A Hierarchical Architecture for Adaptive Brain-Computer Interfacing

Mike Chung[1], **Willy Cheung**[1], **Reinhold Scherer**[2] **and Rajesh P. N. Rao**[1]
[1]Computer Science & Engineering, University of Washington, Seattle, USA
[2]Institute for Knowledge Discovery, Graz University of Technology, Graz, Austria
[1]{mjyc, wllychng, rao}@cs.washington.edu, [2]{Reinhold.Scherer}@tugraz.at

Abstract

Brain-computer interfaces (BCIs) allow a user to directly control devices such as cursors and robots using brain signals. Non-invasive BCIs, e.g., those based on electroencephalographic (EEG) signals recorded from the scalp, suffer from low signal-to-noise ratio which limits the bandwidth of control. Invasive BCIs allow fine-grained control but can leave users exhausted since control is typically exerted on a moment-by-moment basis. In this paper, we address these problems by proposing a new adaptive hierarchical architecture for brain-computer interfacing. The approach allows a user to teach the BCI new skills on-the-fly; these learned skills are later invoked directly as high-level commands, relieving the user of tedious low-level control. We report results from four subjects who used a hierarchical EEG-based BCI to successfully train and control a humanoid robot in a virtual home environment. Gaussian processes were used for learning high-level commands, allowing a BCI to switch between autonomous and user-guided modes based on the current estimate of uncertainty. We also report the first instance of multi-tasking in a BCI, involving simultaneous control of two different devices by a single user. Our results suggest that hierarchical BCIs can provide a flexible and robust way of controlling complex robotic devices in real-world environments.

1 Introduction

Brain-computer interfaces (BCIs) have received considerable attention in recent years due to their novel hands-free mode of interaction with the environment [Rao and Scherer, 2010; Scherer *et al.*, 2008; Faller *et al.*, 2010]. In particular, the field has seen rapid growth due to its potential for offering a new means of control for devices tailored to severely disabled and paralyzed people: examples include directing the motion of a motorized wheelchair, controlling a semiautonomous assistive robot, and using a neuroprosthesis [Galán *et al.*, 2008; Bell *et al.*, 2008; Müller-Putz *et al.*, 2005].

The most commonly used brain signal source for non-invasive BCIs in humans is the electroencephalogram (EEG). However, due to its non-stationarity, inherent variability, and low signal-to-noise ratio, a reliable translation of EEG into appropriate control messages for devices can be difficult and slow. Therefore, EEG signals have often been used to select a task that can be semi-autonomously performed by an application (e.g., control of a humanoid robot in [Bell *et al.*, 2008]). Invasive BCIs, on the other hand, offer higher bandwidth and allow fine-grained control of robotic devices (e.g., [Velliste *et al.*, 2008]), but moment-by-moment control over long periods of time can place a high cognitive load on the user.

To overcome these problems, we propose an adaptive hierarchical architecture for brain-computer interfacing which allows the user to teach the system new and useful tasks on an ongoing basis: low-level actions are first learned and later semi-autonomously executed using a higher-level command (e.g., the command "Go to kitchen" for a semi-autonomous mobile robot). Such higher-level control frees the the user from having to engage in tedious moment-by-moment control once a command has been learned.

In addition, we introduce, to our knowledge, the first use of uncertainty for guiding a BCI's behavior during hierarchical control. We use Gaussian processes (GPs) for learning high-level commands and exploit the fact that they provide a measure of uncertainty in their output [Rasmussen, 2004]. When the uncertainty in a given region of task space is too high (e.g., due to lack of training in that part of the space), the BCI switches to user control for further guidance rather than continuing to execute the unreliable and potentially dangerous high-level command. Such uncertainty-guided decision making is critical for real-world BCI applications, such as BCI-control of a robotic wheelchair or assistive robot, where user safety and the safety of those around the robot are of paramount importance.

We present results from user studies involving four human subjects who successfully taught and controlled a humanoid assistive robot in a simulated home environment. We also report the first example of multi-tasking in a BCI, where a user was able to simultaneously control two different devices. Our results provide a proof-of-concept demonstration that hierarchical BCIs may offer a scalable and robust approach to controlling complex robotic devices in real-world environments.

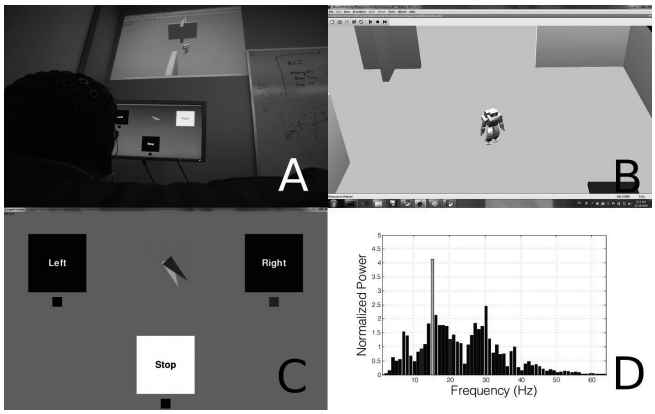

Figure 1: **A Hierarchical BCI System.** A. Experimental setup: User selects from a menu shown on a monitor while a view of the robot in its environment is shown in a larger immersive setting above, B. Simulated robot in its environment: The robot is a Fujitsu HOAP-2 humanoid simulated using the Webots software, C. A screen shot of the menu and SSVEP stimulation, D. Frequency domain representation of a subject's EEG signal illustrating a high SSVEP response to 15Hz stimulation.

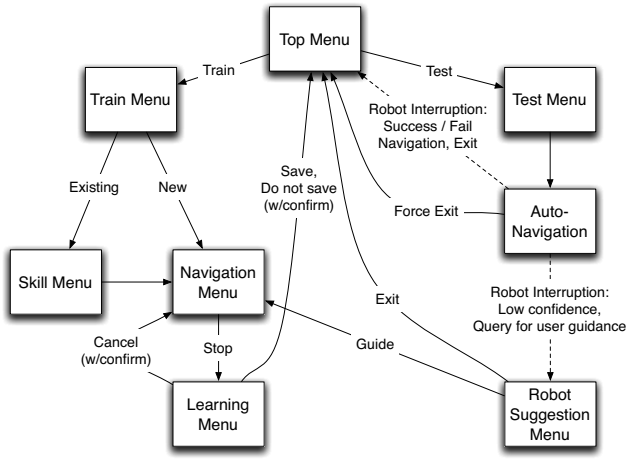

Figure 2: **Overview of control flow in the hierarchical menu system.**

2 Methods

2.1 A Hierarchical Architecture for BCI

The hierarchical BCI proposed in this paper is composed of three main components: (A) an EEG-based BCI; we used a steady state visual evoked potential (SSVEP) based BCI [Müller-Putz and Pfurtscheller, 2007], but other commonly used EEG responses such as P300 or mental imagery could also be used. We used SSVEPs because they offer relatively high information transfer rates (ITR) with minimal user training: (B) a hierarchical menu and learning system that allows the user to teach the BCI new skills, and (C) the application, which, in the present case, is a simulation of a humanoid robot in a home environment that mimics the physics of the real world (Figure 1.A and 1.B). The three components interact closely to make the system work. In particular, the hierarchical adaptive menu system displays available commands as flashing stimuli for the user to choose using the SSVEP-based BCI. The user makes the desired selection by focusing on the desired command in the menu (Figure 1.C). The BCI detects the option the user is focusing on and sends its classification output to the hierarchical menu system, which in turn sends a command to the robot and switches to the next appropriate menu. The robot executes the command it receives, which can be either a lower-level command such as turn left/right or a higher-level learned command. Finally, the user closes the control loop by observing the simulated robot's action and making the next desired selection based on the updated menu.

We describe each of the components of the hierarchical BCI system in more detail below.

SSVEP-based BCI

Flickering stimuli used to elicit SSVEPs were presented on a TFT computer screen with a refresh rate of 60 Hz. Up to three different options (12 Hz, 15 Hz, and 20 Hz) could be presented to the user in any given menu.

Continuous EEG was recorded bipolarly from gold electrodes placed at electrode positions Oz and Cz (ground was linked to Cz), notch filtered at 60 Hz and digitized at 256 Hz (gUSBamp, Guger Technologies, Graz, Austria).

To detect the flashing stimulus the user was focusing on, the power spectrum was estimated using the Fast Fourier Transform (FFT). FFT was applied to 1s segments of EEG data (Hamming window) every 0.5s and the power for each frequency was then calculated using squared values. The data used for final classification was a 4-second average of these power values (calculated from 8 FFT values). The frequency with the highest power among the three target frequencies of 12, 15, and 20Hz was classified as the user's choice for that decision period (see Figure 1.D for an example).

The BCI menu on the computer monitor and a video projection of the robot simulator were placed one above the other (Figure 1.A). When users desired BCI control, they focused on the monitor, while at other times, they watched the robot move in its environment. When users were not focusing on the BCI menu, the power in the recorded EEG channel was markedly different, allowing a simple threshold-based detector to self-initiate the SSVEP-BCI whenever the user required control.

Hierarchical Adaptive Menu

The hierarchical menu (Figure 2) is the interface the subject uses to interact with the hierarchical learning system. It displays the available commands for the hierarchical learning system, which are selected using SSVEP. The top-level menu presents two options: 'Train' and 'Test'.

Selecting 'Train' allows the user to either teach the system a new task ('new' option) or update an existing one ('existing' option). If 'new' is selected, the next menu presented is the robot navigation menu. If 'existing' is selected, the user must choose a task to update before the navigation menu is displayed (see Figure 2). In navigating the robot, the user

has three choices: left, right, and a stop option indicating the user is done with the task. To continue moving forward in the current direction, the user need not make a choice. When 'stop' is selected, a menu offers the user the option of saving the task for inclusion in the training dataset for learning the corresponding high-level command (see below). In order to mitigate the effects of erroneous classifications, the system includes various confirmation menus, giving users the ability to verify or correct their last choice.

Selecting 'Test' allows the user to select a task that was previously learned by the system. After the user has demonstrated and saved examples of a task, the BCI learns the task (using a learning algorithm such as a Gaussian process (GP); see next section) and the system incorporates this task into the hierarchical menu as a new option in the 'Test' menu. The user can now simply select the task as a high-level command, and be at ease (or perform another BCI task) while the robot autonomously performs the task.

For allowing the BCI to make decisions based on current uncertainty in the learned model, we included one additional menu. While robot is performing a selected high-level command and enters a region where the output of the learned model has high uncertainty (e.g., high variance in the case of a GP), the BCI interrupts and displays a menu with two options: 'guide' and 'exit'. The user can select the 'guide' command to guide the robot to the desired destination (thereby generating more training data), or return to the top-level menu by selecting the 'exit' option.

We also conducted a preliminary study with two new multi-tasking options added to the 'Auto-Navigation' menu (see Figure 2). While robot is executing a high-level command, the user can choose one of these two multi-tasking options, currently mapped to turning an overhead light on/off on the right and left side respectively of the simulated home environment. Note that these multi-tasking options could be mapped to any of a set of actions that could help the user achieve a desired goal faster. Such multi-tasking effectively expands the bandwidth of control through the use of a hierarchy.

Robot Application

Bell et al. demonstrated a BCI for high-level control of a Fujitsu HOAP-2 humanoid robot [Bell *et al.*, 2008] but their work involved a fixed set of high-level commands. We used the Webots simulator [Cyberbotics Ltd., 2010] to simulate the HOAP-2 robot in a simple home environment. Note that rather than representing an animation of the robot, the Webots software simulates the physical dynamics of the Fujitsu HOAP-2 robot as well as the environment; this facilitates the transition of the results to the real-world.

The simulated robot was pre-programmed with a basic set of routines to walk forward, turn right, turn left, and make smooth transitions from one motion to another motion. The robot was also programmed with a simple collision avoidance behavior to keep the robot from walking into a wall or other obstacles during navigation. Given these basic navigational routines, we developed a controller for robot navigation, with the user having a birds-eye view of the robot. The robot was always in motion unless stopped by the user or the collision avoidance behavior.

The hierarchical BCI differs from traditional BCI systems in its ability to learn new behaviors from user demonstrations. Learning occurs in the robotic component of the BCI system, and is then abstracted into the hierarchical menu system. In the current implementation, we utilized a simple position-based approach to learning to navigate in the home environment; other robotic devices such as prosthetics will require more sophisticated methods for learning new skills.

In our experiments, we tested two learning algorithms: radial basis function (RBF) networks [The MathWorks, Inc., 2010] and Gaussian processes (GP) [Rasmussen, 2004]. Using a simulated on-board GPS sensor, the robot's position data was logged at a sampling rate of 0.5hz as the user guided the robot to a desired location. When the user subsequently commands the robot to learn the demonstrated navigation skill, this training data was used to learn mapping from position data to global navigation direction. The logged data and learned model are stored locally, and the user can update a selected skill with more demonstrations as needed, improving performance over time. This arrangement also allows training over multiple days. We used off-the-shelf software packages for learning the RBF and GP models. RBF models were learned using the "newgrnn" function in Matlab's Neural Networks Toolbox. For GPs, the GPML Matlab Code package [The Gaussian Processes Web Site, 2011] was used with isotropic squared exponential covariance function and Gaussian likelihood function with standard deviation of the noise hyperparameter set to 0.1.

During execution of a high-level command, the robot queries the BCI for navigation direction based on the current position. If the GP model is used, one obtains both a predicted mean value as well as the variance of the prediction. This variance can be related to the "confidence" of the BCI in the learned model: high variance implies low confidence in the predicted navigational command and vice versa. For the current implementation, we used a simple threshold to decide when the robot should ask the user for the guidance based on this confidence metric.

3 Experiments and Results

3.1 Study I: Testing the Hierarchical Architecture

Four healthy, able-bodied individuals participated in the first set of experiments designed to test the hierarchical BCI architecture (all male, age ranging from 20-30). None of the subjects had any prior experience with the hierarchical BCI system. One subject had participated in SSVEP-based BCI experiments in the past. All subjects read and signed a consent form approved by UW Human Subjects Division.

The first set of experiments used RBF networks for learning. First, preliminary trials consisting of only the SSVEP portion were run, lasting about 10 minutes; this was done to familiarize subjects with flashing stimuli, and to allow us to perform initial analysis to characterize each subject's SSVEP response. Subjects then used the entire system to navigate the robot from its initial position (lower-left corner) to an assigned goal position (lower-right corner) using low-level commands (left/right/stop). In the test phase, they were asked to reproduce the same task but using the high-level command

Table 1: **Performance Comparison**

	Low-level BCI	Hierarchical BCI
Mean among four subjects (std)		
Num Selections	20 (7)	5 (2)
Task Time (s)	220 (67)	112 (25)
Nav Time (s)	124 (37)	73 (19)
Mean of three trials from best subject (std)		
	Low-level BCIs	Hierarchical BCIs
Num Selections	15 (5)	4 (1)
Task Time (s)	141 (42)	85 (4)
Nav Time (s)	99 (30)	74 (9)
Minimum		
	Low-level BCIs	Hierarchical BCIs
Num Selections	8	4
Task Time (s)	91	75
Nav Time (s)	59	59

learned by the hierarchical learning system. This took about 20-30 minutes on average for the subjects. We additionally conducted a more extensive experimental session with the best subject from our first set of experiments, where he was asked to perform the navigation task three times using low-level control and three times using the high-level command.

To compare performance of the hierarchical BCI to the low-level-only BCI, we employed three metrics (Table 1): cognitive load, measured by the number of commands the user had to issue to achieve a given task ('Num Selections' or number of selections); the time taken to complete the task ('Task Time'); and the time spent only on controlling the robot ('Nav Time').

3.2 Study I: Results

All four subjects were able to use the hierarchical BCI to complete the assigned tasks. The average SSVEP-based 3-class accuracy for the four subjects from the preliminary set of trials was 77.5% (standard deviation 13.8). Although somewhat lower than other SSVEP rates reported in the literature, we found that subjects exhibited higher SSVEP accuracy when using the entire system with closed-loop feedback. Results obtained for the three different performance metrics are shown in Table 1. In the table, we also include for comparison purposes the minimum values for these metrics, assuming a user with 100% SSVEP accuracy.

The results indicate that for all three metrics, subjects demonstrate improved performance using the hierarchical BCI: both the mean and variance for all three performance metrics are lower when using the hierarchical BCI compared to the low-level BCI. Results from the best performing subject provide interesting insights regarding the use of high-level commands in a hierarchical BCI. Due to the high SSVEP accuracy of this subject, the difference in the mean values between low-level and hierarchical modes of control was less, but the variance for low-level control was significantly greater than for higher-level control (Table 1). This is corroborated by the navigational traces in Figure 3, where we see that trajectories from the hierarchical BCI tend to follow the minimal path to the goal location based on the learned

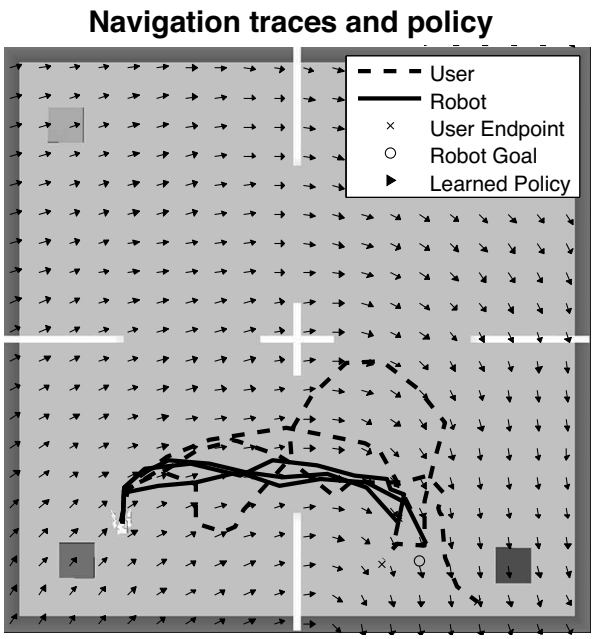

Figure 3: **Example Robot Trajectories from User-Demonstrated Low-Level Control and Hierarchical Control**. The dashed trajectories represent low-level navigational control by the user. These trajectories were used to train an RBF neural network. The solid trajectories represent autonomous navigation by the robot using the learned RBF network after selection of the corresponding high-level command by the user. The small arrows indicate the vector field learned by the RBF network ('Learned Policy') based on the user's demonstrated trajectories.

representation in the neural network. This result confirms the expectation that the network learns an interpolated trajectory that minimizes the variances inherent in the training trajectories, with more training data leading to better performance.

3.3 Study II: Uncertainty-Guided Actions and Multi-Tasking

An important observation from Study I was that the learned high-level commands were not reliable in parts of the task space where there is insufficient training data. Ideally, we would like the system to identify if it is able to safely execute the desired high-level command, preventing potentially catastrophic accidents. We investigated such an approach in Study II by utilizing Gaussian processes (GP) for learning instead of RBF networks.

The experiments were conducted with the subject that performed best in Study I. The navigation task was similar but used a room that was 2.25 times larger and had various obstacles. The enlarged size and presence of non-wall shaped obstacles increased the difficulty of robot navigation by requiring longer and more focused control. The environment had two overhead lights on the right and left side of room that could be controlled in the multi-tasking task. Additionally, Study II also varied the starting position of the robot, making

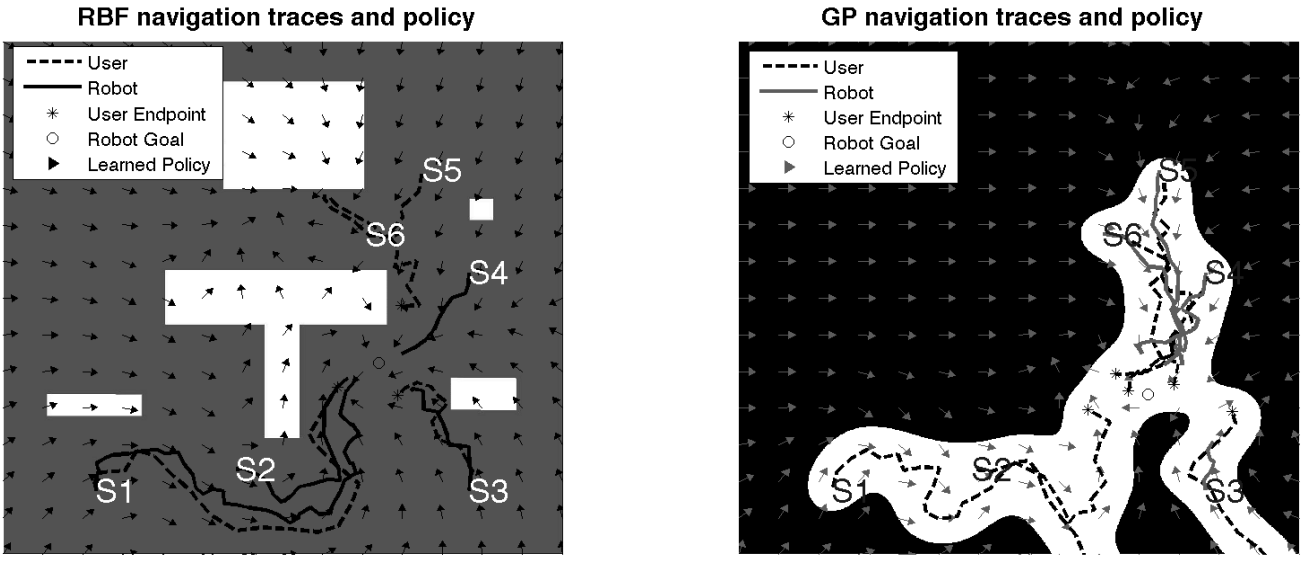

Figure 4: **Navigation traces comparing RBF and GP models for learning**. The white region in the GP plot represents the high confidence region where autonomous navigation is allowed; user input is solicited whenever the robot enters a high uncertainty (dark) region where there was insufficient training data.

the learning problem more challenging.

There were four days of experiments; two days of RBF runs on the new environment, and two days of GP runs on the new environment. On the first day for each type, the user was instructed to alternate runs of training and testing. In Figure 4, starting points S2, S4, S6 represent test starting locations, and S1, S3, S5 represent starting points of the robot in training mode. The second day only involved test trials from each of the six starting locations based on the first day's learned model. Additionally, for GP runs, to test the ability to multi-task, the user was instructed to turn on the lights on the side of the environment where the goal of the high-level command was located once the robot started autonomously navigating.

We measured two performance metrics (Figure 5): time spent controlling the robot using low-level control versus high-level commands ('Navigation time'), and number of selections the user had to make to achieve a given task ('Number of selections'). To compare the performance of GP to RBF learning, we measured the success rate of the high-level commands, defined by number of times a high-level command was successfully executed (i.e., the robot reached the destination) divided by number of times a high-level command was selected. Note that lack of success implies that the robot experienced a fall or another mode of failure.

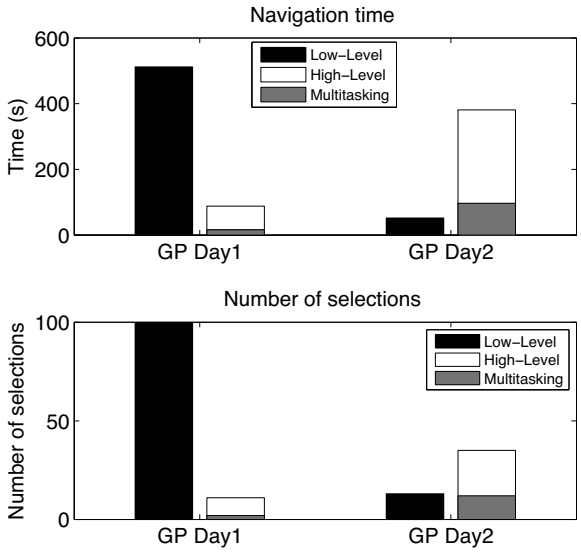

Figure 5: **Performance of the GP-based Hierarchical BCI over 2 Days.** Note the larger amount of time spent in the hierarchical mode and the greater number of high-level commands issued on day 2, indicating increased autonomy. This increased autonomy allows a greater degree of multi-tasking, as seen in the plot.

3.4 Study II: Results

The user successfully performed the entire experiment as instructed, managing a total of 24 runs over four days. As shown in Figure 5, the GP-based hierarchical BCI resorted to frequent user guidance on Day 1 (large amount of time and selections for low-level). On Day 2, however, the user was able to invoke a learned high-level command, resulting in a larger number of selections and large amount of time for

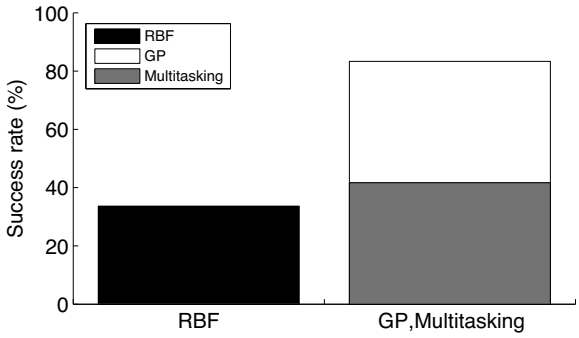

Figure 6: **Success Rate of High-Level Commands.** As expected, the GP-based hierarchical BCI has a much higher success rate due to its ability to recognize highly uncertain regions and obtain training data for these regions on an as needed basis.

high-level commands. This allowed the user to multi-task and select the appropriate light to turn on, while the robot was autonomously navigating ("Multitasking"). Figure 6 compares the success rate of high-level commands for GP versus RBF-based hierarchical BCIs. As expected, the GP-based BCI exhibits a higher success rate for performing high-level commands due to its ability to switch to user-control in low-confidence areas.

4 Summary and Conclusion

BCIs for robotic control have in the past faced a trade-off between cognitive load and flexibility. More robotic autonomy [Bell *et al.*, 2008] implied coarse-grained control and less flexibility, while fine-grained control provided greater flexibility but higher cognitive load. This paper proposes a new hierarchical architecture for BCIs that overcomes this trade-off by combining the advantages of these two approaches.

Our results from two studies using EEG-based hierarchical BCIs demonstrate that (1) users can use the hierarchical BCI to train a robot in a simulated environment, allowing learned skills to be translated to high-level commands, (2) the problem of day-to-day variability in BCI performance can be alleviated by storing user-taught skills in a learned model for long-term use, allowing the learned skill to be selected as a high-level command and executed consistently from day to day, (3) a probabilistic model for learning (e.g., GPs) can be used to mediate the switch between high-level autonomous control and low-level user control, safeguarding against potentially catastrophic accidents, and (4) the hierarchical architecture allows the user to simultaneously control multiple devices, opening the door to multi-tasking BCIs. Our ongoing efforts are focused on testing the approach with a larger number of subjects and investigating its applicability to other challenging problems such as controlling a robotic arm with grasping capabilities.

Acknowledgments

This work was supported by the National Science Foundation (0622252 & 0930908), the Office of Naval Research (ONR), and the ICT Collaborative Project BrainAble (247447). We thank Rawichote Chalodhorn for helping with the HOAP-2 robot and Webots programming aspects of the project, and Josef Faller for helping with the implementation of the SSVEP stimuli.

References

[Bell *et al.*, 2008] C.J. Bell, P. Shenoy, R. Chalodhorn, and R.P.N. Rao. Control of a humanoid robot by a noninvasive brain–computer interface in humans. *Journal of Neural Engineering*, 5:214, 2008.

[Cyberbotics Ltd., 2010] Webots. `http://www.cyberbotics.com/`, 2010. [Online; accessed 12-13-2010].

[Faller *et al.*, 2010] J. Faller, G. Müller-Putz, D. Schmalstieg, and G. Pfurtscheller. An application framework for controlling an avatar in a desktop-based virtual environment via a software ssvep brain-computer interface. *Presence: Teleoperators and Virtual Environments*, 19(1):25–34, 2010.

[Galán *et al.*, 2008] F. Galán, M. Nuttin, E. Lew, P. Ferrez, G. Vanacker, J. Philips, and J. del R. Millán. A brain-actuated wheelchair: Asynchronous and non-invasive brain-computer interfaces for continuous control of robots. *Clinical Neurophysiology*, 119(9):2159–2169, 2008.

[Müller-Putz and Pfurtscheller, 2007] G. R. Müller-Putz and G. Pfurtscheller. Control of an electrical prosthesis with an SSVEP-based BCI. *Biomedical Engineering, IEEE Transactions on*, 55(1):361–364, 2007.

[Müller-Putz *et al.*, 2005] G. R. Müller-Putz, R. Scherer, G. Pfurtscheller, and R. Rupp. EEG-based neuroprosthesis control: a step towards clinical practice. *Neuroscience Letters*, 382:169–174, 2005.

[Rao and Scherer, 2010] R.P.N. Rao and R. Scherer. Brain-computer interfacing. *IEEE Signal Processing Magazine*, 27(4):152–150, July 2010.

[Rasmussen, 2004] C.E. Rasmussen. Gaussian processes in machine learning. *Advanced Lectures on Machine Learning*, pages 63–71, 2004.

[Scherer *et al.*, 2008] R. Scherer, F. Lee, A. Schlogl, R. Leeb, H. Bischof, and G. Pfurtscheller. Toward self-paced brain–computer communication: Navigation through virtual worlds. *Biomedical Engineering, IEEE Transactions on*, 55(2):675–682, 2008.

[The Gaussian Processes Web Site, 2011] Gpml matlab code version 3.1. `http://www.gaussianprocess.org/gpml/code/matlab/doc/`, 2011. [Online; accessed 01-24-2011].

[The MathWorks, Inc., 2010] Matlab newgrnn. `http://www.mathworks.com/help/toolbox/nnet/newgrnn.html`, 2010. [Online; accessed 12-13-2010].

[Velliste *et al.*, 2008] M. Velliste, S. Perel, M.C. Spalding, A.S. Whitford, and A.B. Schwartz. Cortical control of a prosthetic arm for self-feeding. *Nature*, 453(7198):1098–1101, 2008.