Shared Navigational Control and User Intent Detection in an Intelligent Walker

Cunjun Huang*, Glenn Wasson†, Majd Alwan‡, Pradip Sheth*, Alexandre Ledoux*
University of Virginia
* Department of Mechanical and Aerospace Engineering
† Computer Science Department
‡ Medical Automation Research Center
{ ch8me | wasson | ma5x | pns7q | ledoux } @virginia.edu

Abstract

This paper describes the navigational control scheme used in the CO-Operative Locomotion Aide (*COOL Aide)*, an intelligent walker designed to assist the elderly or the disabled with normal, and routine walking tasks. Navigation is achieved through a shared control architecture that recognizes the goals of both the human user and the walker. The control system is based on a synthesis of heuristic logic that exploits a dynamic model of walker system that can detect sliding and loss of walker stability. The model is used to predict the user's intended path, based on the history of information collected from the walker's sensors. Sensor information consists primarily of the forces and moments the user exerts on the walker's handles during the natural assisted walking process, as well as the user's local environment. Based on the model's prediction, the walker's state, and the walker's environment, the control system can confirm or overturn the hypotheses of user's intent it put forward and can influence the walker's heading if the system believes the user will not reach the perceived intended goal unassisted. This paper discusses the model's use in the shared control scheme and the mechanism for detecting/handling errors in the model's predictions.

Keywords-Assistive and Healthcare Robotics, Personal Robots, Human Computer Interaction, Real-Time Systems, Shared Control, User Intent Detection

1 Introduction

One of the most important factors in quality of life for the older adults is their ability to move about independently. Not only is mobility crucial for performing the activities of daily living (ADLs), but for maintaining fitness and vitality. Lack of independence and exercise can lead to a vicious cycle. Decreased mobility due to a perceived lack of safety can cause muscular atrophy and a loss of the feeling of empowerment (both of which contribute to further decreased mobility).

An intelligent pedestrian mobility aide, termed *COOL Aide*, is being developed to help the frail elders negotiate obstacles in indoor environment. The primary goal of this work is to augment a user's ability to walk, not replace it. In this sense, we are seeking to help those who can and want to walk perform this task more safely and easily. As the world's elderly population rises (the US Senior population will double over the next 30 years (Kramarow *et al*., 1999) and the cost of healthcare skyrockets (to $4 trillion over the same period (Ciole *et al*., 1999)), robotic mobility aides will increase in importance.

Two key issues in developing *COOL Aide* are how to determine a user's intent and how to activate a smooth control without destabilizing the user. This paper describes a mathematical model that can analyze walker system dynamics with human input on-line. A goal of this project is to provide a very intuitive and natural interface and thus no provision is made for any explicit means by which the human "enters their goal". Instead, the model is used to predict the user's short term (a few seconds) goal/path based on the forces and moments they apply to the walker's handles in the natural process of assisted walking, in addition to the user's local environment and the walker's current state (such as velocity). In this way, operation of *COOL Aide* is identical to operation of any other wheeled walker, the user simply pushes it. The output of the model can then be used to influence the steering angle of the walker frame. However, the control system only attempts to adjust the walker's heading if the user is believed to be in a difficult or dangerous situation. The model and the control system are built with the understanding that predictions can be imprecise or incorrect as a user may perform similar actions to achieve different goals. The control system therefore uses the model to make multiple predictions over time and switches between different control logics/ actions when suitable differences between predicted and received handle inputs occur.

There are a number of applications of robotic technology for assistive devices for walking to help the elderly or the blind. Among them, MIT's "PAMM" (Dubowsky *et al*., 2000) and CMU robotic walker (Morris *et al*., 2003) can detect the user's intent. However, these systems do not infer the implicit user's intent from rich sensory information.

Moreover, there are several applications of user intent detection and shard control from wheelchair community. Such as "NavChair" (Levine *et al*., 1999) and "Sharioto" (Vanhooydonck *et al*., 2003). "Sharioto", for example, attempted to estimate the user's intent from user's noisy input signal, through a joystick, and the interaction with the perceived environment to generate navigational behaviors. The shared control of the wheelchair is different from that of a walker in that the user of the wheelchair does not provide the propulsion force; he/she only needs to show his/her command explicitly by operating the joystick. In our shared control architecture, the user is responsible to the propulsion of the walker. The user's intent is not entered explicitly; instead, the users' implicit intent is inferred from the forces and moments applied to the walker's handles as they push the walker naturally, and the controller will detect and behave accordingly.

The *COOL Aide* is built on a commercial three-wheeled Invacare walker frame. Two six-axis force and moment sensors from ATI Industrial Automation (US120-160) are mounted on the handles. The absolute encoder mounted to measure the heading angle of the frontal wheel and the two incremental encoders mounted at shaft of back wheels are used to estimate the walker's velocity, position, and heading. An infrared obstacle detection sensor (PBS-03JN from Hokuyo Automatic Co., Ltd.), provides a 180° radial depth map of the environment surrounding *COOL Aide*. Control of the walker's front wheel is achieved by a stepper motor driven belt and pulley system. A clutch is used to disconnect the motor from the wheel when control is not required. *COOL Aide* uses a laptop to run all needed software. A picture of the system is shown at figure 1.

Figure 1. *COOL Aide* hardware configuration

2 Walker System Dynamics Modeling

The physics based mathematical model (Huang and Sheth, 2004) to representing the walker's system dynamics with human inputs is a key element for the shared navigational control. This dynamic model is comprised of a set of coupled nonlinear differential equations derived from the Lagrangian and the Lagrange multipliers of the system. The general form for the model can be represented as, with reference to Figure 2:

$$\begin{cases} M(\vec{q})\ddot{\vec{q}} = F(\vec{q},\dot{\vec{q}}) + \left(\dfrac{\partial g(\vec{q},\dot{\vec{q}})}{\partial \dot{\vec{q}}}\right)^T \lambda \\ g(\vec{q},\dot{\vec{q}}) = 0; \text{ This is a 4 degree-of-freedom model with} \\ \vec{q} \equiv \begin{Bmatrix} x_W \\ y_W \\ \theta_W \\ \theta_C \end{Bmatrix} \end{cases} \quad (1)$$

The system mass matrix is $\underset{(4\times4)}{M}$. The force vector F models the handle forces and moments while the Lagrange Multipliers are λ for the two nonholonomic (rolling without sliding) constraints represented by $g(\vec{q},\dot{\vec{q}}) = 0$. One of the two constraints is for the front wheel and the second for the consolidated axle of the two rear wheels. By monitoring the forces in these nonholonomic constraints, the model is able to detect the sliding of the walker. The sliding condition is considered to be jeopardizing the user's balance and hence if this case is detected, it indicates the need to change the current control. The model can also detect user instability when vertical reaction forces on the wheels become zero or negative.

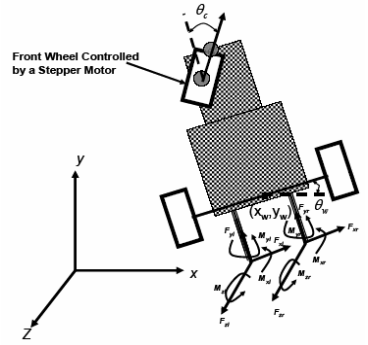

Figure 2. Two dimensional motion dynamics model of the walker with three dimensional human forces/moments input

Preliminary results of the path prediction using this model were validated against VICON motion capture data and were presented in (Wasson et al., 2004).

3 Shared Navigational Control

Navigation is controlled by both the user and the *COOL Aide's* navigational control system. *COOL Aide* activates control of the steering when a collision is likely to occur or when the controller believes that some pro-active movement of the walker frame is necessary to help the user reach his/her short term goal. Based on the requirement of shared control through user intent detection, three general design goals of *COOL Aide's* controller are:

1. maintain the user's safety and stability.

2. assist users in accomplishing their navigational goals.
3. do not increase the user's cognitive load

Achieving the first goal requires the ability to detect the user's potential instability and an understanding of how the control actions may jeopardize stability. The second goal requires the ability to understand the user's navigational intent and how it relates to the local environment. The final goal requires the first two goals to be performed via an intuitive interface to the *COOL Aide*. The remainder of this section addresses aspects of the system related to these goals.

3.1 Environment Modeling for Shared Control

COOL Aide's environment is sensed by an infrared sensor that produces a radial depth map. The inherent uncertainty created by the sensor's noise must be accounted for when building a map. Also, obstacles in the environment must be modeled such that they can be used within the physics-based model of Section 2 to achieve appropriate shared control.

A 2D grid style map is adopted due to the two dimensional nature of the infrared sensor data. Histogramic in Motion Mapping (HIMM) (Borenstein and Koren, 1991) was adopted for map-building in this research due to its computational efficiency (Murphy 2000).

121 data points, each 1.8° apart, are acquired in each scan of the infrared sensor. *COOL Aide* processes the inner 180° arc at a range between 200mm and 3000mm. Thus the sensor can detect an object wider than 0.1m. Based on these data, the system builds a 40 x 80 grid map with the grid size 100mm.

The system combines the previous map with the current instantaneous map to get the most up-to-date grid local map using the following formula:

$$m[i, j] = m[i, j]_{prev} \times e^{-\lambda \Delta t} + m[i, j]_{curr} \qquad (2)$$

where λ is the "forgetting" factor, Δt is the time difference between the two consecutive scan, these two together determine the forgetting speed of the algorithm for the old data in memory.

By tuning the parameters used in HIMM method and the forgetting factor λ , a local map suited for local navigational control can be achieved.

Once a map of the environment is built, the forces and moments exerted on the walker's handles can be "interpreted" within the context of the environment. Once the user's likely goal is determined, a decision about whether or not to activate control and what control mode, to apply, if any, can be made.

For *COOL Aide*, we chose a method inspired by the Virtual Force Field (VFF) method (Borenstein and Koren, 1989), which can be viewed as an extension of the Potential Field/Function method (Khatib, 1985). *COOL Aide* is different from the autonomous robots used in the navigation literature because it is mostly a passive robot. This means that the user provides the propulsion and *COOL Aide* controls only the angle of the front steering wheel. So in contrast to the VFF method where virtual forces of the occupied cells are applied directly to the center of the robotic system which in turn determines the movement of the robotic system, in *COOL Aide* the occupied cells only have influence on the rotational joint connecting the front wheel to the walker frame. Occupied cells in *COOL Aide*'s map have virtual repulsive moments on that joint, which influences the steering of the front wheel guiding the walker together with the control from the user.

The repulsive virtual moment should satisfy the following requirements:

1. The virtual moment should only be effective within a certain distance from the obstacle, r_0 for example. Continuous and smooth condition should apply on the boundary where the change of virtual moment from ineffective to effective happens.
2. The repulsive moment should have an upper limit to accommodate the ability of the human user to adjust to control decisions made by the walker. Omni-directional mobile robots can respond appropriately to large forces, but *COOL Aide* must consider the stability of its user and cannot turn too sharply. The traditional potential function (Khatib, 1985; Borenstein and Koren, 1989) which has infinite upper boundary also makes certain tasks such as docking onto an object or passing through a tight space difficult. Using a bounded, repulsive virtual moment, and a set of heuristic rules (see below), *COOL Aide* can respond correctly in these circumstances.
3. Because the infrared sensor is only capable of detecting obstacles in the half plane in front of the walker, the virtual moment is only computed in the half plane in front of the sensor.
4. An obstacle on the walker's path should have a larger influence than the obstacle on side of the walker's path.
5. The influence of the relative velocity between the obstacle and walker is also considered in the calculation of the virtual moment. In this research, due to the limit of the sensor and slow motion character of the elderly user, the mapped objects were assumed to be static or quasi-static. Therefore, the relative velocity of the walker to the obstacle (defined as v_{obst}) is the projection of the velocity of walker in local coordinates to the line between the walker and obstacle. Then the influence of the relative velocity can be expressed in a coefficient as indicated in Equation (3):

$$K_{velo} = \begin{cases} 1 + \frac{1}{2}\{\tanh[10(v_{obst} - 1)] + 1\} & v_{obst} > \varepsilon \\ 0 & v_{obst} \le \varepsilon \end{cases} \qquad (3)$$

Where ε is a small positive number, or zero, based on different control status.

Based on the requirement above, the repulsive virtual moment of the i_{th} occupied cell on the walker is expressed as:

$$M_{obst}(i)=\begin{cases}-sign(\vec{r},\vec{v})KK_{velo}\left[1+\cos\left(\frac{|\vec{r}|}{r_0}\pi\right)\right]\cos^2(\angle(\vec{r},\vec{v})) & |\vec{r}|\le r_0\\ 0 & |\vec{r}|>r_0\end{cases} \quad (4)$$

Where $sign(\vec{r},\vec{v})=\begin{cases}1 & \angle(\vec{r},\vec{v})\in[0,\frac{\pi}{2}]\\ -1 & \angle(\vec{r},\vec{v})\in[-\frac{\pi}{2},0)\end{cases}$

$\angle(\vec{r},\vec{v})\equiv\angle\vec{r}-\angle\vec{v}$

It should be noted that the repulsive virtual moment of the left occupied cells that lie on the left of the central vertical axis of the map may counteract the repulsive virtual moment of occupied cells that are to the right part of the same axis. This can make an obstacle that is most dangerous to the user have a small repulsive virtual moment. This can mislead the system into not taking the proper action in time. To avoid this, the system calculates the repulsive virtual moment generated from the map of the environment in the following way. First, system detects obstacle that lies on the central vertical axis of the map. Then the absolute value of the virtual moments contributed by all occupied cells by this obstacle (the sum of absolute value of Equation (4)) is computed. This value is assigned a sign according to whether the center of obstacle is in left/right half of the map. Then Equation (4) is used to calculate the repulsive virtual moment of obstacles that don't lie across the central vertical axis.

3.2 Shared Control Strategy for Navigational Control

In *COOL Aide*, the user has the control authority for the navigation of the walker, while the control from *COOL Aide* will activate only when necessary, although the controller monitors *COOL Aide* all the time. To achieve this, *COOL Aide* uses the math model described in section 2 and heuristic rules in the control.

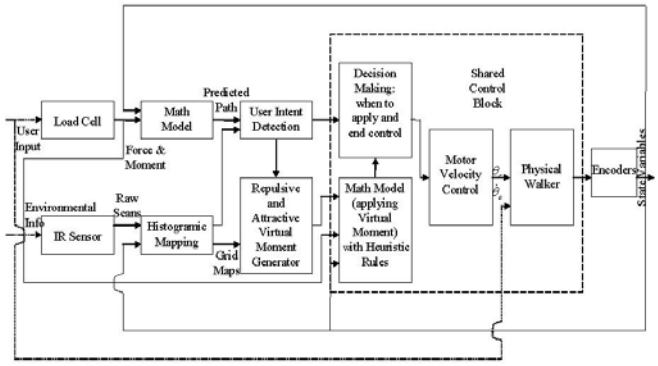

Figure 3 Shared control system architecture

With reference to the shared control block diagram shown in figure 3, the user's propulsive forces/moments and the environment are considered inputs into the system's dynamics. User's forces/moments input contribute to all the general forces in the model, while the influence of environment only affects the general force to the front wheel (F_{θ_c}).

When obstacles are on or near the path of the walker, there can be two possibilities: one is that user isn't aware of the obstacles appearing in his/her way, and this requires the system to take action to help the user avoid the obstacles. The second is that the user is aware of the obstacles appearing and he/she wants to go there (docking, for example). As control logic, the controller of *COOL Aide* considers the obstacle avoidance to be the first choice and the docking as the second choice.

To avoid activating the obstacle avoidance too early which can be a cause of oscillations in the navigational control, the controller of *COOL Aide* will only activate the obstacle avoidance after the absolute value of the repulsive virtual moment exceeds a certain threshold. When obstacle avoidance activates, part of the walker's control is transferred from user to the *COOL Aide*. The walker is controlled by both the controller (controls steering) and the user (controls propulsion).

Under this obstacle avoidance control strategy, system first evaluates the predicted response of the walker dynamics with the consideration of both the user's input and environmental influence, together with walker's nonholonomic constraints (i.e. the actual path), then uses the filtered predicted result of the rotational velocity of the front wheel ($\dot{\theta}_c(t)$) to estimate the needed command parameter for the motor velocity control in the next time step, and then sends these commands to be executed by the lower-level controller. To make the steering smooth and comfortable to the user under the requirement of achieving local navigation, *COOL Aide* uses velocity control mode for the stepper motor as the low level controller.

The obstacle avoidance action is terminated when the absolute value of the repulsive virtual moment drops below the threshold, or if during this period, the control system detects that the user either stopped walking or was fighting the steering guidance of the walker. "Control fighting" implies a conflict between the user's need and controller's intent, jeopardizes user stability, and may occur during docking. The controller monitors this conflict by using the resultant fighting moment as the "conflict index". When this index exceeds a certain threshold, the controller can infer, with confidence, the user's disagreement with the control action. There may be several choices to resolve this conflict. Currently the controller takes the simplest choice by stopping the motor, disengaging the clutch, and returning the full control back to the user.

3.3 Detection of User's Navigational Intent

Besides the obstacle avoidance, *COOL Aide* also uses models to detect the user's intent over a short time horizon. The controller continuously analyzes the predicted paths, within the context of environment map acquired, and

continuously detects and updates the user's navigational intent. The predicted path is the output of the deterministic physics-based model for walker dynamics. It changes fast, reflecting the changes in the user's force-moment input. The force-moment data reflect the user's intent through actuation affected by the human's muskelo-skeletal system, with potentially some perceptual or motor "noise", thus the analysis result of instantaneous predicted path and current map may not reflect the true user's navigational intent. Moreover, the instantaneous predicted path represents a snapshot of the user's navigational intent. A better representation of the user's true navigational intent could be drawn on a series of these instantaneous observations, i.e. based on history.

To this end, Dempster-Shafer theory (DST) is used to extract the user's navigational intent from the historical observations, or evidences. Dempster-Shafer theory belongs to possibility theory. In Dempster-Shafer theory, belief functions are mappings from the power set of possible evidence causes to the closed interval between 0 and 1 of real numbers (Hoffman). This set is called Frame of Discernment (FOD) Θ. The elements inside, which are called focal element, are assigned belief mass by a belief function. *Any two belief functions over the same FOD with at least one focal element in common may be combined into a new belief function over FOD using Dempster's rule of combination* (Murphy 1998). This rule strongly emphasizes the agreement between multiple sources and ignores all the conflicting evidence through a normalization factor. The advantages of the Dempster-Shafer theory over the Bayesian framework include its ability to represent ignorance, and the fact that it does not require a priori probabilities. Here ignorance includes incompleteness, imprecision, and uncertainty. It allows ignorance when assigning the possibility values (belief mass). Moreover, this method can combine evidence explicitly. Because of these advantages, Dempster-Shafer theory has had a large number of applications in engineering in the past years. It has the versatility to represent and combine different types of evidence obtained from multiple sources as well. Based on these merits, the Dempster-Shafer theory was applied to the problem of detecting the user's navigational intent.

The user's navigational intent detection algorithm entails 3 iterative: a) Analyze current map and predicted path to assign belief mass to the valid passages on the map, b) Track the valid passages in subsequent maps, and c) apply Dempster-Shafer's evidence combining rule to obtain the possibilities for the valid passages of being the user's navigational goal. The details of these procedures are illustrated below.

3.3.1 Analyzing the relationship between current map and predicted path. The algorithm starts by searching for valid passages (a valid passage is an opening on the map whose width is larger than the dimension of the walker. This concept takes into consideration the geometric dimensions of the walker) and assigns each a corresponding FOD. The advantage of this procedure over lumping all evidence causes of all valid passages into one FOD is that FOD is relatively simple and uniform, there are always only 4 elements in the set (as will be illustrated below), and thus the belief function of interest is equal to the belief mass after assignment/ applying the combination rule. This choice simplifies implementation in the software and makes it computationally more efficient. The disadvantage of this choice lies in that the usual decision making criteria, such as max belief criteria or max plausibility criteria, can not be simply applied. With our method, when considering all the valid passages, these multi FODs adopted here make the possibility set non-exclusive between different FODs. Thus there may exist such a case where multiple FODs from different passages form the user's intent. For example, the user may want to pass passage A as well as passage B, as far they are sequential from the user's point of view, and thus passing both A and B will become the user's intent.

Therefore for each valid passage on the environment map at a particular time, there exists a power set: $\Theta = \{\phi, A, \overline{A}, \theta\}$, among them: ϕ represents null subset, it is always zero in implementation; A represents the possibility of user will pass this valid passage; $\overline{A}$ represents the possibility that user will not pass this valid passage; θ represents the possibility of $A \cup \overline{A}$, which reflects ignorance. Belief mass is assigned to each focal element; the relationship among them is obvious, $m(A) + m(\overline{A}) + m(\theta) = 1$.

Then the rules for assigning the belief mass are defined as:

Rule 1: If the predicted path crosses the valid passage, then assign belief mass so that $m(A) + m(\theta) = 1$. Assign a constant value to *m(A)*. Here, 0.75 is assigned.

Rule 2: If the predicted path does not cross the valid passage, then assign the belief mass so that $m(\overline{A}) + m(\theta) = 1$. Meanwhile, assign $m(\overline{A})$ of each passage based on the proportion of the length between the end point of predicted path and the center of the valid passage.

The continuity between whether or not predicted path crosses a valid passage can be satisfied with the consideration of plausibility. When predicted path approaches the valid passage but does not cross it, $m(\overline{A})$ approaches zero according to rule 2, value of ignorance $m(\theta)$ approaches 1. At the condition that the end point of predicted path crosses the valid passage, rule 1 applies. Then:

$$P(A_1) = m(A_1) + m(\theta_1) = m(\theta_2) = m(A_2) + m(\theta_2) = P(A_2)$$

These rules guarantee the continuity of the plausibility of passing a valid passage.

This procedure can be practically implemented by:

1. Finding all the occupied blocks in the current local map and extract their boundaries.
2. Find all the valid passages on the map.
3. Assign belief mass to FOD of each valid passage.

A dynamic list is used to record the valid passages that have ever appeared on the map and contribute to the detection of the user's navigational intent. The size of the list is neither pre-fixed nor limited, with elements continuously being dropped from the list and added into the list.

3.3.2 Tracking the valid passages. To dynamically track the valid passages in the updated maps, a matching problem solving approach was adopted: match the elements (passages in this case) in the list with the counterparts from the current map. However, our matching problem is not a classical one due to the open nature of our set: first, the number of elements in the two matching sets is not the same; second, for each match, elements may have been removed, or newly added to the set to be matched. To overcome this peculiarity, the open set matching problem is then separated into two sub problems. One is a target tracking problem: track the elements in the list with the current measurements. Here, measurement means certain environmental geometric properties acquired from the environmental mapping. But it is not simple a target tracking problem, as the system has to cater for the facts that old objects may go out of sensor's detection region, and that new objects may come into the detection region. Therefore, after the tracking finished, the other sub problem is to deal with what the target tracking process did not resolve, namely to process the elements that are in the list but did not appear in the current map and the new passages that appeared in the current map.

A more challenging problem for matching is caused by the characteristics of the environmental sensor. The sensor emits beams to detect the environment. When position of the sensor changes, the shape of the objects on the map based on the sensor data will change, thus causing the change of the geometric properties of the corresponding passages, sometimes dramatically. However, in this research we have not addressed this problem due to the fact that of the walker moves in a slow motion. Hence, the probability of the dramatic change in shape of the objects on the map is small, and would only happen if the orientation of the walker was changes dramatically abruptly.

Because of the relative consistency of the user's intent compared to force/moment change, it is assumed that the elements in the list that represent the user's current navigational intent will be the elements appearing in the current map. However, in the first sub problem, the list will not include new passages appearing in the current map, because the preference of user to a certain place in the map, as an evidence, should appear several times on the map to increase the confidence in being an intended destination. Since the candidate indicating the user's current navigational intent corresponds to the current measurement, the elements that don't have a matched measurement in the current map are marked as inactive elements and will be eliminated in the iteration that follow.

Then the first sub problem: tracking the elements in the list with the current measurement becomes tracking the elements that appear as measurement in the latest previous map. To track the active elements in the list with the current measurement, first, the geometric characters (such as passages) in the latest previous environmental map is projected into the current local coordinate system via coordinate transformation, this is the predicted measurement. Then, the geometric characters are compared between the current measurements and the predicted measurement to obtain the correct pair of the elements in the list and the current measurement. Currently, the nearest neighbor (NN) method and method of pattern match in a region are used together for the tracking.

For the second sub problem, as described above, the inactive elements that do not have matched counterpart measurement in the current map will be eliminated from the list when it is identified to have been passed away from user's vision. This will prevent the list from getting huge and keep the processing speed fast; the new valid passages that do not have the matched counterpart of the elements in the list will be considered as new elements to be added into the list.

3.3.3 Combining the evidences to obtain the user's navigational intent. The elements in the list are then combined with their counterpart in current measurement list to the updated list. To combine evidence of user's intent by Dempster-Shafer theory, the specific combining rules applied are:

$$m_{i\oplus j}(A)=\frac{m(A_i)m(A_j)+m(A_i)m(\theta_j)+m(\theta_i)m(A_j)}{1-\left[m(A_i)m(\overline{A}_j)+m(\overline{A}_i)m(A_j)\right]} \quad (5)$$

$$m_{i\oplus j}(\overline{A})=\frac{m(\overline{A}_i)m(\overline{A}_j)+m(\overline{A}_i)m(\theta_j)+m(\theta_i)m(\overline{A}_j)}{1-\left[m(A_i)m(\overline{A}_j)+m(\overline{A}_i)m(A_j)\right]} \quad (6)$$

$$m_{i\oplus j}(\theta)=\frac{m(\theta_i)m(\theta_j)}{1-\left[m(A_i)m(\overline{A}_j)+m(\overline{A}_i)m(A_j)\right]} \quad (7)$$

It is worthy to note that with the method illustrated here, FOD will keep the highest possibility value in history. This is not applicable to user's intent detection, because the user's current intent should always be the focus. So in order to overcome this problem, a forgetting factor is applied to FODs in the list. The possibility value whether user will pass through a passage $m(A)$ or not $m(\overline{A})$ is modified to decay with time in the following way: $m(a_i)=m(a_i)e^{-\lambda\Delta t}$, where λ is the forgetting factor, Δt is the time between iterative user's intent detection. This change will lead FOD to converge to total ignorance eventually, if no new supporting evidence is added. With

this modification of the belief functions for each FOD, (5), (6) and (7) will be used to update FODs for user's intent.

Following these procedure, the controller will get a list containing hypotheses. Each element in the list is actually a hypothesis of where the user may want to go, that is, his/her potential navigational intent. These hypotheses are confirmed or refuted by evidence tracking and combining. By comparing their possibility values, these hypotheses compete with each other. The user's navigational intent lays in the hypotheses whose accumulated possibility values will be among those with high values. Once these values exceed a certain threshold (90%), the controller will be confident about this intent.

If the controller believes it "knows" the user's intended destination, it will act proactively to guide the user towards it.

4 Experiment Results

Figure 4 shows *COOL Aide* performing obstacle avoidance. The user pushes the walker along a hallway toward an obstacle. The user's intent, i.e. the control action that would be achieved if the forces/moments they are exerting on the walker frame were followed, would result in a collision with the obstacle (the dashed line). The *COOL Aide* controller activated to guide the user away from the obstacle. From the natural feedback of the motion change felt through the handles, the user realized the obstacle and followed *COOL Aide*'s control action. The *COOL Aide* controller ceased active control when the user has left the danger zone, leaving the user in control again.

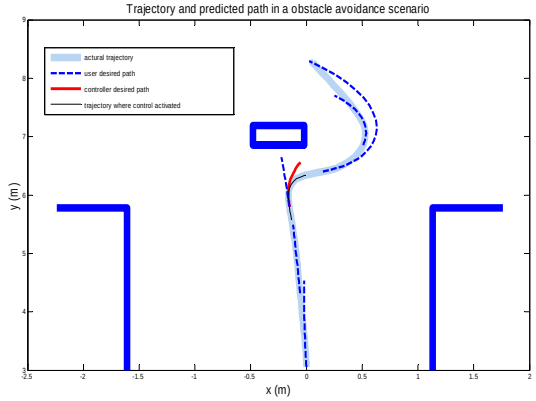

Figure 4. Obstacle avoidance scenario: user accepted the guidance from the controller

In figure 5, the user pushes the walker toward an obstacle which the controller detects, attempting to guide the user away to the empty space to the left. However, the user resists the guiding of the controller as detected by increased resultant turning moment at the handles. This can be seen from the difference between the user desired path (dashed line) and the controller desired path (solid line) in the figure. The controller detected the increase of the conflict between the user intent and the controller's intent and relents, giving control back to the user. In general, *COOL Aide* can use this "fighting" detection to switch between multiple hypotheses about its user's intent – causing a corresponding switch in the control strategy.

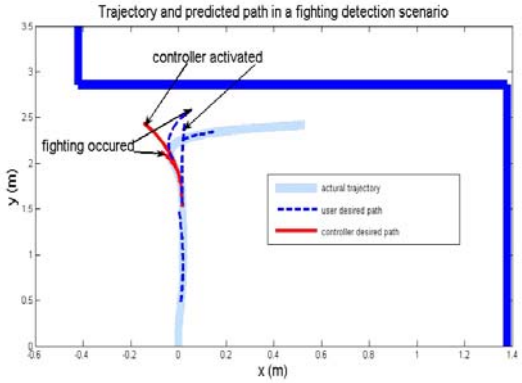

Figure 5. "Fighting" scenario: user disagreed with the guidance of the controller

The speed of approach to an obstacle is an important factor for the control, in addition to the relative distance. The system considers a slow approach to an object not be a potential harm to the user, and thus the controller will not activate when this case happens (see figure 6). This could happen when the elderly user was aware of the object and wanted to dock onto it.

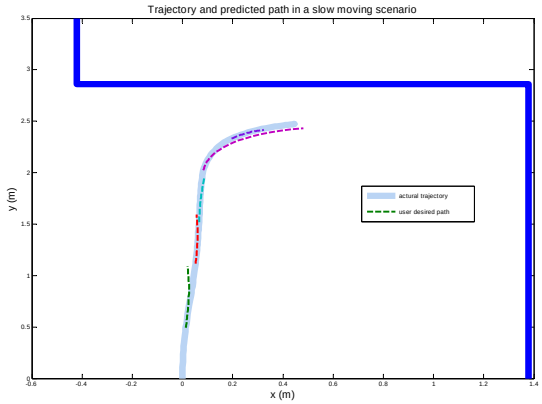

Figure 6. No control action will happen when the user approaches an obstacle slowly.

In figure 7, the user walks along a hallway and wants to turn and go through the doorway on the left. The controller detects that the user's desired path is toward a tight corridor and so switches to an appropriate control strategy. In this case, obstacle avoidance is inappropriate. Instead, *COOL Aide* turns the wheel to a slightly different angle than that indicated by the user's input and holds the front wheel there until the door has been passed. This is an example of how *COOL Aide* can be used to pro-actively perform control actions to help in navigating in tight spaces in cluttered environments and beyond simple obstacle avoidance. It is hoped that *COOL Aide* will be able to take important control actions that are only slightly

differing from the user's desired actions (i.e. the user knows approximately what to do) and so the user will not necessarily be aware that they did not achieve the desired result alone.

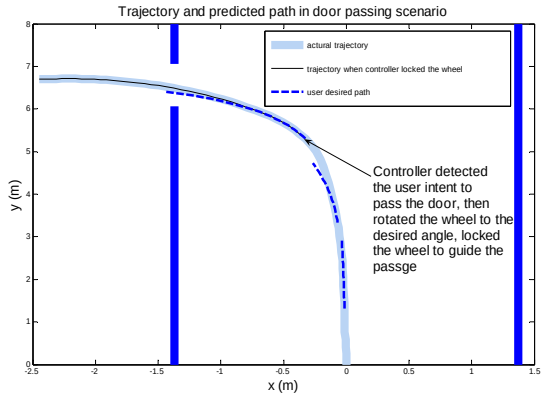

Figure 7. Door passing scenario: controller detects the user's goal and provides corresponding assistance

5 Conclusions and Future Work

Shared control and user's intent detection were implemented on the *COOL Aide* platform, an intelligent pedestrian mobility aid for the elderly. This paper used walker system dynamics as the foundation for determining parameters needed to generate control commands. The commands, for shared local navigational control, took into consideration both the user's input and the influence from the surrounding environment. Using heuristic rules, the controller detected the conflict between user and machine and solved this conflict accordingly. User's intent detection from the natural walking process is a key part for shared control scheme. It is the foundation from which the system makes proactive control to help user. The test results show that the controller of *COOL Aide* successfully detected the user's navigational intent and took the suitable control action to help user achieve his/her goal or avoid danger. Also when the conflict of user's intent and controller's intent occurred, the controller detected this conflict and took appropriate action, namely yielding the user's will. Future work will entail evaluating the navigation system with elder walker users.

Acknowledgments

This research was funded by the National Science Foundation (NSF award ID 0004247). The authors would like to thank Jawad Altaf Chaudhry and Mathew B Wagner for their input.

References

Kramarow, E., Lentzner, H., Rooks R., Weeks, J. and Saydah S. Health and Aging Chartbook, Health United States, National Center for Health Statistics. http://www.cdc.gov/nchs/data/hus99cht.pdf

Ciole R. and Trusko, B. HealthCare 2020: Challenges for the Millennium, Health Management Technology, August 1999: 34-38.

Huang, Cunjun, and Sheth, P. N., 2004. Derivation of the 2-D Model of Walker Dynamics with 3-D Forces and Moments Measured at the Handles. Available at: http://www.cs.virginia.edu/~gsw2c/walker.htm

Glenn Wasson, Pradip Sheth, Cunjun Huang, Alexandre Ledoux, Majd Alwan, A Physics-Based Model for Predicting User Intent in Shared-Control Pedestrian Mobility Aids, *Proc. 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep.28-Oct.2, Sendai, Japan

Johann Borenstein, Yoram Koren, Histogramic In-Motion Mapping for Mobile Robot Obstacle Avoidance, *IEEE Trans. Robotics & Automation*, Vol.7, No.4, Aug. 1991, p535-539

Robin R. Murphy, *An Introduction to AI Robotics*, the MIT Press, 2000

Khatib, O., Real-Time Obstacle Avoidance for Manipulators and Mobile Robots, *IEEE Int. Conf. on Robotics and Automation*, March 1985, pp. 500-505.

Borenstein, J., and Koren, Y., Real-time Obstacle Avoidance for Fast Mobile Robots, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 19, No. 5, Sept./Oct. 1989, pp. 1179-1187.

Dubowsky, S., Genot, F., Godding, S., Kozono, H., Skwersky, A., Yu, H., and Yu, L.S. "PAMM - A Robotic Aid to the Elderly for Mobility Assistance and Monitoring: A "Helping-Hand" for the Elderly." *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 00)*, San Francisco, CA, Vol. 1, pp. 570-576, April 2000.

A.C. Morris, R.R. Donamukkala, A. Kapuria, A.M. Steinfeld, J. Matthews, J. Dunbar-Jacobs, and S. Thrun, A Robotic Walker that provides guidance, *Proceedings of the 2003 IEEE Conference on Robotics and Automation (ICRA '03)*, May, 2003

Simon P. Levine, David A. Bell, Lincoln A. Jaros, Richard C. Simpson, Yoram Koren, Johann Borenstein, "The NavChair Assistive WheelChair Navigation System", *IEEE Transaction on Rehabilitation Engineering*, Vol.7, No.4, pp. 443-451, Dec. 1999

D. Vanhooydonck, E. Demeester, M. Nuttin and H. Van Brussel, "Shared Control for Intelligent Wheelchairs: an Implicit Estimation of the User Intention", *Proc. of ASER 2003*, pp. 176-182, March 13-15, 2003, Bardolino, Italy.

Shafer, G., A Mathematical Theory of Evidence, Princeton University Press, 1976.

James C. Hoffman, A Generalization of Dempster's Rule for Combining Belief: A Tutorial.

Robin R. Murphy, Dempster-Shafer Theory for Sensor Fusion in Autonomous Mobile Robots, *IEEE Transaction on Robots and Automation*, Vol. 14, No.2, April 1998.