

HelpStar Technology for Semi-Autonomous Wheelchairs

L Deligiannidis, WD Potter, BJ Wimpey, H. Uchiyama, R. Deng, S. Radhakrishnan, and D. Barnhard
Computer Science Department and Artificial Intelligence Center
University of Georgia, Athens GA, USA
ldeligia@cs.uga.edu

Abstract. This paper describes our virtual reality research prototype for supporting a semi-autonomous wheelchair. The VR component enables the “HelpStar” feature, which provides a user who is visually impaired with mobility independence. Our “HelpStar” enabled semi-autonomous wheelchair functions more like a personal assistant, allowing much greater user independence. When the user finds them self in an unforeseen circumstance, the “HelpStar” feature can be activated to allow a remote operator to use VR technologies to provide helpful navigational instructions or to send commands directly to the wheelchair. This paper demonstrates the successful integration of VR technologies with semi-autonomous remote vehicles for assistive support.

Keywords

Systems for Real-Life Applications, Human-Robot Interaction, Robotics, Semiautonomous Vehicles, Virtual Reality

Introduction

A semi-autonomous (SA) wheelchair is an electric powered wheelchair that contains perceptual and navigational capabilities for assisting a person who is visually impaired and using a wheelchair. The goal of an SA wheelchair is to improve the independent mobility of individuals with multiple disabilities based upon integrated sensory information and human-machine interaction. In a nutshell, the SA wheelchair provides the user with enough information about the environment to allow the user to navigate effectively. This is similar to the assistance a sighted, human attendant might provide while assisting with moving the user from one location to another. The user actually controls the motions of the wheelchair but is directed by the attendant.

However, there are circumstances where the SA wheelchair user might need assistance with overcoming some unforeseen predicament. Usually, this requires the user to ask a passerby for assistance or to telephone a nearby friend to come help out. When owners of General Motors vehicles with the OnStar feature face some sort of difficulty while driving, they can request assistance from the OnStar service staff with the touch of a button. Likewise, stay-at-home customers of ADT's Companion Services contact the ADT 24-hour help staff by pressing the button on their personal alert device. Our virtual reality help system (called HelpStar) provides a similar feature but for a different type of user; the visually-impaired wheelchair user.

With the touch of a button, a member of the HelpStar staff makes contact with the SA wheelchair user having difficulty. The sensory information routinely collected by the wheelchair is instantly forwarded to the HelpStar center. This information is used to establish a virtual environment in the HelpStar center that reflects the environment encountered by the wheelchair user. This allows the HelpStar staff to analyze, diagnose, and resolve the current problem faced by the user. Corrective feedback could either be in the form of commands to the user (similar to what a local human attendant might do), or commands directly to the SA wheelchair. In either case, the user's immediate problem is resolved with the minimum amount of local interference, and they are free to continue with their activity such as going to class.

The key concept behind the HelpStar project is independence. The SA wheelchair provides an enormous amount of mobility independence to the (essentially blind, using a wheelchair) user. HelpStar provides immediate assistance when the user encounters a problem. However, more importantly, HelpStar provides security and peace-of-mind to the user; if they need help, they know help is just a button push away. The remainder of this paper describes the approach we are taking to develop the HelpStar system. We discuss the major aspects of our semi-autonomous wheelchair, the sensory information acquisition systems, and the HelpStar virtual reality feature. We conclude the paper with a discussion of the current HelpStar prototype implementation.



Figure 1: The Power Wheelchair (Invacare Nutron R-32).

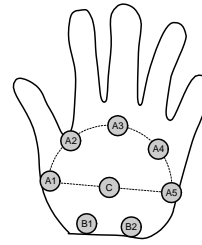


Figure 2: The arrayed motors of the Vibrotactile Glove

Background

Most public institutions and facilities, such as universities, provide certain types of disability services. For example, the University of Georgia provides an on-campus curb-to-curb van transportation service to students with mobility, visual, and other health-related impairments. Students with disabilities need not worry with outdoor (building to building) transportation. However, no official attendant service is provided for navigating within a university building. This is typically the case on nearly all public university campuses. In addition, many universities have a rich heritage of historic building architecture. Unfortunately, many of these older buildings are not disability friendly. Even when situated in a disability friendly building, maneuvering to a particular destination is not an easy task without the aid of a sighted human attendant.

A number of studies have been conducted in the field of assistive technology which combine robotics and artificial intelligence to develop autonomous wheelchair control. Many of these autonomous wheelchairs are equipped with a computer and a set of sensors, such as cameras, infrared sensors, ultrasonic sensors, and laser rangefinders. This assortment of equipment is used to address a number of specific problems such as: obstacle avoidance, local environment mapping, and route navigation. With autonomous control, the system probes the environment, detects an obstacle, plans a navigation route, makes a decision, and actually controls the wheelchair. The user simply goes along for the ride. Consequently, the system is ultimately responsible for the results, which leaves the user totally dependent upon the equipment. Most of these autonomous wheelchairs have been employed for research purposes only. *NavChair*, developed at the University of Michigan [10], transports the user by autonomously selecting three different modes (tasks): obstacle avoidance, door passage, and wall following. The *Tao* series provided by Applied AI Systems Incorporated is mainly designed for indoor use and features escape from a crowd and landmark-based navigation behaviors in addition to the three common tasks accomplished by *NavChair* [6]. *Tinman II* [13] and *Rolland* [9] also provide similar functionalities. In each case, the user is not involved with the motion of the wheelchair but is a passenger.

Our goal is to customize a standard wheelchair with enough information gathering capability to allow an unsighted user to effectively control it. Our base wheelchair is a standard power chair (Figure 1) that consists of two front pivot wheels, two rear motorized wheels, a battery pack, and a controller (joystick). The perceptual navigation system consists of a computer, a collection of sensors (e.g. ultrasonic, infrared, and CCD camera), and a man-machine interface.

An SA wheelchair automatically acquires sensory inputs from the environment, processes them, and provides navigational information transformed to fit the user's available sensory resources, such as audible or tactile perception. As a man-machine interface, we developed a tactile "display" designed for the back of the hand, which consists of an array of very small vibrating motors (Figure 2: the Vibrotactile Glove). The Vibrotactile Glove conveys relatively simple navigational and environmental information by activating one or more vibrating motors, which can be intuitively interpreted by the user. By wearing the Vibrotactile Glove connected to the SA wheelchair, the user is able to expand their limited sensory perception (i.e., combine their own sensory perceptions with those of the on-board sensors) for use with navigational decision making. In other words, the user has navigational control over the wheelchair, and uses available sensory information and system commands to pilot the wheelchair.

Our SA wheelchair is designed for users with multiple disabilities (mental disabilities are excluded), specifically users with a combination of physical and sensory disabilities. In the United States over two million individuals are bound to wheelchairs, 67% of which report suffering from two or more disabilities. Likewise 1.8 million people in the United States are counted as having impaired eye-sight including blindness, 63% of which have multiple disabilities (2000 US Census data). A growing number of elderly individuals in the United States and other countries are also potential users of the SA wheelchair.

Environmental information provided by our on-board system of sensors combined with decision making information is passed to the user in the form of navigational commands. The user receives these commands through the Vibrotactile Glove where different commands are presented as different vibration sequences via the small motors. However, there will surely be times when the user encounters a situation where they are in need of assistance. A human care attendant can assist with these sorts of emergencies, but having an attendant available all the time may not be possible and certainly does not improve independent mobility for the user. HelpStar is designed to provide the necessary assistance without the need for an attendant by utilizing virtual reality (VR) technology.

There are a number of studies that have been conducted, as well as some existing consumer applications, that employ the combination of VR with assistive technology. Most of these efforts focus upon training novice wheelchair users using a wheelchair simulator or virtual environment [1, 8]. Gundersen and his team [7] studied the use of virtual presence control on board a wheelchair at Utah State University. In their project, the on-board system was connected to the remote control booth via an RS-232 serial radio frequency (RF) link. Due to limitations with the RF band, the maximum range between the wheelchair and the remote center was approximately 1000 feet. The wheelchair was manipulated either by an attendant using the remote system or by the on-board (fully) autonomous control system. In either case, the user was not involved with control of the wheelchair.

Utilizing VR technology for remote attendance, we enrich our SA wheelchair control system by providing an “on-demand” care attendant to the SA wheelchair user. When the user hits the “HelpStar” button, the SA wheelchair control system connects to the remote attendant, the HelpStar staff member. The environmental information collected by the SA wheelchair’s sensors, and the images acquired by the on-board camera(s) are transmitted to the HelpStar center via the Internet. The equipment available at the HelpStar center re-creates (in a virtual world) the situation encountered by the SA wheelchair user. Of course, the primary limitation is the necessary existence of a wireless cloud in the user's location. However, most college campuses (especially campus buildings and surrounding areas) are enclosed within a wireless cloud with direct access to the Internet.

Our Current Prototype

Our current prototype development efforts are divided into two directions: the SA wheelchair, and the HelpStar system. Our SA wheelchair is described in detail in [16]. This section discusses the HelpStar prototype; our proof of concept implementation. The hardware utilized for the current HelpStar platform is a commercially available robot kit called ER1, which is supplied by Evolution Robotics [5]. The robot kit includes control software, aluminum beams and connectors for constructing the chassis, two assembled non-holonomic scooter wheels powered by two stepper motors, one 360 degree rotating caster wheel, a power module, a 12V 5.4A battery, and a web-camera. A Dell Latitude C640 laptop computer (Intel Mobile Pentium 4 processor 2.0GHz with 512 MB RAM running Windows XP) is used as the controller device. Additional accessories were also used such as a one-dimension gripper arm, infrared sensors, and additional aluminum beams and connectors. The chassis is reconfigurable and this enables us to design a chassis that meets our needs. The laptop is equipped with a PCMCIA card that provides four additional USB ports. The ports are utilized by the web-camera, the infrared sensors, the gripper, and the stepper motors. The software that comes with the ER1 robot, which is called the “ER1 Robot Control Center”, can be placed in one of three different configurations.

1. Remotely control an ER1 using another instance of the Control Center on the remote machine.
2. Remotely control an ER1 using TCP/IP.
3. Control the ER1 by running behaviors.

The first configuration enables one to control the ER1 remotely from another computer via a peer to peer connection using another instance of the Control Center on the remote computer. The second configuration enables one to open a TCP connection to a specified port on the Control Center and send ER1 commands to it such as move, open, and close via a wireless internet connection. In the third configuration one can specify on-board behaviors that the robot will execute such as find a specific object and then play a sound.

More complex behaviors can be specified using Evolution's toolkit called ERSP. With the behaviors, one can instruct the robot to find different objects or colors, and perform an action when certain conditions are met. The Control Center contains a module to recognize objects seen by the mounted web-camera. We instructed the Control Center to accept commands from a remote machine for its operations, using configuration 2. We placed the camera a little bit behind the chassis in order for the gripper to be in the web-camera's field of view. We also placed the gripper as far as possible from the laptop to avoid dropping objects accidentally on top of the laptop. The following is a picture of one of our robots showing the placement of the web-camera and the gripper.



Figure X001. Picture of the customized ER1 robot.

Interacting With The Robot

We developed a new user interface based on Virtual Reality to remotely control multiple ER1 robots (the idea being that the HelpStar center might need to provide multiple, concurrent assistance). The Virtual environment consists of three-dimensional objects that each represents a robot (an SA wheelchair user). These 3D objects are referred to as TVs, (televisions). The position and orientation of these TVs in the Virtual Environment are unrelated to the physical position and orientation of the real robots. The TVs could be any three-dimensional object but we utilized simple cubes, as shown in figure X002.

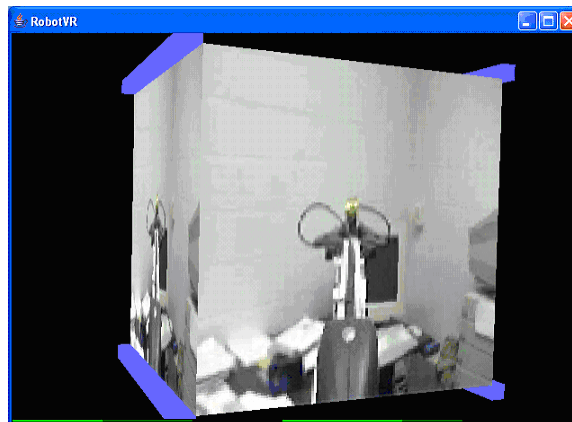


Figure X002. Snapshot of a TV; video stream from the robot's web-camera is texture mapped onto a cube (robot's gripper is visible)

The images from the robots' web-cameras are transmitted to the remote machine utilizing RTP (Real Time Protocol). These live feeds from the robots' web-cameras are converted into images that we texture map onto the TVs; we utilized Java's Media Framework (JMF) to implement this part of the application. This enables a fully immersed person (the HelpStar attendant) to walk around the TVs and see whatever the web-cameras of the robots see. The live feeds from the robots' cameras are transmitted to the VR machine. The VR machine is attached to an electromagnetic tracking system, LIBERTY™ [14], which consists of a six-degree-of-freedom (6DOF) tracker with three sensors; LIBERTY™ supports up to eight sensors. One sensor is attached to the Head Mounted Display (HMD) and the other two sensors are attached to the attendant's left and right hands. We also utilize two Pinch Gloves™ provided by Fakespace Systems Incorporated to recognize gestures and send commands to the robots. We have a couple of HMDs where

one of them has stereo capability. We also have three different PCs that are capable of driving the application, all of which are equipped with high-end video cards. The VR machine is also attached to an eye-tracking machine. We currently use the eye-tracking machine to simply select a desired TV. The fully immersed person (the HelpStar attendant) can pick up any of the TVs, move them, rotate them, and group them together to place related TVs together. The TVs have some decoration around them to easily distinguish the different TVs. The decoration could include some other objects around the TVs or the name of the user on top of the TVs. This functionality is implemented using Pinch Gloves™ and a LIBERTY™ six-degree-of-freedom (6DOF) tracker. When the attendant's hand intersects with one of the TVs and the attendant performs the gesture shown in Figure 4, the selected TV follows the motion of the attendant's hand until they release the TV as show in Figure 5. The attendant can utilize both of his/her hands to pick up two TVs, or simply pick up one TV with one hand and hand it over to the other hand; the application is aware of two hand interaction.



Figures 4 & 5. Grasping and Releasing a TV. *May show more gestures*

The HelpStar attendant using eye-tracking technology can select one of the three-dimensional objects (TVs) that represent a robot. Since the attendant may simply look around and not want to select a particular TV, to select a TV they have to look at it and then perform a gesture, as shown in Figures 6 and 7 to select the TV being looked at.



Figure 6 & 7. Transition to the “Select” state.

When the TV is selected, the TV's position and orientation change dynamically so that it is always in front of the attendant, even if the attendant moves around. There can be only one TV selected. To deselect a TV the attendant performs the same gesture again. The application has eight states and is aware of state transitions; actions may be performed on a state or at a state transition as shown in Figure X003.

Figure X003 shows all eight states and the gestures needed to move to a particular state. The “Idle” state is a state that indicates no communication with the robots, except that the application is receiving live feed information from the robots' cameras, yet there is no interaction between the attendant and the TVs. While in the “Idle” state, the attendant can pick up a TV with their left or right hand, or even both. The attendant needs to touch a TV and perform a gesture to attach the TV to their virtual hand; the gesture is: touch the thumb and the index finger. As soon as the attendant releases the touching fingers, the hand-TV relationship is terminated and the TV does not follow the attendant's hand anymore. The state machine reverts back to the “Idle” state. While in the “Idle” state, the attendant can also look at a TV and then touch and release the right thumb and middle fingers to select a TV. This transitions the state machine to the “Selected” state where the TV is locked in front of the attendant's field of view. As the attendant moves around, the TV appears in front and the attendant does not see the rest of the Virtual Environment that primarily consists of other TVs. This is the main state of the state machine where the attendant can either deselect the TV or send commands to the robot.

The speed of the robot can be changed from the “Speed” state; the state machine transitions to the “Speed” state from the “Selected” state when the user pinches and releases his/her thumbs. There are two speeds that can be modified: 1) the linear velocity – forward/backward speed, and 2) angular velocity – how fast the robot rotates. Each of these speeds can be incremented and decremented. The minimum linear velocity is 5cm/sec and the maximum is 50cm/sec. The minimum and maximum values for the angular velocity are 5 and 90 degrees/second, respectively.

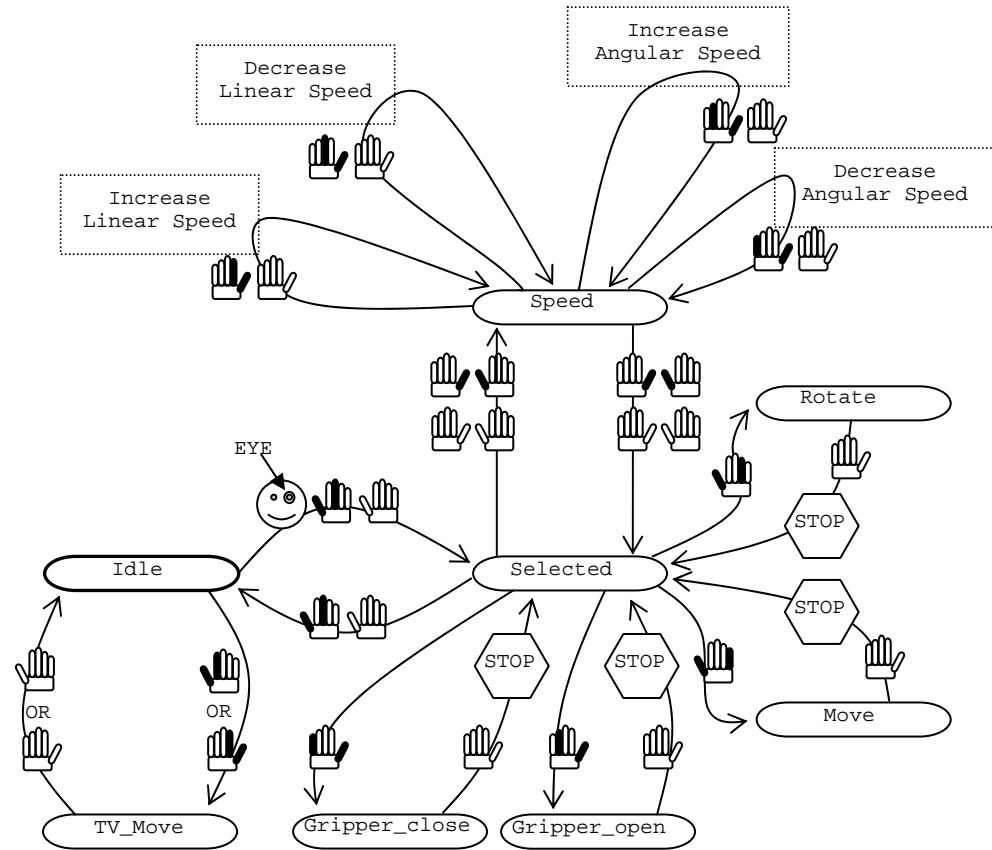


Figure X003. The state machine.

Note that the ER1 is not capable of changing velocity settings while it is in motion; this is a limitation of the ER1's hardware. The velocity settings are shown at the bottom of 3D environment as two greenish bars as shown in figure X006. The left bar is the indicator for the linear speed, and the right one is the indicator for the angular velocity. The bars have two shades, the darker shade shows all values that can be assigned where the highlighted bar as shown in figure X006, which is drawn on top of the darker bar, shows the current setting. The velocities can be incremented or decremented in units of five. These 2D widgets only occupy a couple of pixels at the bottom of the display and do not occlude the HelpStar attendant's field of view. The widgets are only visible when a TV is selected, and the settings are stored within a TV allowing each TV to have its own settings.

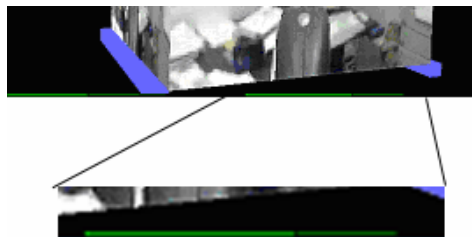


Figure X006. The angular velocity widget enlarged

The gripper operates using the left thumb and the left pinky and ring fingers. As long as the state machine is in one of the "Gripper_open" or "Gripper_close" states, the gripper keeps opening or closing respectively. Upon releasing the fingers the state machine transitions to the "Selected" state at which point the "stop" command is transmitted. The stop command instructs the robot to cancel any operation that is being executed. This enables the attendant to partially open or close the gripper. The other two states are

used to maneuver, rotate left or right, and move the robot forward or backwards. When the state machine transitions from either the “Move” or “Rotate” states to the “Selected” state the “stop” command is transmitted to stop the robot. We use two states, one for the rotation and one for the move because of the robot’s limitations. An ER1 cannot move and at the same time rotate. So, either the attendant can instruct the robot to move straight (forward or backwards) or rotate (clockwise or counterclockwise). To instruct the robot to move forward, the attendant needs to simply lean forward and pinch the right thumb and pinky fingers. Similarly, to instruct the robot to move backwards the attendant simply needs to lean backwards and perform the same pinch. Since there is a Polhemus 3D sensor attached to the attendant’s HMD to track their position and orientation in space, we define a plane in space that divides the space into two parts as shown in Figure 9. In Figure 9 we only show two of the dimensions but in real life a plane instead of a line is defined. We keep track of the attendant’s position orientation continuously and upon the appropriate gesture we define the plane in space. The attendant can move between the divided space to instruct the robot to move forward or backwards. To instruct the robot to rotate clockwise or counterclockwise, the attendant first needs to perform the right gesture for the state machine to transition to the “Rotate” state at which point the robot follows the rotation of the attendant’s head. If the attendant rotates his/her head 20 degrees to the left, the robot also rotates 20 degrees to the left. Since the robot’s motors are not as fast as the attendant’s head rotation speed, the attendant should rotate slowly to give enough time to the robot to perform the rotation. The rotation angle we are tracking in real time is the rotation around the Y axis, which is pointing upwards.

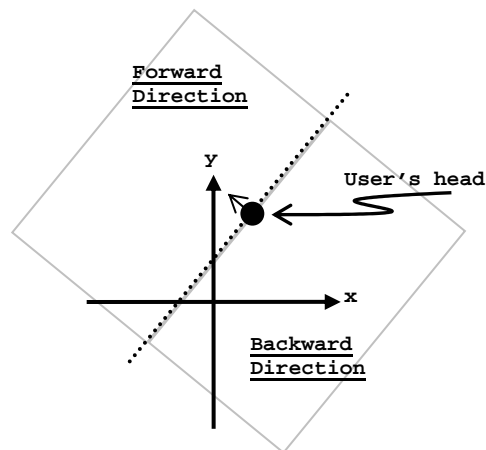


Figure X002. Splitting the space in half to define direction of “move”.

The rotation or direction of the robot depends on local coordinates. That means that even if the attendant rotates his/her body 180 degrees, forward means forward to the robot and the attendant’s left means left to the robot, something that is not true if one tries to maneuver the robot using a conventional mouse. Even if one uses the “Control Center” to remotely control the ER1, changing the speed of the robot would require multiple mouse clicks on different windows. However, utilizing a Virtual Reality interface makes remotely operating an ER1 seem more natural, and the attendant can send more commands to the robot by simple gestures/postures.

Application Layer Protocol Description

We utilized Java’s Media Framework (JMF) to implement the prototype. Java Media Framework (JMF) is an optional Java package that specifies a unified architecture to synchronize and control audio, video, and other real-time based data within Java applications and applets. JMF supports Real-time Transport Protocol (RTP), and Real Time Control Protocol (RTCP). RTP provides end-to-end network transport functions for applications transmitting real-time data, and RTCP allows monitoring of the data delivery in a manner scalable to large multicast networks. RTP seems to be the most suitable protocol for transmitting streaming audio and video feeds. RTP is built on top of UDP, which is by nature an unreliable transport protocol but computationally inexpensive compared to TCP. On the other hand, controlling the hardware of the robot and the initiation and termination of the audio/video stream transmission, we utilized TCP to guarantee the delivery of our commands from the VR machine to the robot’s computer. During the

initialization, the VR machine connects to all robots in a five step connection oriented manner as shown in the figure below.

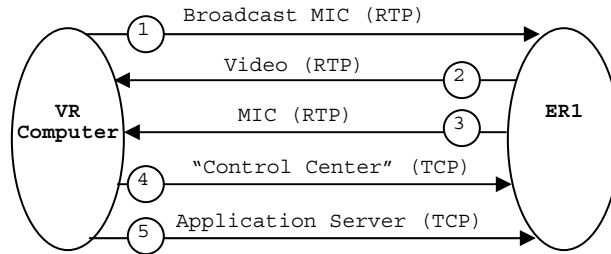


Figure X1. Communication Protocol; initialization phase

The VR machine first starts broadcasting the audio stream generated by the microphone attached to the fully immersed person, the HelpStar attendant, to all robots (number 1 connection in the figure above). As soon as the robots accept the connection, they begin transmitting audio and video streams on different channels to the VR machine using RTP (connections 2 and 3 in the figure above). When the connections are accepted by the VR machine, the VR machine opens two more connections, using TCP, to each robot. The first connection, number 4 in the figure above, is a connection to the “Control Center” of the ER1. Using this connection the VR machine can instruct a robot to move forwards/backwards, open/close the gripper, rotate, etc. The second TCP connection is used to instruct the application on the robot to resume or suspend the transmission of audio and/or video; primarily to save network bandwidth.

Even though RTP is a connectionless protocol, in order for the receiver to set up its sockets, a sender must be activated in order for the receiver to associate a socket with a particular client. That is why at the initialization phase all transmitters and receivers on the sender and on the receiver side are activated. Note that the VR machine as well as the robots contain receivers and transmitters; they both receive audio streams as shown in number 1 and 3 in the figure above. However, after all connections are established, the VR machine instructs the robots’ transmitters and receivers to suspend operation. The following table describes the sequence of instructions that are transmitted via the TCP connection from the VR machine to the robots, connection number 5 in the figure above, after all connections are established. The table also shows the state of each transmitter and receiver.

Table 1. State of the transmitters and receivers

	VR machine	Robot
1	Transmitter stops transmission of microphone after all robots complete the initialization phase.	Receiver suspends reception of VR machine’s microphone.
2	VR machine starts receiving video streams from each robot and does not suspend reception until a TV is selected.	Transmitter of video stream begins transmission and does not suspend operation until this robot is selected later.
3	As soon as the robot’s microphone stream connection is accepted, the receiver suspends its operation, and instructs the robot to suspend transmission.	After the connection of the robot’s microphone stream connection is accepted, it suspends the transmitter’s operation.
4	Only one connection is allowed by the “Control Center”. The connection stays up throughout the execution of the system.	Receiver accepts the TCP connection and is listening for commands that control the ER1.
5	Only one connection exists to the application which controls the transmitters’ and receivers’ operations.	Receiver accepts the TCP connection and is listening for administrative commands.

The pseudo-code below illustrates the sequence of operations that are involved when a TV is selected by the HelpStar attendant. When the TV, which is the representation of a robot, is selected with the appropriate gesture (lines 1 – 3), we instruct the robot (line 6) to start transmission of its audio stream,

which is the sound captured by the on-board microphone. We also instruct the robot to start the receiver that receives the stream of the VR machine's microphone transmission (line 7). We then (line 8) instruct every other robot to suspend transmission of the video stream to preserve network bandwidth; this gives all of the bandwidth to the robot's video transmitter so we can get as many frames as possible. We finally display and update the linear and angular speed widgets at the bottom of the fully immersed person's screen.

```

1.  if( pinch(Right_MIDDLE, Right_THUMB) && (state==STATE_IDLE) ) then
2.      target_TV = GetTVfromEyeTracker();
3.      if( target_TV is a TV ) then
4.          Select the TV target_TV
5.          Move state machine to the SELECTED state
6.          Instruct robot target_TV to start transmitting its MIC stream
7.          Instruct robot target_TV to start receiving out MIC stream
8.          Instruct every other robot to stop transmission of VIDEO
9.          Start updating speed widget at the bottom of the 3D window

```

When the HelpStar attendant deselects a TV with the appropriate gesture, we instruct the robot to suspend its microphone's transmission, and also to stop receiving the HelpStar's audio stream, so that nothing will be played back on the speakers of the robot. We then instruct every other robot to resume transmission of their video streams; note that the target's video transmission was never suspended. We then hide the speed widgets from the screen since there is no robot-speed association.

```

1.  if( pinch(Right_MIDDLE, Right_THUMB) && (state==SELECTED) ) then
2.      Move state machine to the IDLE state
3.      Instruct robot target_TV to stop transmitting its MIC stream
4.      Instruct robot target_TV to stop receiving out MIC stream
5.      Instruct every other robot to start transmission of VIDEO
6.      Stop updating speed widget at the bottom of the 3D window
7.      target_TV = -1; // no robot is selected

```

- **Configuration file** Should we talk about the configuration file that is shared among all robots and the VR machine? SURE.

Conclusions

HelpStar is our proposed system for remote assistance to a semi-autonomous wheelchair user using Virtual Reality as an invisible assistive service. The system is specifically designed for individuals who are visually-impaired, use a wheelchair, and want to be involved with their own mobility. A single HelpStar attendant can virtually see multiple users and provide immediate assistance to one or more of them. The SA wheelchair employed in the design allows the user to expand their limited sensory perception for use in navigational decision making. If the SA wheelchair user encounters an unusual situation, all they have to do is push a button to contact the HelpStar center. The key idea, the feature that makes this all worthwhile, is to provide mobility independence to the user, something lacking under the watchful eye of a human attendant.

References

1. Adelola, I. A., Cox, S. L., and Rahman, A., (2002). Adaptive Virtual Interface for Powered Wheelchair Training for Disabled Children, In *Proc. of 4th Intl. Conference of Disability, Virtual Reality & Assoc. Technology*, Veszprém, Hungary, pp. 173-180.
5. Evolution Robotics, (2004), Evolution Robotics ER1 Robot Kit, Retrieved October 12, 2004, from <http://www.evolution.com/education/er1/>
6. Gomi, T. and Griffith, A. (1998) Developing intelligent wheelchairs for the handicapped. In Mittal et al. eds., *Assistive technology and AI*. LNAI-1458, Berlin: Springer-Verlag, pp. 150-78.
7. Gundersen, R. T., Smith, S. J., and Abbott, B. A. (1996) Applications of Virtual Reality Technology to Wheelchair Remote Steering System, In *Proc. of 1st Euro Conf of Disability, Virtual Reality & Assoc. Technology*, Maidenhead, UK, pp. 47-56.
8. Inman, D. P., and Loge, K. (1995). Teaching Motorized Wheelchair Operation in Virtual Reality. In *Proceedings of the 1995 CSUN Virtual Reality Conference*. Northridge: California State University, Retrieved October 1, 2004 from; <http://www.csun.edu/cod/conf/1995/proceedings/1001.htm>

9. Lankenau, A., Röfer, T. and Krieg-Bruckner, B. (2003) Self-localization in large-scale environments for the Bremen Autonomous Wheelchair. In Freksa and et al. eds., *Spatial Cognition III*. LNAI-2685. Berlin: Springer-Verlag, pp. 34-61.
10. Levine, S.P. and et al. (1999) The NavChair Assistive Wheelchair Navigation System. *IEEE Transactions on Rehabilitation Engineering* 7(4): pp. 443-51.
13. Miller, D. (1998) Assistive robotics: an overview. In Mittal et al. eds., *Assistive technology and AI*. LNAI-1458. Berlin: Springer-Verlag, pp. 126-136.
14. Polhemus Inc., (2004), LIBERTY™ , Retrieved October 12, 2004, from <http://www.polhemus.com/LIBERTY™.htm>
16. Uchiyama, H. (2003) Behavior-Based Perceptual Navigational Systems for Powered Wheelchair Operations, *Master Thesis Proposal at the University of Georgia*, Retrieved October 11, 2004, from <http://www.cs.uga.edu/~potter/robotics/HajimeThesisProposal.pdf>
- 19 Polhemus Inc.'s home page <http://www.polhemus.com/>
20. FakeSpace Inc.'s home page <http://www.fakespace.com>