Modular and Adaptive Wheelchair Automation

Brenna D. Argall

Northwestern University, Evanston IL, 60208 USA
Rehabilitation Institute of Chicago, Chicago IL, 60211 USA
brenna.argall@northwestern.edu

Abstract. We present in this paper a novel framework for the design of a modular and adaptive partial-automation wheelchair. Our design in particular aims to address hurdles to the adoption of partial-automation wheelchairs within general society. In this experimental work, a single assistance module (assisted doorway traversal) is evaluated, with arbitration between multiple goals (from multiple detected doors) and multiple control signals (from an autonomous path planner, and the human user). The experimental work provides the foundation and proof-of-concept for the technical components of our proposed modular and adaptive wheelchair robot. The system is evaluated within multiple environmental scenarios and shows good performance.

1 Introduction and Related Work

We envision a future where the partial-automation of powered wheelchairs will be the standard: that when a person is being fit for a wheelchair by a therapist, a variety of autonomy options will be available, just like today a variety of seating options are available.

While many individuals achieve sufficient mobility using manual and powered wheelchairs, a survey of 65 clinicians within the United States found that between 10% and 40% could not be prescribed either [1], leaving those individuals reliant on a caretaker for mobility. The potential for "smart" wheelchairs—which incorporate robotics technologies—to aid the mobility of those with motor or cognitive impairments has been recognized for decades [2]. A survey of epidemiological data estimates that between 1.4 and 2.1 million individuals would benefit from a smart wheelchair at least some of the time [3]. Robotics autonomy can help with obstacle avoidance, navigation, route planning and spatially-constrained maneuvers. However, despite decades of development, and significant advances in capabilities [4, 5], control [6–8] and interfaces [9, 10], very few smart wheelchair technologies have made the transition to the public and commercial sectors.[1]

One dominating confound to practical adoption is cost: At least in the short term and within the United States, these technologies will not be covered by Medicare/Medicaid or insurance plans, and so any system that is going to be of

[1] Exceptions include the Smart Wheelchair from Smile Rehab [11] and TAO-7 Intelligent Wheelchair Base from AAI Canada [12].

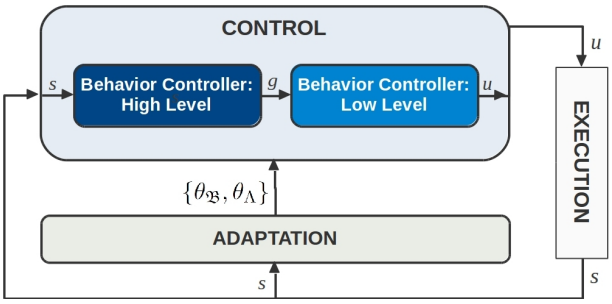

Fig. 1. Schematic of our full control architecture with adaptation. Behavior parameters $\theta_{\mathfrak{B}}$ and control sharing strategy parameters θ_{Λ} are adapted in response to cues from the user and metrics computed from data observed by the robot's sensors about world state s. The high-level behavior controller generates goals g, and the low-level behavior controller generates control commands $\mathbf{u}$.

practical benefit to general society must be reasonable to finance out of pocket. The general trend for the majority of work in smart wheelchairs has been to offer a complete system: that is very capable, but also involves a fair amount of infrastructure, and components that are costly. Many are developed in their entirety from the ground up, including the wheelchair hardware and software systems [13–18]. While this is the most common approach for smart wheelchair development, some projects do take a modular approach: to software, for example to accommodate multiple control interfaces [14, 15] or sensor modules [19]; or to hardware, to be able to interface with existing powered wheelchairs [15–17].

An important observation is that users of assistive devices overwhelmingly prefer to retain as much control as possible, and cede only a minimum amount of control authority to the machine [20, 21]. Thus, many smart wheelchairs offer a variety, often a hierarchy, of autonomous and semi-autonomous control modes within their shared control schemes [14, 22, 23]. Others explicitly target low-profile automation [10, 24], create new customized levels of autonomy [25], or blend the user's control commands with the automation's control commands [21, 26, 27]. Most commonly, shared-control smart wheelchair platforms place the high-level control (e.g. goal selection, route planning) with the user, and the low-level control (e.g. motion control commands, obstacle avoidance) with the machine [6, 14, 28–30].

In this paper, we introduce a system that prioritizes *customization, modularity* and the use of *commercial hardware*, to facilitate practical adoption by users. The result of this project will be a complete system consisting of modular software and hardware components, easily added to a commercial wheelchair platform, and able to be customized to and by the user. We present in this paper our control framework, and first expermimental results.

2 Technical Approach

We introduce a system of modular software and hardware components—which scale with a user's physical needs, financial means and personal preferences. The framework introduced in this section will be grounded in Section 3 with concrete implementations of goal arbitration and control sharing, including example data.

2.1 High and Low Level Behaviors

Our control framework (Figure 1) assumes the existence of a set $\mathfrak{B}$ of autonomous robot behaviors, and a set Λ of control sharing strategies. The user is able to select a custom set $\mathfrak{B}_u \subseteq \mathfrak{B}$ of behaviors. The set $\mathfrak{B}$ furthermore is partitioned into high-level behaviors $\mathfrak{B}_h$ and low-level behaviors $\mathfrak{B}_\ell$. In our work with wheelchair navigation, behaviors in $\mathfrak{B}_h$ typically are associated with planning (e.g., path generation), while behaviors in $\mathfrak{B}_\ell$ are associated with motion generation (e.g., path driving).

A high-level behavior $b_h \in \mathfrak{B}_h$ outputs a goal g given state input $\mathbf{x}$,

$$g \leftarrow b_h(\mathbf{x}) \tag{1}$$

Each autonomy goal g is evaluated for confidence c_g that it is the user's goal. A goal g is passed to the low-level control module only if its confidence is both over threshold, $\tau_g < c_g$, and significantly higher than the second-highest confident goal, $\delta\tau_g < c_g - c_{g_\circ}$. (Grounded in Section 3.1.)

A low-level behavior $b_\ell \in \mathfrak{B}_\ell$ outputs a control command $\mathbf{u}$ given state input $\mathbf{x}$ and goal g,

$$\mathbf{u} \leftarrow b_\ell(\mathbf{x}, g) \tag{2}$$

(Figure 2). Within the low-level control module, commands generated by the autonomy are then reasoned about within the control sharing logic. (Grounded in Section 3.2.)

The robot autonomy and control sharing components together enable a flexible and modular architecture, where any combination of autonomy behaviors in $\mathfrak{B}$ can be selected by the user for inclusion in $\mathfrak{B}_u$. The autonomy resolves conflicts between competing behaviors in $\mathfrak{B}_u$—via a resource controller that registers the data and control signal needs of each behavior $b \in \mathfrak{B}_u$, as well as what data that behavior provides. The robot autonomy and control sharing also reason about human input—which might play a role in behavior selection, be a cue for an already-running behavior or be blended with autonomy-generated commands for reasons of safety.

2.2 Control Sharing

A control sharing strategy $\lambda \in \Lambda$ can take on one of three formulations: 1. All control to the user; 2. All control to the automation; or 3. A shared control formulation that blends the two control inputs.

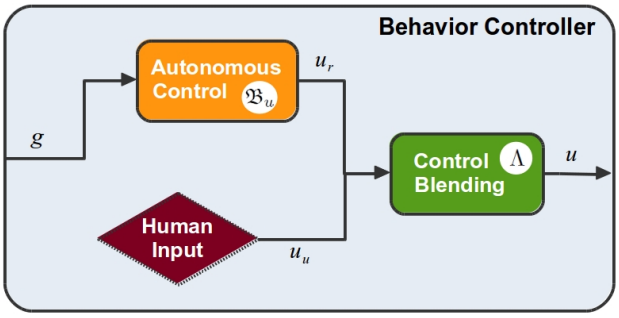

Fig. 2. Schematic of a behavior controller within our system. (Low-level; an equivalent architecture is used for the high-level behavior controller.) The behavior controller considers multiple autonomous behaviors (in $\mathfrak{B}_u$) and considers user control signals ($\mathbf{u}_u$) to produce a single blended output signal ($\mathbf{u}$).

Each behavior $b \in \mathfrak{B}$ has an associated control sharing strategy $\lambda_b \in \Lambda$. A strategy λ_b furthermore has an associated set of values $\boldsymbol{\theta}_{\lambda_b}$, that parameterize the unction used to blend the user command $\mathbf{u}_u$ and autonomy command $\mathbf{u}_r$. Thus, each autonomy behavior b has associated with it a unique combination of control sharing strategy λ_b and paramterization $\boldsymbol{\theta}_{\lambda_b}$ for that strategy.

2.3 Adaptation

A defining feature of our architecture is the adaptation of the autonomy behaviors in $\mathfrak{B}_u$ and control strategies Λ_u associated with them.

Each behavior $b \in \mathfrak{B}$ available within our system has an associated set of parameters $\boldsymbol{\theta}_b$ which are available for modulation by the adaptation component of our framework. For example, the path planner [31] used on our development platform (Figure 4) has parameters to modulate how much curvature there is in the generated trajectory, and how aggressively the robot will attempt to reach the goal position.

Similarly, each control sharing strategy $\lambda \in \Lambda$ has an associated set of parameters $\boldsymbol{\theta}_\lambda$ which are available for modulation by the adaptation component of our framework. For example, in a linear control sharing formulation, the parameter which dictates how much control is allocated to the user might be increased as the user becomes a more proficient driver.

Exactly what influence the parameters $\boldsymbol{\theta}_b$ have on associated behavior b (and parameters $\boldsymbol{\theta}_\lambda$ on strategy λ) varies greatly across behaviors (and strategies). However, the approaches used to *modulate* the parameters will be common across behaviors and strategies, and any number of machine learning algorithms may be used to perform this modulation. One key factor to consider will be the *feedback signal* received by a machine learning algorithm—which will be computed autonomously from environmental cues and also gathered from the user, who

not only is a non-expert in the area of robotics but additionally has motor impairments and provides signals through a possibly limited control interface (e.g., used to drive the wheelchair).

2.4 Integration with Commercial Hardware

Lastly, each behavior has an associated specification ϑ of what form of input signals it expects to receive from the human user, which will change depending on the control device employed. For example, a traditional 2-axis joystick provides a 2-D continuous-valued control signal; while a Sip-N-Puff interface typically provides a 1-D *non-proportional* control signal, whose magnitude does not scale with the magnitude of the user's input (i.e., blowing or sucking). For a given human-input specification ϑ, the set $\mathfrak{B}$ is partitioned into a subset of behaviors $\mathfrak{B}_\vartheta$ which satisfy that specification.

Fig. 3. *Left:* Sip-N-Puff control interface [32], where commands are issued by blowing and sucking on a straw. *Right:* Switch-based electronic head array with three proximity sensors [33].

Our system will prioritize good performance with non-proportional control interfaces (e.g., Figure 3), which we believe provide a greater opportunity for autonomy to make an impact—since control with these devices is more difficult. More broadly, the interfaces used for human input will be restricted to those which are commercially available. Not only is this technology extensively validated (having been evaluated by thousands of users), but it also is covered by insurance—and thus, from the standpoint of financial feasibility, more readily accessible to users.

To interface with multiple electronics packages from different wheelchair vendors, our proposed add-on system will be inserted between the input device and the control module of the commercial wheelchair system (Figure 4, right). Several makes and models of wheelchairs are specifically designed to accept signals from expandable input controllers,[2] which present the user's signals to the proprietary control electronics. Importantly, presenting our control signals to an expandable input mechanism *should not void* the wheelchair warranty.

By restricting our system to integrate easily with commercial hardware and maintain a low price point, we are knowingly making trade-offs with respect to how "complete" the system is in the capabilities it offers to the user. The idea is that, in exchange, the system becomes more accessible, and transfer to the general public thus more feasible.

[2] Including: the Q-Logic system from Pride Mobility; PG Drive Systems' R-Net electronics system used by Permobil, Pride Mobility, and Sunrise Medical; and the MK6i electronics system used by Invacare [34, 35].

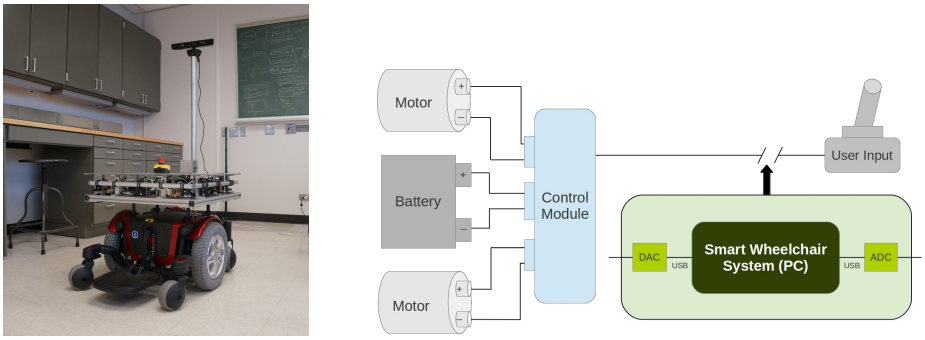

Fig. 4. *Left:* Development platform. A differential drive mobile robot built on a wheelchair base, with a ring of IR and ultrasonic sensors and a top-mounted Kinect. *Right:* Proposed integration with commercial wheelchair platforms. Signals from the user input device (e.g., joystick) are interrupted and processed, along with data from the onboard sensors, within our smart wheelchair PC system. Control signals—generated by the human (i.e., unmodified input signals), by the automation, or a blend of the two—are then sent to the commercial control module of the wheelchair.

2.5 Development Platform

Our development platform (Figure 4, left) consists of a Pride Mobility Quantum 600 base [36], modified to be drive-by-wire (including inverter and wheel encoders) by Sensible Machines [37]. To this we have added the sensing and computing components detailed in the first column of Table 1.[3]

Column 2 details our proposed modular add-on components to a commercial wheelchair. The base system components are detailed ($670), along with configurations that add IR and ultra-sonic sensors ($940) and an IMU ($1140). In comparison with the development platform, the add-on system additionally requires an inverter and input device interface board; in order to interface with the expandable input mechanism of various commercial wheelchairs. This modular system will be interfaced with participants' own wheelchairs in future subject studies.

The base system, consisting of only a Kinect sensor, is evaluated in the following section. All software has been developed within the Robot Operating System (ROS), with each high and low level behavior operating as an individual ROS node. Customization thus consists of bringing up only those nodes identified in $\mathfrak{B}_u$, and having the resource controller reason about and resolve any conflicts between those nodes.

[3] Full specifications: *IR range sensor* = Sharp GP2Y0A02YK IR distance sensors; *Ultra-sonic range sensor* = Maxbotix LV-MaxSonar-EZ1 Ultra-sonic range sensors; *Sensor interface board* = Arduino Mega2560; *mini-PC* = Shuttle XH61 mini-PC with Intel i7-2600S processor, 16GB DDR3 SDRAM, and 40GB solid state hard drive; *Embedded computing* = 2 x Hard Kernel ODroid U2; *Input device interface board* = Arduino Due DEV-11589; *IMU* = Pololu CHR-UM6 9-DoF IMU.

Development Platform (*current*)		Modular Add-on Components	
		1 x Input device interface board	$50
		1 x Inverter	$120
1 x Microsoft Kinect	$150	1 x Microsoft Kinect	$150
1 x mini-PC	$700	1 x Embedded computing	$350
		Total:	$670
10 x IR range sensors	$250	4 x IR range sensors	$100
4 x Ultra-sonic range sensors	$120	4 x Ultra-sonic range sensors	$120
1 x Sensor interface board	$50	1 x Sensor interface board	$50
Total:	$1020	*Total:*	$940
		1 x IMU	$200
		Total:	$1140

Table 1. Hardware specifications of our development platform, and the modular add-on components system to augment a commercial powered wheelchair.

3 Assessment

In this experimental work, a single assistance module (assisted doorway traversal) is evaluated, with arbitration between multiple goals (from multiple detected doors and the inferred user's goal) and multiple control signals (from an autonomous path planner and the human user).[4] Doorway navigation was chosen as a task frequently cited as challenging for powered wheelchair drivers, due to tight spatial constraints [1]. Doorways are identified by our autonomous doorway detection algorithm [38], which provides both the location and orientation of an observed doorway.

Our system has been evaluated under various testing conditions, from which illustrative results are presented here. The system was found to successfully identify multiple high-level goals, autonomously, and then reason between them within the goal arbitration module. Speed commands generated by the autonomous motion planner and the human operator were blended, with the result of successful and safe task execution.

3.1 Goal Inference and Arbitration

Our first assessment concerns the arbitration between multiple goals, including the user's inferred goal—to ground the framework presented in Section 2.1.

We infer the user's goal from only those control signals used to teleoperate the wheelchair (rotational and translational speed commands). While there are undoubtedly many advantages to using custom interfaces like touch screens, that are tailored to the task or a user's particular needs, our intent here is to instead to push the limits of existing commercial control interface technologies by using software solutions whenever possible. Also, while small screens with

[4] Implementation of the adaptation components of our framework are currently under development, and are not included in this assessment.

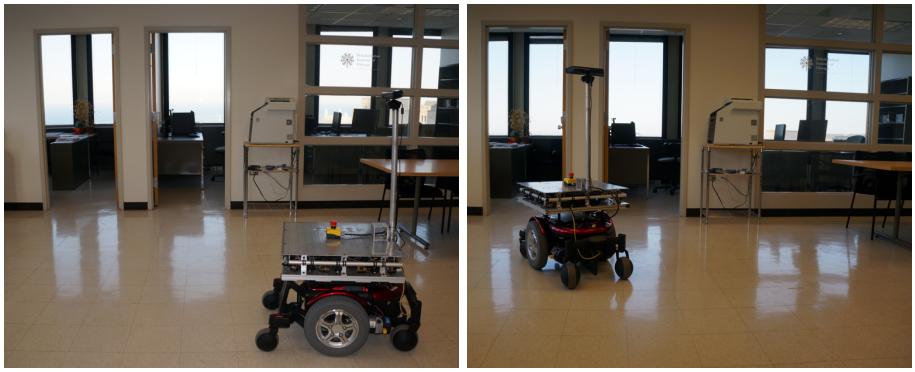

Fig. 5. Example environment scenario, with two side-by-side doors.

menu-based interfaces are available with many control devices (especially ones like in Figure 3), menu navigation with these interfaces can be cumbersome. Our aim is for the user to be able to indicate their goal or preference more intuitively.

To infer the user's immediate goal g_u, our system maintains a smoothed estimate $\tilde{\mathbf{u}}_u$ of the user's command, weighted by the time since the last update:

$$\tilde{\mathbf{u}}_u^t \leftarrow \kappa \cdot \mathbf{u}_u^t + (\kappa - 1) \cdot \tilde{\mathbf{u}}_u^{t-1} \tag{3}$$

$$\kappa = e^{-\Delta\tau}$$

where $\Delta\tau$ is the difference between the timestamps of $\tilde{\mathbf{u}}_u^{t-1}$ and $\mathbf{u}_u^t$. The smooth command is then forward projected to calculate the immediate user goal g_u (in our implementation, the projection time is $3.0s$).

To determine which of the autonomy goals might be the user's final (high-level) goal, a confidence measure is computed for each autonomously detected goal. Associated with each autonomy goal g is a set of N navigation goals $\{g_{nav}^i\}_{i=0}^N$. Navigation goals are executed as a sequence, with one goal $g_{nav}^\star \in \{g_{nav}\}$ active at a given time. For example, during doorway assistance two navigation goals are set ($N = 2$), along the normal of the pose of the identified door on opposite sides of the door frame—achieving the first aligns the robot for doorway traversal and achieving the second has the robot pass through the doorway.

In our implementation, the confidence $c_g \in [0, 1]$ associated with an observed goal g is calculated based on the distance d and heading ϕ (absolute value) to the current active navigation goal $g_{nav}^\star$:

$$c_g = c_p \cdot \left(\beta \cdot \left(\frac{2}{1 + e^\phi}\right) + (1 - \beta) \cdot \left(\frac{2}{1 + e^d}\right)\right) \tag{4}$$

$$\beta = \texttt{min}\,(1, d)$$

where $c_p \in [0, 1]$ is the perception confidence from observing g.[5] The parameter β dictates that when the robot is far from $g^{\star}_{nav}$ ($d > 1m$), aiming towards the navigation goal ($\phi \to 0$) during driving matters most; while closing the distance matters most when near to $g^{\star}_{nav}$.

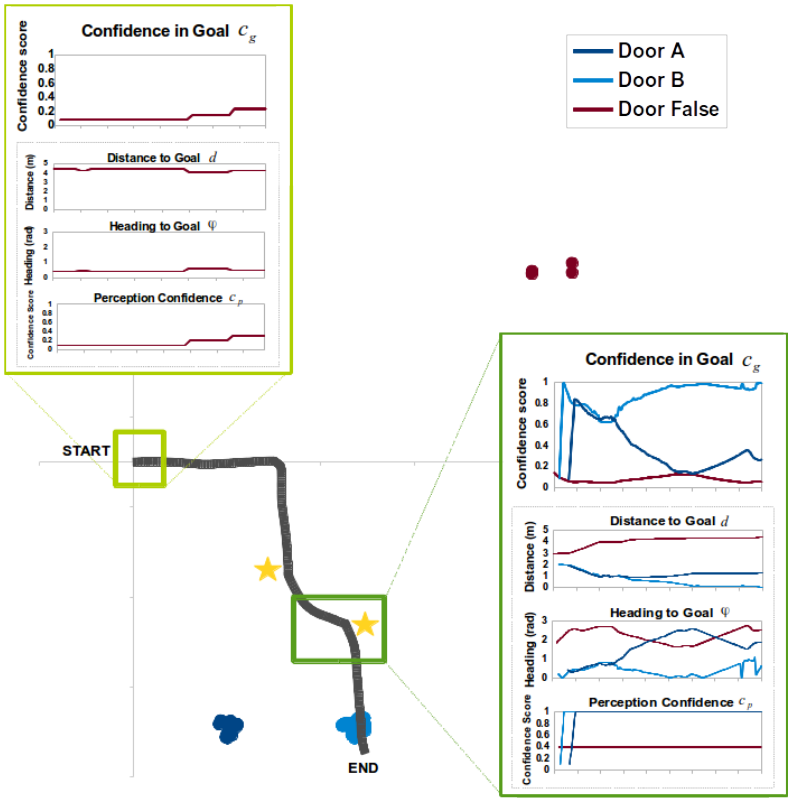

Fig. 6. Confidence associated with autonomously observed goals. Two doors (light blue, dark blue) exist within the environment; a third (dark red) is falsely identified. Plot panels show (top→bottom) goal confidence c_g, distance to navigation goal d, heading to navigation goal ϕ and perception confidence c_p. At the start of the run, only the false positive door is observed, however the low perception confidence keeps the overall confidence c_g also low. As the robot turns towards the doors, both are identified with similarly high confidence. As the user issues commands that show preference for Door B, its confidence rises until it is sufficiently greater than that of Door A for Door B to become the active goal (first star). The user initially retains control however, as dictated by the associated control sharing strategy (under the condition that the user continues to issue commands). When the user ceases issuing commands (second star), the autonomy takes over in full. Robot ground path shown in dark gray.

[5] Since d and ϕ are both always positive, the logistic function (fractions in parentheses) range is $[0, 0.5]$. The factor of 2 in this equation compensates for this fact, so that the range of c_g becomes $[0, 1]$.

If there exists a goal g whose confidence is both above threshold and significantly so ($\delta\tau_g < c_g - c_{g_\circ}$), this goal is considered active and passed to the low-level control module. If no autonomy goal is active, then the user-inferred goal is sent instead.

Figure 6 shows the goal confidence calculated over a sample run in an environment with two side-by-side doors (Figure 5).

3.2 Command Arbitration and Safety Monitoring

Our second assessment concerns the sharing of control between the user and the autonomy—to ground the framework presented in Section 2.2.

Goals passed from the high-level behavior module, post-arbitration, and presented to the low-level control module to generate control commands. In the presented assessment, the low-level controller is a velocity-based path planner [31], which generates rotational and translational speeds for robot. The autonomy command $\mathbf{u}_r$ is blended with the user's command $\mathbf{u}_u$, according to the control sharing strategy λ_b associated with the behavior b that generated the goal g.

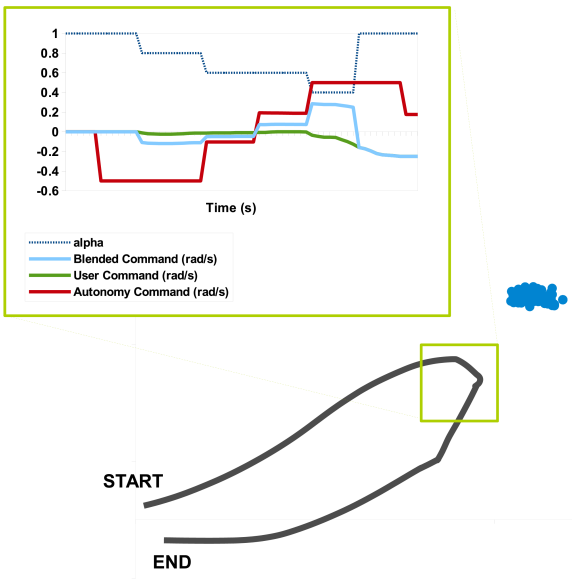

Fig. 7. Nimble transfers of control authority during control blending. The user never relinquishes control (indicated via all non-zero commands) as s/he drives near to an autonomously detected goal (blue dots), which is also near an obstacle. The autonomy therefore never takes control with the aim of achieving the detected goal. However, as the robot path (dark gray line) nears the obstacle, the autonomy gradually takes over some of the control (reduced α, dashed line in plot) to avoid collision. When the user turns away from the obstacle and goal however, the autonomy immediately transfers control back to the user ($\alpha = 1$).

The results presented in Figures 6– 8 utilize a linear control blending formulation:

$$\mathbf{u} \leftarrow \alpha \cdot \mathbf{u}_u + (1 - \alpha) \cdot \mathbf{u}_r \tag{5}$$

$$\alpha \in [0, 1]$$

The automation command $\mathbf{u}_r$ is generated by the path planner, which takes the inferred user goal g_u as its target.

The control sharing strategies implemented within our architecture to date include: *all-user* ($\alpha = 1$); *all-autonomy* ($\alpha = 0$); *blending-zero-relinquish* ($0 \leq \alpha \leq 1$), where a zero command from the user is interpreted relinquishing control to the autonomy; and *blending-zero-stop* ($0 \leq \alpha \leq 1$), where a zero command from the user is interpreted as a stop command.

Figure 7 presents the blending of control commands issued by the user and the autonomy during a sample run in the presence of a detected autonomy goal when the user never reliquishes control (under sharing strategy *blending-zero-relinquish*).

Before the command $\mathbf{u}$ is passed to the robot for execution, it is assessed for safety by forward projecting ($3.0s$) the command and evaluating the resultant path for collisions. If the projected path collides with an obstacle, the control

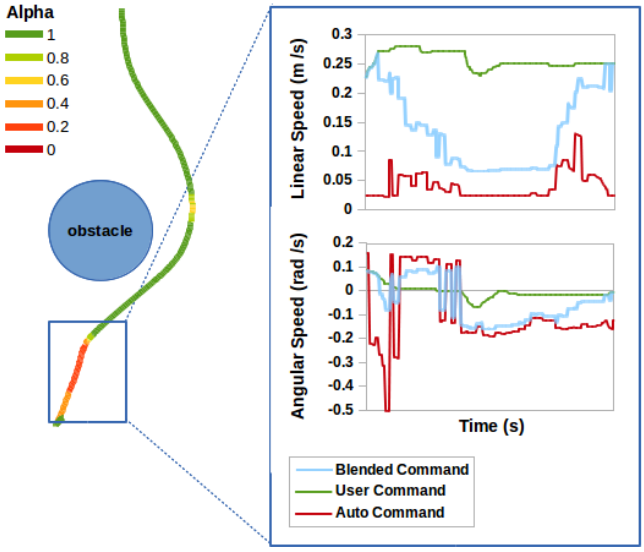

Fig. 8. Command blending to maintain safety. As the forward projection of the user's commands (green in plots) generate a path which collides with an obstacle, control is iteratively shifted from the user to the autonomy (by reducing the value of α). The resultant blended command (light blue in plots) prioritizes foremost safety, but also keeping as much control as possible (and within the constraints of $\boldsymbol{\theta}_\lambda$) with the user. Robot ground path shown with colors that reflect the value of the control blending parameter α at that time.

balance is iteratively shifted away from the user and to the autonomy, whose path planner is accounting for obstacles, until the resulting command projection no longer results in a collision.

Specifically, α is initialized to the value specified in $\boldsymbol{\theta}_\lambda$, but decremented according to $\alpha \leftarrow \alpha - \delta\alpha$ if the projected path collides with an obstacle. The decrementation is incremental, until either the projected command no longer collides with an obstacle or all of the control lies with the automation ($\alpha = 0$). This paradigm is a balance between competing aims: keeping control maximally with the user (i.e., α as large as possible), but limiting the number of forward projection roll-outs (to limit computational costs). In practice $\delta\alpha = \texttt{min}(\frac{\alpha}{5}, 0.1)$, and so the upper limit on the number of roll-outs that might occur is 10.

Figure 8 presents the blending of control commands issued by the user and the autonomy during a sample run in the presence of an obstacle and user commands that would collide with that obstacle.

4 Conclusions and Future Work

The customization of exactly which behaviors are selected for inclusion in $\mathfrak{B}_u$ is one mechanism by which customization to the user's physical abilities and preferences is accomplished. The other mechanism will be the *adaptation* of the autonomous behaviors in $\mathfrak{B}_u$, and of the control strategies Λ_u associated with them (Figure 1). The idea is to leverage machine learning to autonomously adapt the autonomy behaviors and control sharing strategies in order to customize to a user's physical abilities and personal preferences. The next step in the development of our software architecture thus will be to complete the development of and evaluate the adaptation modules.

The experimental work presented in this paper has provided the foundation and proof-of-concept for the technical components of our proposed modular and adaptive wheelchair robot. Our system prioritizes simple integration with existing commercial chairs and control interfaces, to mitigate costs not covered by insurance and thus accelerate adoption by users. Furthermore, our system is distinguished by its focus on customization to the user, via the selection and adaptation of a unique set of autonomy behaviors and control sharing strategies. The system has been evaluated within multiple environmental scenarios and shown good performance. The technical components are of course only one half of the story, and our future work will evaluate the operation of this system by limited-mobility users at the Rehabilitation Institute of Chicago—the #1 ranked rehabilitation hospital in the United States.

Acknowledgments. Many thanks to Matthew Derry for his significant contributions to the development of the software infrastructure for this system.

References

1. Fehr, L., Langbein, W.E., Skaar, S.B..: Adequacy of power wheelchair control interfaces for persons with severe disabilities : A clinical survey. Journal of Rehabilitation Research & Development **37**(3) (2000) 353–360
2. Simpson, R.: Smart wheelchairs: A literature review. Journal of Rehabilitation Research & Development **42**(4) (2005) 423–438
3. Simpson, R., LoPresti, E., Cooper, R.: How many people would benefit from a smart wheelchair? Journal of Rehabilitation Research & Development **45**(1) (2008) 53–72
4. Wang, Y., Chen, W.: Hybrid map-based navigation for intelligent wheelchair. In: Proceedings of ICRA. (2011)
5. Demeester, E., Hüntemann, A., Vanhooydonck, D., Vanacker, G., Degeest, A., Brussel, H.V., Nuttin, M.: Bayesian estimation of wheelchair driver intents: Modeling intents as geometric paths tracked by the driver. In: Proceedings of IROS. (2006)
6. Röfer, T., Mandel, C., Laue, T.: Controlling an automated wheelchair via joystick/head-joystick supported by smart driving assistance. In: Proceedings of ICORR. (2009)
7. Li, Q., Chen, W., Wang, J.: Dynamic shared control for human-wheelchair cooperation. In: Proceedings of ICRA. (2011)
8. Philips, J., del R. Millán, J., Vanacker, G., Lew, E., Galan, F., Ferrez, P.W., Brussel, H.V., Nuttin, M.: Adaptive shared control of a brain-actuated simulated wheelchair. In: Proceedings of ICORR. (2007)
9. Iturrate, I., Antelis, J.M., Kübler, A., Minguez, J.: A noninvasive brain-actuated wheelchair based on a p300 neurophysiological protocol and automated navigation. IEEE Transactions on Industrial Electronics **25**(3) (2009)
10. Katsura, S., Ohnishi, K.: Human cooperative wheelchair for haptic interaction based on dual compliance control. IEEE Transactions on Industrial Electronics **51**(1) (2004)
11. : The Smart Wheelchair from Smile Rehab Ltd: http://smilerehab.com/smart-wheelchair.php.
12. : The TAO-7 Intelligent Wheelchair Base from AAI Canada, Inc.: http://www.aai.ca/robots/tao_7.html.
13. Borgolte, U., Hoyer, H., Bühler, C., Heck, H., Hoelper, R.: Architectural concepts of a semi-autonomous wheelchair. Journal of Intelligent and Robotic Systems **22**(3-4) (1998) 233–253
14. Mazo, M.: An integral system for assisted mobility. IEEE Robotics & Automation Magazine **8**(1) (2001) 46–56
15. Braga, R.A., Petry, M., Reis, L.P., Moreira, A.P.: Intellwheels: Modular development platform for intelligent wheelchairs. Journal of Rehabilitation Research and Development **48**(9) (2011)
16. Urdiales, C.: Collaborative Assistive Robot for Mobility Enhancement (CARMEN): The bare necessities assisted wheelchair navigation and beyond. Springer Publishing Company, Incorporated (2012)
17. Desmond, R., Dickerman, M., Fleming, J., Sinyukov, D., Schaufeld, J., Padir, T.: Development of modular sensors for semi-autonomous wheelchairs. In: Proceedings of TePRA. (2013)
18. Civit-Balcells, A., Gonz, J.A.: Tetranauta: A wheelchair controller for users with very severe mobility restrictions. In: Proceedings of the 3rd TIDE Congress. (1998)

19. Prassler, E., Scholz, J., Fiorini, P.: A robotics wheelchair for crowded public environment. IEEE Robotics & Automation Magazine **8**(1) (2001) 38–45
20. Biddiss, E.A., Chau, T.T.: Upper limb prosthesis use and abandonment: A survey of the last 25 years. Prosthetics and Orthotics International **31**(3) (2007) 236–257
21. Lankenau, A., Röfer, T.: A versatile and safe mobility assistant. IEEE Robotics Automation Magazine **8**(1) (2001) 29–37
22. Bley, F., Rous, M., Canzler, U., Kraiss, K.F.: Supervised navigation and manipulation for impaired wheelchair users. In: Proceedings of SMC. (2004)
23. Parikh, S.P., Rao, R., Jung, S.H., Kumar, V., Ostrowski, J.P., Taylor, C.J.: Human robot interaction and usability studies for a smart wheelchair. In: Proceedings of IROS. (2003)
24. Zeng, Q., Teo, C.L., Rebsamen, B., Burdet, E.: A collaborative wheelchair system. IEEE Transactions on Neural Systems and Rehabilitation Engineering **16**(2) (2008) 161–170
25. Desai, M., Yanco, H.A.: Blending human and robot inputs for sliding scale autonomy. In: Proceedings of RO-MAN. (2005)
26. Simpson, R.C., Poirot, D., Baxter, F.: The hephaestus smart wheelchair system. IEEE Transactions on Neural Systems and Rehabilitation Engineering **10**(2) (2002)
27. System, T.N.A.W.N.: Simon p. levine and david a. bell and lincoln a. jaros and richard c. simpson and yoram koren and johann borenstein. IEEE Transactions on Rehabilitation Engineering **7**(4) (1999)
28. Bourhis, G., Horn, O., Habert, O., Pruski, A.: An autonomous vehicle for people with motor disabilities. IEEE Robotics & Automation Magazine **8**(1) (2001) 20–28
29. naki Iturrate, I., Antelis, J.M., Kübler, A., Minguez, J.: A noninvasive brain-actuated wheelchair based on a p300 neurophysiological protocol and automated navigation. IEEE Transactions on Industrial Electronics **25**(3) (2009)
30. Yanco, H.A.: Shared User-Computer Control of a Robotic Wheelchair System. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Boston, MA (2000)
31. Park, J.J., Johnson, C., Kuipers, B.: Robot navigation with model predictive equilibrium point control. In: Proceedings of IROS. (2012)
32. : Sip-N-Puff control interface from Therafin Corporation: http://www.therafin.com.
33. : Electronic head array from Adaptive Switch Laboratories: http://www.asl-inc.com.
34. Simpson, R., LoPresti, E., Hayashi, S., Nourbakhsh, I., Miller, D.: The smart wheelchair component system. Journal of Rehabilitation Research & Development **41**(3B) (2004)
35. Sharma, V., Simpson, R., LoPresti, E., Schmeler, M.: Evaluation of semiautonomous navigation assistance system for power wheelchairs with blindfolded nondisabled individuals. Journal of Rehabilitation Research & Development **47**(9) (2010)
36. : The Quantum 600 powered wheelchair from Pride Mobility: http://www.pridemobility.com.
37. : The MR-100 drive-by-wire wheelchair-base robot from Sensible Machines: http://www.sensiblemachines.com/enabling-modules.php.
38. Derry, M., Argall, B.: Automated doorway detection for assistive shared-control wheelchairs. In: Proceedings of ICRA. (2013)