Paralyzed Subject Controls Telepresence Mobile Robot Using Novel sEMG Brain-Computer Interface: Case Study

Kenneth R. Lyons, *Student Member, IEEE* and Sanjay S. Joshi, *Senior Member, IEEE*
Department of Mechanical and Aerospace Engineering
University of California, Davis
Davis, CA, USA 95616
ixjlyons@gmail.com, ctljoshi@gmail.com

***Abstract*—Here we demonstrate the use of a new single-signal surface electromyography (sEMG) brain-computer interface (BCI) to control a mobile robot in a remote location. Previous work on this BCI has shown that users are able to perform cursor-to-target tasks in two-dimensional space using only a single sEMG signal by continuously modulating the signal power in two frequency bands. Using the cursor-to-target paradigm, targets are shown on the screen of a tablet computer so that the user can select them, commanding the robot to move in different directions for a fixed distance/angle. A Wifi-enabled camera transmits video from the robot's perspective, giving the user feedback about robot motion. Current results show a case study with a C3-C4 spinal cord injury (SCI) patient using a single auricularis posterior muscle site to navigate a simple obstacle course. Performance metrics for operation of the BCI as well as completion of the telerobotic command task are developed. It is anticipated that this noninvasive and mobile system will open communication opportunities for the severely paralyzed, possibly using only a single sensor.**

***Keywords*—human-machine interface, brain-computer interface (BCI), surface electromyography (sEMG), mobile robot, brain-muscle-computer interface (BMCI)**

I. Introduction

We employ a new muscle-based brain-computer interface which uses the sEMG signal of a single muscle site to control computers and other devices with a cursor moving in multiple dimensions. In this system, the neuromuscular system is trained to send the appropriate required electrical signal to the auricularis posterior muscle site (in back of the ear). The auricularis posterior is available even to spinal cord injured individuals, as head and face muscles are innervated at the brainstem and not the spinal cord. Users continuously manipulate the partial powers of two power spectral density frequency bands simultaneously, creating two independent continuous control channels from a single sEMG signal. This type of frequency band control is unlikely to be needed for general muscle contraction, thus representing a new learned neuromotor skill at the electrical level. In the current study, a paralyzed subject continuously guides a computer cursor in two dimensions from the origin location to three button locations on the screen, each representing a different mobile robot action (move forward, turn left in place, turn right in place). The user then guides a telepresence mobile robot through an obstacle course using the robot's view as feedback.

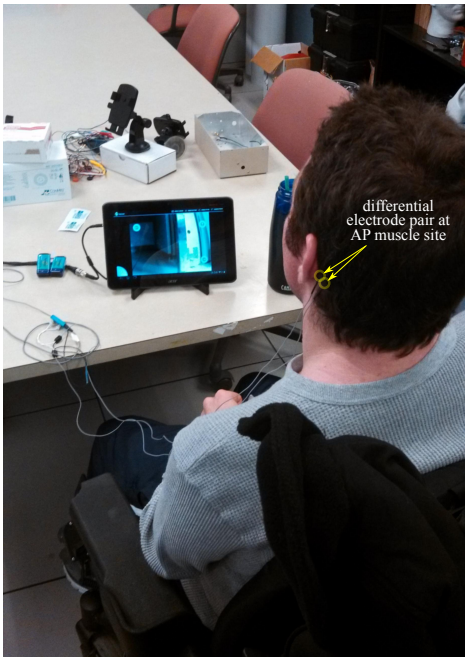


Fig. 1. A picture of the disabled participant in our study using the BCI. The tablet computer running the BCI and robot control software is shown resting in front of the participant on the table. A differential pair of Ag-AgCl electrodes are shown placed on the participant's left auricularis posterior muscle.

Our method is fundamentally different from traditional sEMG prosthetics research, in which total signal strength from a single muscle site is used to continuously control one dimension of movement. As such, two muscle sites are usually required for two continuous degrees of freedom (such as 2D cursor control), whereas our system uses one muscle site to achieve two continuous controllers. Alternatively, previous prosthetics research methods have used a single muscle site to select one of several functions for the smart prosthetic to take over and perform, which is referred to as discrete function control. These systems typically employ sophisticated classifiers, which sample a natural contraction at a muscle site in open loop and categorize it as a specific intended prosthetic movement. No notion of continuously guiding an object in closed loop is considered. Our system uses no classifiers at all to achieve multidimensional control from a single sensor.

The foundation for the work presented here is given in [1] and [2], where the technique of utilizing the power in two frequency bands of a single sEMG signal to control a cursor in two-dimensional space was established. The signal processing involved and the mapping from frequency band power to cursor position is described in Section II. Other work on this basis includes controlling a wheelchair [3] and porting the BCI software to run on a mobile phone [4].

Although the underlying processes driving our BCI controller is unique, there has been some previous work done on control of mobile robots using brain-computer interfaces, most of which is based on electroencephalography (EEG). Millán *et al.* [5] used a classifier approach to recognize three mental states based on EEG recordings. These states were then used in the discrete function control of Khepera robots to go forward, turn left, turn right, etc. Escolano *et al.* [6] used EEG to control a mobile robot with some autonomy and image processing capabilities by allowing the user to select from a scanning grid of robot locations viewed from the robot's perspective. Xue *et al.* [7] presented another BCI discrete function controller, heavily emphasizing the integration of a BCI and Internet-based teleoperation of a mobile robot. Gergondet *et al.* [8] investigated the degradation of performance with an EEG-VEP-based BCI for control of a humanoid robot when the background of the user interface is changing (video feedback). Note that all EEG systems use multiple signals measured from the scalp, whereas our system needs only a differential pair of electrodes at a single muscle site. In addition, EEG signals are 10-100 times smaller in magnitude than sEMG signals.

II. Brain-Muscle-Computer Interface

The system described in [4] for a smartphone is extended here to utilize the larger screen size as well as the standalone audio connectivity of tablet computers. Using the increasingly common 3.5 mm TRRS phone connector – the standard connector for headsets which output stereo audio and have mono microphone input – we can directly connect a sEMG sensor-amplifier package to many different devices. Figure 2 shows the setup we currently use. A differential pair of 4 mm shielded Ag-AgCl electrodes from BIOPAC are connected to a Motion Lab Systems EMG preamplifier. An off-the-shelf headset adapter splits the headset signals to mono microphone input and stereo headphone output. This allows us to send the single amplified signal directly to the tablet's microphone input hardware while still allowing for the use of headphones or speakers.

A. Signal Processing

Once the connection is made between the sEMG sensor and the tablet's microphone input hardware, we can begin the signal processing procedure. Typically, our sEMG signals are below 1500 Hz, so we sample the signal at 8000 Hz using the tablet's audio recording functionality and downsample to 4000 Hz. Samples are fed into a fixed-length buffer which, when filled, is copied so that we obtain two channels with initially the same data. Each of the channels is processed through a bandpass filter with a different passband (80-100 Hz and 130-150 Hz, chosen ad-hoc) and subsequently the channels are referred to as *bands*. The signal power in each band is then calculated via Parseval's theorem. The rest of the signal processing procedure can then be thought of in two stages: normalization and cursor positioning. After these stages are complete, the cursor position is used to draw the cursor on screen and the process repeats, starting with bandpass filtering of the new data in the buffer. Currently, the buffer size is set such that the cursor position is updated at a rate of 4 Hz.

1) Normalization: Before beginning a cursor control session, the user must go through a calibration process. First, the user is asked to contract the muscle being measured at a comfortable level for two seconds a total of three times. While the muscle contracts, the maximum power values in each of the two bands is recorded. These two maximum values, $P_{1_{\max}}$ and $P_{2_{\max}}$, are then used for normalization to account for slight changes from session to session caused by changes in muscle state, environment, etc. A second level of calibration is performed with a step that customizes the BCI through *effort* values, e_1 and e_2, which are variable between 1 and 100 (though held constant after calibration is finished). These can be thought of as specifying the percentage of $P_{1_{\max}}$ or $P_{2_{\max}}$ needed to maximize the inputs to the cursor positioning stage. Implemented in this way, the effort values help to avoid muscle fatigue and they allow for adjustment of the space of band values reachable by the user, which are calculated as

$$b_1(n) = \frac{P_1(n)}{P_{1_{\max}}}\frac{100}{e_1}, \quad b_2(n) = \frac{P_2(n)}{P_{2_{\max}}}\frac{100}{e_2} \tag{1}$$

where $P_1(n)$ and $P_2(n)$ are the power values measured at time step n. Note that because the power values are always positive and the effort values are always between 1 and 100, the coordinate pair, (b_1, b_2), will always exist in the first quadrant of the Cartesian plane. The region of the quadrant that the point lies in is further restricted by the fact that zero power is not achievable in practice due to noise, etc. and increasing the power in one band will lead to at least some increase in the other.

2) Cursor Positioning: We apply a linear transformation to (b_1, b_2) so that the entire first quadrant is more easily accessible. This transformation is given by

$$\begin{bmatrix} x(n) \\ y(n) \end{bmatrix} = \begin{bmatrix} 1.75 & -0.75 \\ -0.75 & 1.75 \end{bmatrix} \begin{bmatrix} b_1(n) \\ b_2(n) \end{bmatrix} \tag{2}$$

where $x(n)$ and $y(n)$ are the transformed coordinates at time n. After adjusting the values to fit to the screen dimensions, they can be directly used as the cursor position coordinates to draw the cursor to the screen, but the result is quite jittery. To help smooth out the cursor's motion, we apply a moving average filter using the current point and four previous points, which has been found to provide smooth progression of cursor positions while still allowing for agile control. In addition, we linearly interpolate between the calculated cursor positions (obtained at 4 Hz) so that the perceived update rate of the cursor position is 32 Hz.

B. User Interface

The graphical user interface is largely a derivative of our previous work on two-dimensional cursor control [4]. Once the cursor position is obtained from the signal processing described above, a small dot is drawn at that location on the screen (cursor), with the origin at the lower left-hand corner, as shown in Figure 3(a). The other four BCI-specific regions on

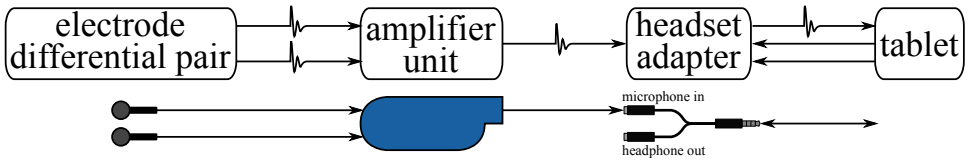
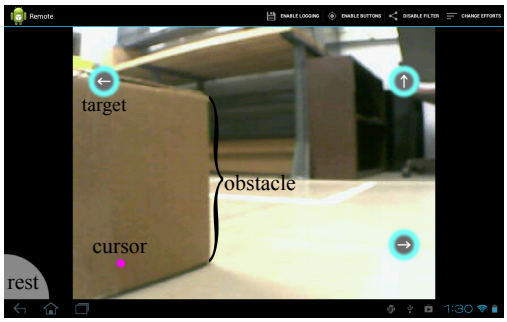
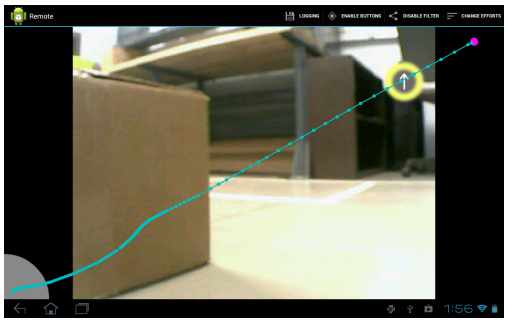


Fig. 2. Diagram of hardware connections made to input the sEMG signal to the tablet computer.



(a) Cursor on its way to a target.



(b) Forward target selected

Fig. 3. Screenshots of the user interface while the robot is navigating the course. Video from the robot's camera is shown in the background (with an obstacle in view), while the brain-computer interface regions overlay it. The cursor's path in time is obtained from logged data and is applied on top of the screenshot in (b).

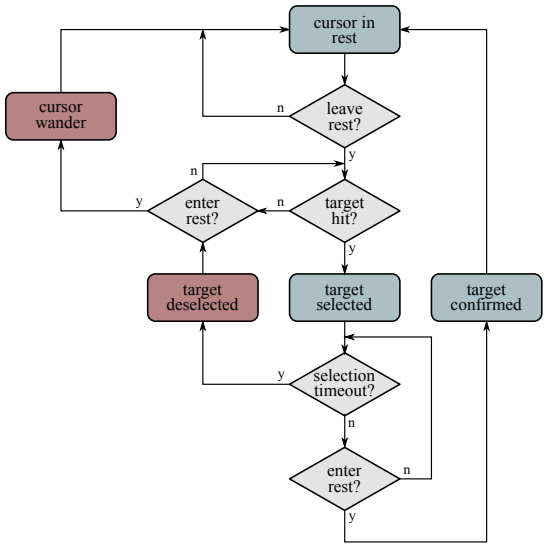


Fig. 4. A flowchart illustrating the processes involved in operating the BCI without regard to robot state. Several cycles appear including target selection–confirmation, target selection–deselection, and cursor wandering and resting.

the screen include three targets (1% of total screen area) and a rest area. Video from the robot's camera is also displayed in back of the BCI elements.

The cursor begins from the rest area shown in grey at the lower left corner (the origin). As the user's muscle contracts, the cursor moves around the screen. It is important to note that muscle movement is not required for the BCI, only electrical activity at the muscle which can be produced using a slight isometric contraction. By manipulating the contraction, the user can observe the effect this has on the cursor position. Hence, a feedback loop is formed in which the user has no knowledge of the underlying process, but he or she obtains visual information from the cursor position as well as proprioceptive feedback from the muscle itself. When the user successfully guides the cursor to one of the target areas on the screen, that target is said to be *selected*. An example of the cursor's path to a target obtained from actual logged data is shown on top of the screenshot in Figure 3(b). Note that the user does not see this path when using the BCI (only the current cursor position is shown). Each button has an arrow inside of it which indicates the direction the robot will move if that target is *confirmed*. A target is confirmed when the user relaxes so that the cursor falls back to the rest area (negligible power in both bands). At the moment when the cursor crosses the boundary into the rest area, the state of the target changes from *selected* to *confirmed* and the corresponding control message is transmitted to the robot. The process then begins again and the user can immediately start moving the cursor to another target.

To demonstrate the more subtle steps involved in operating the BCI, a flowchart is given in Figure 4. This portrays the underlying cycles that occur when controlling the BCI in practice. The best performance is achieved when the user is able to make the cursor efficiently leave rest, hit a target, then fall back to rest by relaxing the muscle, all in timely sequence. Two notable indications of inefficient use are the *cursor wander* and *target deselected* processes. Wandering occurs when the cursor leaves rest as though it is going to hit a target, but fails to do so before going back into rest. It will be shown in Section V that the accumulation of wandering time accounts for a significant portion of the time spent using the BCI. Target deselection is another inefficiency that occurs when the cursor hits a target, but does not fall back to rest within a predefined selection timeout interval (set to three seconds in our study). While this is the only way for a target to be deselected, it can happen for two reasons: either the user is unable to allow the cursor to fall back to rest quickly enough, or the user has unintentionally hit a target and does not want to confirm it. The former is easily fixed by recalibration or increasing the selection timeout interval, whereas the later

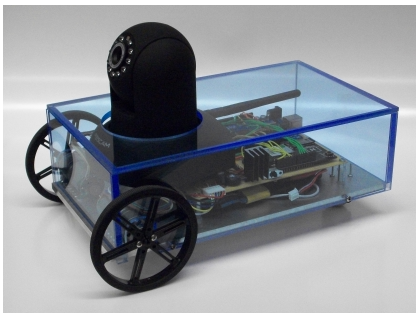


Fig. 5. The robot developed for our work. The camera head protrudes from the blue acrylic shell while all other electrical components are housed inside the shell.

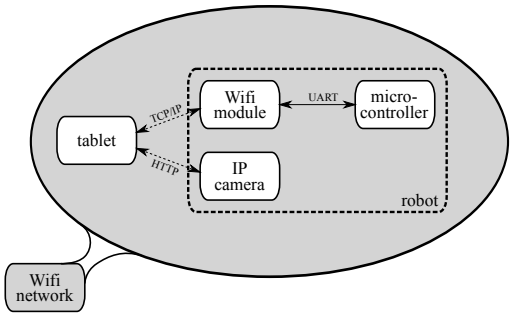


Fig. 6. Diagram of the communication architecture with which the robot receives motion commands from and transmits video to the BCI. Solid and dashed arrows denote wired and wireless connections, respectively.

indicates a more serious performance issue. Logging the cursor position data as well as the list of events (target hit, command sent, command complete, etc.) allows us to see when targets were deselected, but it does not help determine the reason behind the deselection. A full cursor path analysis can provide more information.

III. Robotic System

The robot, built in-house and pictured in Figure 5, was designed to represent a small floor-bound car that allows the user to drive through a typical household environment via smartphone or tablet computer within a local Wifi network. Approximations of some of the general characteristics of the robot are given in Table I. A pair of DC motors drive the system from the front while two ball casters provide support in the rear. This gives the robot a differential drive style, which works well with the rudimentary control commands available to the user. Although there is room in the design for sensors to allow the robot to avoid hitting obstacles or perform more high-level autonomous operations without direct control commands from the BCI, no such autonomy has been implemented to date. A single lithium polymer battery powers the robot for several hours of continuous use. A Wifi router somewhere in the area facilitates wireless communication between a set of devices in the local network, which in our system includes a tablet or smartphone (control device) and two modules on the robot. In parallel with the signal processing described in Section II, the control device sends robot motion commands and displays a video feed from the robot's perspective. These two capabilities are split into two distinct communication methodologies including motion control and video feedback, as shown in Figure 6.

TABLE I. Robot Characteristics

Dimensions	30 cm (l), 19 cm (w), 19 cm (h)
Mass	5 kg
Max. Speed	0.7 m/s
Battery life	$\sim$3 hrs (heavy driving), $\sim$6 hrs (light driving)

A. Motion Control

Motion control messages are sent to the robot via Transmission Control Protocol (TCP) to a DIY Sandbox Hydrogen Wifi module, which behaves as a device on the network. Once a control message is received, the data is then transferred over a wired Universal Asynchronous Receiver/Transmitter (UART) connection to an Arduino Mega (ATmega2560) microcontroller board. A program running inside the microcontroller receives these commands and outputs two pulse-width modulated signals to the DC motors. A PID control loop utilizes feedback from rotary encoders on each motor shaft to control the angular velocity of each motor for three different motion commands: move forward one meter, rotate to the right 15°, and rotate to the left 15°. Although more complex "continuous" style control of the robot could be implemented just as the cursor of the BCI is controlled continuously, the current study relies on our participant's known ability to hit targets in order to show that the BCI and robotic system can be integrated. Continuous robot control is a more advanced stage of research. The robot also has the ability to send messages back to the control device such as battery level and command completion confirmation. The command completion message is used to approximate the round trip time for a control message to be sent to the robot, the command performed, and the confirmation message to be received. Note that this time interval includes message transmission delay, though this is usually negligible in the local network setting as opposed to full Internet-based control.

B. Video Feedback

Video feedback is streamed to the control device by a commercially available Wifi-enabled IP camera. This camera is able to connect to a local Wifi network so that other computers and devices on that network can obtain its video images. The program running on the tablet sends HTTP GET requests to the camera's local IP address by encoding commands in a URL. In the case of the "get video" request, the camera responds with a stream in MJPEG video format. Upon receiving the video stream data, the control device decodes the images frame by frame and displays them in the user interface. The camera also provides pan and tilt control functionality as well as a microphone for audio streaming, though these features are not utilized in the current study.

IV. Case Study

To validate the BCI-robot system, we created a simple obstacle course and had a participant attempt to navigate the robot through it remotely. The participant was a 30 year old male with an incomplete C3-C4 spinal cord injury (over 15 years since injury) who has been involved in our previous work using the mobile phone version of the BCI (no robot,

only cursor control) approximately four times per month over the past nine months. An experienced participant was used to eliminate potentially confounding issues involved in learning and focus the study on the end-to-end integration of the robot system and the BCI. Characterization of the learning time to use our device is an important issue and is the subject of a current study in our laboratory using a larger population size. All testing was approved by the Institutional Review Board of the University of California, Davis.

The course, seen in Figure 7, is approximately 4.25 m long and 2 m wide. Two obstacles block the direct path to the goal area, and they are situated such that the user is required to utilize all three motion commands available (i.e. go forward, turn right, turn left) in order to complete the navigation task. One obstacle splits the course so that there is an option to go around it in two different ways. The telerobotic task is defined as sending a series of robot commands in order to get the robot from the start location to the goal location without hitting obstacles or going outside the course boundaries. Ideally, the robot would be capable of traversing the course using approximately 15 commands. However, because of encoder-based odometry errors from wheel slippage and bumps in the path, a more realistic minimum is 20 commands. This is verified by taking the BCI component out of the loop and sending motion commands directly to the robot.

The participant was allowed to see the course before beginning the trial and was instructed on how target selection and confirmation works. He then moved to a location about 10 meters from the robot site with a partition blocking the view of the course. The only feedback from the robot thereafter was the robot's camera image displayed on the control tablet as shown in Figures 1 and 3. After calibration, the participant was given some time to practice sending commands, primarily to see how the robot moves with each command. Following this, the trial began, and the data collected during the single trial is presented below, representing the first time the subject attempted to complete the course from beginning to end.

V. Results

In order to evaluate the performance of the participant using the system and the system itself, we needed to establish some metrics. Perhaps the most important metric in a system such as this is the total amount of time taken to complete the navigation task. This encompasses all aspects of the system including efficiency of BCI usage, command selection, and robot performance. If considering only robot travel time, the course could be completed in approximately one minute. However, this does not include time required to activate the interface or account for errant commands by the user. Our goal was to have the user complete the course in approximately 10 minutes, and in the trial we report, the user completed the course in 8 minutes and 30 seconds. Other numerical values of general interest are given in Table II. The number of target attempts is simply the number of times the cursor entered a target region on the screen. Time to target is defined as the amount of time measured from the moment the cursor leaves rest to the moment it hits a target. The number of commands sent represents a subset of the number of targets hit. A target being confirmed is equivalent to a command being sent. So far, no transmission failures have occurred, so this number can also be seen as equivalent to the number of tasks executed. The total number of commands sent is in the range expected, especially considering that the participant has no overhead view of the course during the trial. Command completion time represents the amount of time between the control device sending a motion control message and the robot responding with a completion message. This can be thought of as travel time, since transmission delays are normally very small. Note that forward travel time is inherently greater because of the much larger wheel rotation needed to complete the motion command. Target deselection was defined previously via Figure 4 as the cycle in which the cursor leaves rest, hits a target, then fails to return to rest within the three-second selection timeout interval, and deselection time is simply the elapsed time during the deselection process. Note that the number of target deselections is not insignificant compared to the number of commands actually sent, and to the number of target attempts.

TABLE II. Evaluation of Target Selection

Metric	Target			
	forward	right	left	all
# target attempts	20	15	12	47
avg. cursor to target time (s)	1.57	2.45	2.92	2.20
# commands sent	12	8	8	28
avg. command completion time (s)	4.09	1.45	1.64	2.64
# deselections	8	7	4	19
avg. deselection time (s)	3.69	4.68	5.38	4.41

The three main performance inefficiencies have been identified as *cursor wandering*, *cursor resting*, and *target deselection*. Cursor wandering, as previously described, is the cycle in which the cursor leaves rest, fails to hit a target, then enters rest. The amount of time for a given "wander" is defined as the time from leaving rest to entering it. The amount of time spent in rest is a weak indicator of inefficiency as there will always be some non-zero resting time due to the nature of the target confirmation process. Also, the user is expected to spend some time relaxing to avoid muscle fatigue and to plan the next command to send to the robot. It should be noted that the participant was not told to navigate the course as quickly as possible. Instead, he was encouraged and reminded to take brief respites before trying to hit the next target. Deselection time is a strong indication of performance inefficiency, however. Note that it is more than simply the number of deselections multiplied by the selection timeout value because of the time taken to reach the target from rest. Not including cursor rest, the inefficiencies noted here account for 48% of the total course navigation time. This of course depends heavily on user performance, but it may be possible to make improvements to the BCI such that cursor wandering and target deselection times are reduced.

TABLE III. Evaluation of Performance Inefficiencies

Metric	Times			
	min (s)	max (s)	mean (s)	total (min:sec)
cursor wander	0.09	9.14	2.57	2:44
cursor rest	0.11	20.08	1.52	2:40
target deselection	–	–	–	1:23

A visual depiction of the results of the trial are shown in Figure 7. This shows the positions in the course in which the robot stopped between successive commands executed during

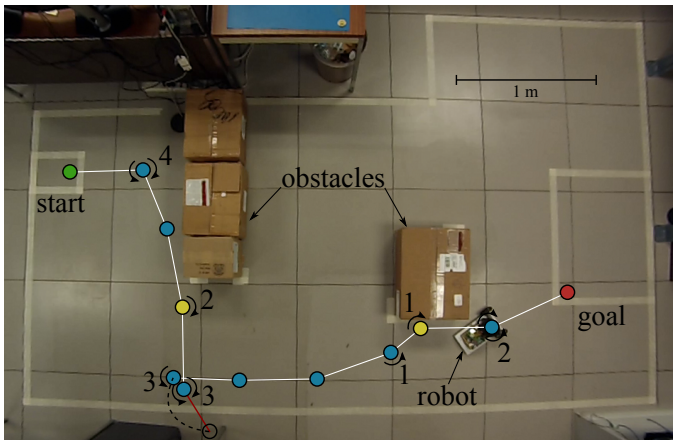


Fig. 7. A map of robot locations while completing the course. Numbers next to the points indicate that the robot turned in place at that location the corresponding number of times before moving to the next point. Point locations are estimated based on overhead video footage taken throughout the trial.

the telerobotic task. A wide-angle camera mounted above the course captured video of the robot as it travelled through the course and this was used to estimate the robot's location after each command was completed. Curved arrows next to the location points in the figure indicate the directions, if any, that the robot turned in place at that location. Numbers next to the points indicate the number of times the robot turned in place. Note that several of the points have no number, which means the robot only stopped at that location and moved forward at the very next command. This shows that only a few extraneous commands were used to complete the task. The two yellow points indicate that the robot contacted an obstacle. In both cases, it simply caused the robot to stop moving, but it did not cause the robot to become stuck at that location, so the trial continued. The red line going outside the boundary of the course is the result of the user hitting an undesired target, causing the robot to go out of bounds. To keep the trial running, the robot was reset to its previous location (indicated by the dashed line) and the trial continued. We believe that the participant would have been capable of continuing toward the goal without being reset to the previous location, but we wanted to contain the trial within the specified boundaries. If the boundary had been established with a physical object, the trial would have likely continued just as with the collisions with the obstacles. As noted previously, simple obstacle detection could easily be added to avoid such collisions, but as it stands, the results show more accurately the level of control the user had in operating the robot via the BCI. Recall also that this was the first time the user attempted to navigate the course.

VI. Conclusions

Based on the results above, there are a few notable improvements to the system that could be made to reduce inefficiencies and improve reliability and ease of use. The ability to distinguish between the two reasons for target deselection could help to reveal factors leading to undesired target selection and inability to rest before target selection timeout. Feedback from our participant indicates that the ability to pause the BCI's control of the robot would be a nice feature which would help to avoid hitting unwanted targets when taking a break. Even though nothing should occur if the cursor stays in rest, the user "resting" does not necessarily imply that their muscle is not contracting (e.g. eating). The robot itself could be improved to better deal with flooring inconsistencies. In addition, there is room in the robot design to implement basic obstacle avoidance with ultrasonic or infrared sensors. Most importantly, changes to the underlying robotic control methodology could significantly improve the system, and there are many ways to use this BCI technology with different control paradigms. Continuous robotic control, in which the muscle contraction more directly affects the position or velocity of a remote robot, is one such exciting concept. Progress toward this idea could create new applications for the BCI and unlock new telepresence communication possibilities for the disabled. Finally, once the system configuration is improved, we can begin to study how a user improves with practice and differences between users. Still, the current paper represents the first use of our new sEMG BCI for control of a telepresence robot.

Acknowledgment

We would like to thank the participant in this study for his enthusiasm and helpful feedback, I.-M. Skavhaug for help with subject testing, C. Tung, M. Marinelli and R. Furey for their work in designing and building the robot, B. Vernon for help with code, and C. Davis and M. Schirle for support in robot design. This work was funded by The Hartwell Foundation, and Grant Number UL1 RR024146 from the National Center for Research Resources (NCRR), a component of the National Institutes of Health (NIH), and NIH Roadmap for Medical Research. Its contents are solely the responsibility of the authors and do not necessarily represent the official view of NCRR or NIH. We thank the UC Davis Clinical and Translational Science Center for support.

References

[1] C. Perez-Maldonado, S. Joshi, and A. S. Wexler, "Control of partial spectral power in different frequency regions of the surface EMG signal of a single muscle," in *Proc. XVIIth Congress of the International Society of Electrophysiology and Kinesiology (ISEK 2008)*, Niagara Falls, Ontario, Canada, Jun. 2008.

[2] C. Perez-Maldonado, A. S. Wexler, and S. S. Joshi, "Two dimensional cursor-to-target control from single muscle site sEMG signals," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 18, pp. 203–209, April 2010.

[3] S. S. Joshi, A. S. Wexler, C. Perez-Maldonado, and S. Vernon, "Brain-muscle-computer interface using a single surface electromyographic signal: Initial results," presented at the Proc. IEEE/EMBS Int. Conf. Neural Engineering, Cancun, Mexico, Apr. 2011.

[4] S. Vernon and S. S. Joshi, "Brain-muscle-computer interface: Mobile-phone prototype development and testing," *IEEE Trans. Inf. Technol. Biomed.*, vol. 15, pp. 531–538, July 2011.

[5] J. R. Millán, F. Renkens, J. Mouriño, and W. Gerstner, "Noninvasive brain-actuated control of a mobile robot by human EEG," *IEEE Trans. Biomed. Eng.*, vol. 51, pp. 1026–1033, June 2004.

[6] C. Escolano, J. M. Antelis, and J. Minguez, "A telepresence mobile robot controlled with a noninvasive brain-computer interface," *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 42, pp. 793–804, June 2012.

[7] K. Xue, R. Liu, and M. Meng, "Design of internet-based teleoperation platform with brain-computer interface," in *Proc. IEEE ROBIO 2009*, Guilin, China, Dec.9–23 2009, pp. 723–728.

[8] P. Gergondet, D. Petit, and A. Kheddar, "Steering a robot with a brain-computer interface: Impact of video feedback on BCI performance," in *Proc. IEEE RO-MAN 2012*, Paris, France, Sep.9–13 2012, pp. 271–276.