dead reckoning system for the wheelchair. This is augmented, when necessary, by vision-based localization with landmarks on the ceiling. The wheelchair platform is discussed in greater detail by Rao et al. (2002).

We run several experiments with the implemented control system. For all the experiments we used odometry to represent ground truth because simple tests conducted revealed that for the distances traveled during the experiments the errors in odometry were not significative (Patel et al., 2002).

Figure 7 shows an overhead view of one of the rooms where the experiments were conducted. Overlapped to the map

is an example of a navigation function that is generated automatically to the specified goal (Latombe, 1991), (Rimon and Koditschek, 1992). If the wheelchair encounters obstacles on its way to the target, it switches to an obstacle avoidance behavior to circumvent the obstacle before switching back to the deliberative plan (Esposito and Kumar, 2002).

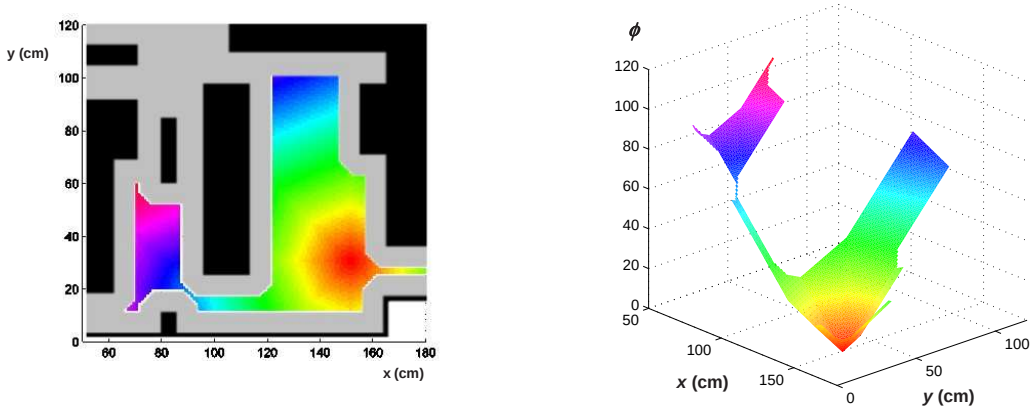

Figure 7. Navigation function for the user specified goal.

Figure 8 (left) shows the path followed by the wheelchair when using the autonomous mode. As seen in the figure, the chair is guided by the potential field lines to the selected location. The chair drives completely autonomously using the deliberative motion plan in tandem with the sensors on the system. The dashed trajectory is the path the chair would have taken if the unmodeled obstacle were not present in the environment.

Figure 8 (right) shows a trajectory took by the chair in the semi-autonomous mode, i.e., when the user starts giving control inputs to the chair while the chair is moving autonomously. In the figure, we point out the different controllers that are used. It can be seen that the semi-autonomous mode involves a smooth integration of the deliberative plan with reactive behaviors as well as human inputs.

Figure 8. Sample trajectory taken by the wheelchair while in the autonomous mode (left). The solid black line shows when the deliberative controller is invoked. The lighter (green) line segments represent where the activated obstacle avoidance behavior is composed with the deliberative behavior. The dashed (blue) line is the trajectory that the chair takes if there are no obstacles. On the right, a sample wheelchair trajectory using a deliberative plan, combined with user input (semi-autonomous mode). Part(a) represents the deliberative path taken, part(b) represents the user's input, which is consistent with the deliberative plan, part(c) represents when the obstacle avoidance behavior is activated.

4. Conclusion

The main focus of this paper is to present an hybrid (deliberative/reactive) architecture developed to allow a human user to share the control of a robot with the autonomous control system.

In the developed architecture we use an approximated navigation function to describe the deliberative motion plan. This function defines at every instant a set of control inputs that can be used to take the robot to the specified goal. Whenever is possible, the system uses the negative gradient of the navigation function as a preferred control input. The reactive behaviors act as a constraint on the set of possible control inputs. This limites the choice for an appropriate control input that conciliates the behaviors with the deliberative motion plan. Besides, human inputs are combined with the reactive behavior and the motion plan. Our hierarchical set of rules gives priority to the user while maintaining safety. Whenever is appropriate the user's input is modified so that it can still complies with the goal of the motion planning.

This architecture was implemented on a robotic wheelchair and some experimental results were presented here. The results showed the user can successfully interact with the robot at any time.

In order to simplify the implementation we use odometry as our ground truth localization and we provide a map of

the environment to the system. In spite of this, the architecture has space for the implementation of more elaborated localization and mapping components. In the architecture presented here the map should be represented in a geometric form such as an occupancy grid. This is important because a navigation function must be defined over such a geometric map. Localization also should be developed according to the resolution of this map.

The architecture presented here was developed mainly to solve navigation tasks while complying with human inputs. For this reason only an obstacle avoidance reactive behavior was tested in the implementation. However, other types of reactive behaviors would also be compatible with this architecture. These behaviors should be described as an inequality constraint that limits the set of control inputs that solves the deliberative motion plan. Esposito and Kumar (2002), for example, defines a formation control behavior as such constraint.

For future work we intend to adapt this architecture to be more suitable for general task. We also intend to explore other kinds of reactive behaviors useful in the context of a robotic wheelchair application and combine these behaviors with the deliberative motion planning and the user's initiated inputs. One example of such behaviors could be getting close to a table and parking there.

5. Acknowledgements

We gratefully acknowledge support from NSF grant IIS-0083240 and DUE-9979635, CAPES grant BEX0112/03-8, FAPESP grant 02/00225-8, and CNPq grant 472727/03-6. We also thank Dr. Vijay Kumar for insightful discussion and the GRASP Laboratory.

6. References

Borgolte, U., Hyer, H., Buhler, C., Heck, H. and Hoelper, R., 1998, "Architectural concepts of a semi-autonomous wheelchair", Journal of Intelligent and Robotic Systems, Vol. 22, pp. 233-253.

Bourhis, G. and Agostini, Y., 1998, "Man-machine cooperation for the control of an intelligent powered wheelchair", Journal of Intelligent and Robotic Systems, Vol. 22, pp. 269-287.

Crisman, J.D. and Cleary, M.E., 1998, "Progress on the deictic controlled wheelchair", Assistive Technology and Artificial Intelligence, Editors: Mittal, V., Yanca, H., Aronis, J. and Simpson, R., Ed. New York: Springer, pp. 137-149.

Esposito, J.M., Kumar, V., 2002, "A method for modifying closed loop motion plans to satisfy unpredictable dynamics constraints at run time", Proceedings of the International Conference on Robotics and Automation, pp. 1691-1696.

Gomi, T. and Griffith, A., 1998, "Developoing intelligent wheelchairs for the handicapped wheelchair", Assistive Technology and Artificial Intelligence, New York:Springer, pp.151-178.

Latombe, J.-C., 1991, "Robot Motion Planning", Boston:Kluwer.

Ma, Y., Košecká, J. and Sastry, S., 1999, "Vision guided navigation for a nonholonomic mobile robot", Transactions on Robotics and Automation, Vol. 15, No. 3, pp. 521-536.

Miller, D.P. and Slack, M.G., 1995, "Design and Testing of a low-cost robotic wheelchair prototype", Autonomous Robots, Vol. 2, pp. 77-88.

Rao, R., Conn, K., Jung, S., Katupitiya, J., Kientz, T., Kumar, V., Ostrowski, J., Patel, S. and Taylor, C., 2002, "Human robot interaction: Applications to smart wheelchairs", Proceedings of the International Conference on Robotics and Automation, Washington, DC.

Rimon, E. and Koditschek, D.E., 1992, "Exact robot navigation using artificial potencial functions", IEEE Trans. on Robotics and Automation, Vol. 8, No. 5, pp. 501-518.

Simpson, R.C. and Levine, S.P., 1999, "Automatic adaptation in the navchair assistive wheelchair navigation system", IEEE Transactions on Rehabilitation Engineering, Vol. 7, No. 4, pp. 452-463.

Parikh, S., Grassi Jr, V., Kumar, V. and Okamoto Jr, J., 2004, "Incorporating user inputs in motion planning for a smart wheelchair", Proceedings of the International Conference on Robotics and Automation, New Orleans, LA, pp. 2043-2048.

Parikh, S., Grassi Jr, V., Kumar, V. and Okamoto Jr, J., 2005, "Usability study of a control framework for an intelligent wheelchair", Proceedings of the International Conference on Robotics and Automation, Barcelona, Spain.

Patel, S., Jung, S.H., Ostrowski, J.P., Rao, R.S., Taylor, C.J., 2002, "Sensor Based Doorway Navigation for a Nonholonomic Vehicle", Proceedings of the International Conference on Robotics and Automation, Washington, DC.

7. Responsibility notice

The author(s) is (are) the only responsible for the printed material included in this paper