A POMDP-Based Optimal Control of P300-Based Brain-Computer Interfaces

Jaeyoung Park and **Kee-Eung Kim**
Department of Computer Science
Korea Advanced Institute of Science and Technology
Daejeon 305-701, Korea

Yoon-Kyu Song
Department of Nano Science and Technology
Seoul National University
Seoul 151-742, Korea

Abstract

Most of the previous work on brain-computer interfaces (BCIs) exploiting the P300 in electroencephalography (EEG) has focused on low-level signal processing algorithms such as feature extraction and classification methods. Although a significant improvement has been made in the past, the accuracy of detecting P300 is limited by the inherently low signal-to-noise ratio in EEGs. In this paper, we present a systematic approach to optimize the interface using partially observable Markov decision processes (POMDPs). Through experiments involving human subjects, we show the P300 speller system that is optimized using the POMDP achieves a significant performance improvement in terms of the communication bandwidth in the interaction.

Introduction

Brain-computer interfaces (BCIs) interpret the electrical activities of the neurons in the brain and convey the corresponding messages or commands to the external systems (Wolpaw et al. 2002). There are a variety of devices and methods for BCIs but non-invasive methods such as electroencephalography (EEG) recorded from the scalp are suitable for practical BCIs due to user safety, ease of use and the low set-up cost. Hence, we only concern ourselves with non-invasive BCIs using EEGs. However, due to the low signal-to-noise ratio in EEGs and hence the low communication bandwidth, they are yet to be widely adopted in practice.

One of the most reliable and popular signal features in EEGs used for constructing BCIs is the P300 component in the event related potential (ERP). The ERP is elicited by an infrequent or particulary significant somatosensory stimulus. P300 is a positive peak component in the ERP at about 300ms after the stimulus (Farwell and Donchin 1988; Wolpaw et al. 2002). The P300 speller (Farwell and Donchin 1988) for the purpose of emulating a keyboard is perhaps one of the most well known BCIs based on P300. In a typical setting of the P300 speller system, the user faces the 6×6 matrix where each entry contains one letter (see Figure 1). To input a letter, the user gazes at the letter that they want to input, and then all the letters in a row or a column are flashed (stimulated) together in a random order. If the gazed letter is flashed, the P300 is elicited with some probability at approximately 300ms after the flash. Thus, using a classifier trained for detecting the P300, we can identify the target letter that the user is gazing, i.e. wants to input. However, since there is a significant chance of classification error, the rows and columns are flashed and classified multiple times and the results are combined. The P300 speller system uses a visual flash as the stimulus, but other type of stimuli (e.g. sound) can be used for constructing P300-based BCIs.

Most of the previous work on P300-based BCIs has focused on improving the accuracy of feature extraction and classification for high performance, while using a simple stimulus control of randomly ordering the stimuli. In addition, the stopping condition for the stimulus control is often specified in an ad-hoc manner. For example, conventional P300 speller systems flashes every row and column in a random order for a prescribed number of times and combine the classification results to select a target letter. Instead, we could search for a stimulus control that reduces the number of flashes and stops with a target letter selection when additional flashes are unnecessary[1].

This paper presents a systematic approach using the partially observable Markov decision process (POMDP) to obtain an optimal stimulus control. We describe how the P300 speller system can be modeled as a POMDP and show that it can significantly improve the performance of the system.

Compared to our previous work (Park, Kim, and Jo 2010) in which we had 4 and 6 targets on the screen, we have extended the approach to the full P300 speller system with 36 targets, and leveraged the regularity in the targets in the POMDP framework for further performance improvement.

P300 Speller System Architecture

This section describes the components used in the typical architecture for P300-based BCIs. Our P300 speller implementation uses the same architecture except for the system controller component, which we cover in the later section.

[1] Hill et al. (2009) were one of the first to propose improving the stimulus control, but they search for an open-loop controller with a fixed number of flashes.

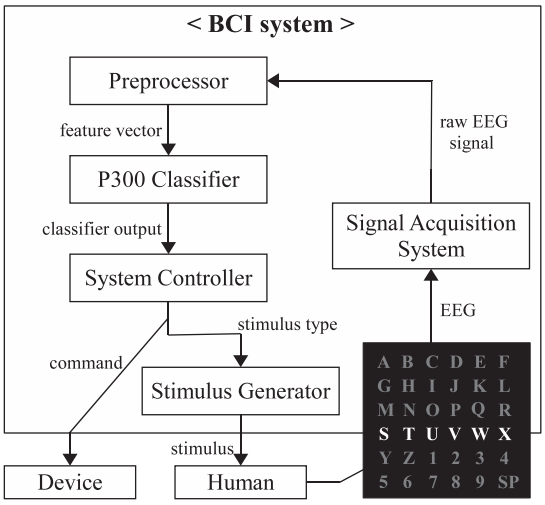

Figure 1: The typical architecture of the P300-based BCI systems. Our focus is concentrated on the system controller component in improving the performance of the system. A P300 speller matrix is depicted on the lower right corner.

Flash timing scheme

The timing of the flashes in the system follows the typical scheme used in P300 speller systems: The user gazes at a target letter that the user wants to input and all the letters in a row and a column of the 6×6 matrix are flashed at the same time, thus 6 letters flashed at a time. The flash interval is 250ms, turning on the letters for 125ms and then turning off for the remaining 125ms. We refer to a *trial* as the sequence of flashes until the system selects a target letter. A pause interval of 2.5s is given between the trials. Each trial is composed of a *row trial* immediately followed by a *column trial*. As can be inferred from their names, the row trial consists of the sequence of flashes for identifying the row that contains the target letter, and vice versa for the column trial. The target letter is determined by the row and column selected in the row and column trials.

Data acquisition, preprocessing, and P300 classification

EEG signals were acquired by the Biopac MP 150 data acquisition system using 16 channels with 1kHz sampling rate. We refer to an *epoch* of EEG signals corresponding to a flash as the windowed signal data from 16 channels between 200ms and 450ms after the flash is given. This is because P300 is expected to appear approximately at 300ms after the stimulus.

The epoch data is then passed into the preprocessor to extract relevant features, and then passed to the classifier to detect the existence of P300. In order to construct the preprocessor and the classifier, we first collected the training data consisting of epoch instances each labeled either a target (i.e. an epoch of EEG signals after the flash on the target letter) or a non-target.

As for the preprocessor, the raw epoch data were band-pass filtered (0.5-30Hz) and down-sampled to 100Hz. We then extracted features from the training data using the spatial projection algorithm (Hoffmann, Vesin, and Ebrahimi 2006). We limited the maximum number of filters to 5 for timely processing of data.

As for the classifier, we trained a support vector machine on the preprocessed feature vectors using the LIBLINEAR package (Fan et al. 2008). In order to obtain the likelihood rather than the binary classification result, we used the L2-regularized logistic regression.

Partially Observable Markov Decision Processes (POMDPs)

A POMDP (Kaelbling, Littman, and Cassandra 1998) is a mathematical model for sequential decision making problems under uncertainty in the observation. It is defined by an 8-tuple $\langle S, A, Z, b_0, T, O, R, \gamma \rangle$: S is the set of environment states; A is the set of actions; Z is the set of observations; b_0 is the initial belief state where $b_0(s)$ denotes the probability that the initial state is s; $T^a_{s,s'}$ is the transition probability of changing the state from s to s' by executing action a; $O^a_{z,s'}$ is the observation probability of observing z when changing to state s' upon executing action a; R^a_s is the reward in state s by executing action a.

The belief state b in POMDP is defined to be the probability distribution over the current states. An optimal policy π^* for the given POMDP model is the policy $\pi : b \mapsto a$ that maximizes the expected return (i.e. value), $V^\pi(b) = E[\sum_{t=0}^{\infty} \gamma^t R^{a_t}_{s_t} | b, \pi]$, for all possible belief states.

Stimulus Control Methods

We describe three stimulus control methods used in the system controller. The first one, π_{random}, is the conventional stimulus control method used in practice. The others, π_{pomdp} and $\pi_{\text{bigram-pomdp}}$, are the POMDP-based stimulus control methods we propose in this paper.

Random control method: π_{random}

This conventional control method uses a random sequence of flashes in a trial. Every row and column is flashed a prescribed equal number of times. The row and column with the maximum sum of the likelihood prediction values from the classifier is selected as the target.

POMDP control method: π_{pomdp}

This control method is an optimal policy computed from the POMDP modeling the P300 speller. Our approach constructs two POMDPs, one for the row trials and the other for the column trials, which are almost identical. Below, we describe the POMDP modeling for the row trials.

Suppose we have N=6 rows in the matrix. The states in the POMDP correspond to potential target rows, hence there are a total of N states. For each row in matrix, we can either flash it in the hope of eliciting P300 (N flash actions) or claim that it's the target row (N select actions), hence there are a total of $2N$ actions. The output value from the P300 classifier serves as the observation, where the real value between 0 and 1 was discretized into intervals of size 0.1 (e.g. z_1 for the output value in $[0.0, 0.1)$, z_2 for the output value

in $[0.1, 0.2)$, etc.), hence there are a total of 10 observations. Initial belief state is given as the uniform distribution and γ is set to 0.99.

To make the system identify the target row as soon as possible, we give -1 reward for the flash actions. On the other hand, to make the system identify the target row as accurately as possible, we give +10 for the select actions that correctly select the target row, and -100 for the select actions that incorrectly select the target row.

Transition probabilities for the flash actions are given as the identity matrix assuming that the target does not change within the trial. Those for the select actions are given as the uniform distribution assuming that the target will randomly change to a new one with equal probability.

The observation probabilities for the flash actions are estimated by constructing the histogram of classifier output values for each human subject from the training data. Although simple, this approach may overfit the true observation probabilities, hence we mix (i.e. smooth out) the estimated observation probabilities $\hat{O}^a_{z,s}$ with the uniform distribution so that $O^a_{z,s} = (1-q_o)\hat{O}^a_{z,s} + q_o(1/N)$, where q_o is the mixing weight. In the experiments, we set q_o to 0.3 for for all subjects. The observation probabilities for the select actions are given as the uniform distribution since the classifier output value does not contain any useful information.

POMDP control method with bigram: $\pi_{\text{bigram-pomdp}}$

This control method is basically the same as π_{pomdp}, except that the POMDP model uses the bigram prediction of the target letters. Using the online text archive of novels, we construct the bigram model of letters, and use the model to compute the initial belief state for the current target row and column given the previously selected target letter. In contrast, π_{pomdp} starts from the uniform distribution.

Solving POMDP Model for P300 Speller

There are two constraints in computing an optimal policy for the POMDP modeling the P300 speller system. First, there is an *observation delay* for the flash actions since P300 appears 300ms after the flash. The epoch window starts at 200ms and ends at 450ms, and a small amount of additional delay incurred by data processing and classification makes the epoch data available shortly after 500ms. Since the flash intervals are of 250ms, this implies that the observations are delayed for 2 time steps. Second, there is a phenomenon known as *repetition blindness* (Fazel-Rezai 2007) which refers to the situation in which P300 may not be elicited when a stimulus on the target is given again within 500ms.

We re-define the optimal value function to incorporate the constraints above:

$$
\begin{aligned}
&V^*(b, \langle a_{t-d}, \ldots, a_{t-1}\rangle) \\
&= \max_{a_t \in A - A'} \textstyle\sum_{s_{t-d},\ldots,s_t} b(s_{t-d}) \prod_{i=t-d}^{t-1} T^{a_i}_{s_i,s_{i+1}} R^{a_t}_{s_t} \\
&\quad + \gamma \textstyle\sum_z Pr(z|b, a_{t-d}) V^*(b^z_{a_{t-d}}, \langle a_{t-d+1}, \ldots, a_t\rangle),
\end{aligned}
$$

where d=2 is the number of time steps in the observation delay, and A' is the set of actions executed within 500ms. $b^z_{a_{t-d}}$ is the successor belief state of b after executing action a_{t-d} and observing z. Note that the value function depends on the belief state as well as the actions that were executed within the delay window.

We extended the point-based value iteration (Pineau, Gordon, and Thrun 2006) to compute the optimal value function and the optimal policy according to above definition. The observation delay makes the dimension of the value function much larger than the original value function. In other words, we have to compute the optimal value function for every delayed action sequence $\langle a_{t-d}, \ldots, a_{t-1}\rangle \in |A|^d$. However, noting that our POMDP model is symmetric (Doshi and Roy 2008; Kim 2008), and we adopted the technique in permutable POMDPs and extended it to handle observation delays. Our POMDP algorithm implemented in Java solves the model within a few minutes.

Experiments

Experimental setting

10 able-bodied students (7 male and 3 female students) at Korea Advanced Institute of Science and Technology (KAIST) participated in the experiment. The experiment was approved by the KAIST Institutional Review Board (approval # KH2010-24).

Each subject went through 4 sessions in the experiment: training, random, pomdp, and bigram-pomdp. In the training session, we collected the training data from 20 trials using the random control method π_{random}. Each trial assigned a random target letter. The trials used 10 flashes for each row and column, hence there were a total of 120 row and column flashes per trial. Once we collected the training data, we obtained the preprocessor and P300 classifier. The observation probabilities in the POMDP were also computed for each subject by constructing the histogram of classifier output values. Thus, using the control methods π_{random}, π_{pomdp}, and $\pi_{\text{bigram-pomdp}}$ obtained from the training data, we asked the subject to input "MACHINE LEARNING" in each of the remaining sessions.

Subjects A, C, E, G, and I went through the sessions in the order of training, pomdp, bigram and random. On the other hand, subjects B, D, F, H, and J went through the sessions in the order of training, random, pomdp, and bigram. We used two different setups because the random session takes significantly more time than the pomdp or bigram sessions, and we wanted to see whether the subject fatigue affected performance.

We measured the performance in terms of accuracy and practical bit rate. The accuracy is defined by the number of correctly claimed letters over the number of total inputted letters. The practical bit rate (Townsend et al. 2010) measures the quantity of the transferred information per unit time, while taking into account the additional time required to correct the error in the trials.

Results

Table 1 shows the performance of the π_{random}, π_{pomdp}, and $\pi_{\text{bigram-pomdp}}$. Since π_{random} does not have a mechanism to determine when to stop flashing, we report the performance

Subject	π_{random}			π_{pomdp}			$\pi_{\text{bigram-pomdp}}$		
	PBR (bits/min)	Acc. (%)	#flashes /target	PBR (bits/min)	Acc. (%)	#flashes /target	PBR (bits/min)	Acc. (%)	#flashes /target
A	36.483	100.00	24	38.965	100.00	21.9 ± 4.6	41.751	100.00	19.8 ± 6.1
B	9.541	100.00	120	26.452	100.00	36.9 ± 13.8	31.437	100.00	29.5 ± 7.9
C	4.051	81.25	120	6.718	87.50	96.8 ± 26.3	10.246	100.00	111.1 ± 54.3
D	0.447	56.25	120	5.736	81.25	81.9 ± 24.7	12.912	100.00	86.1 ± 72.4
E	6.765	87.50	96	6.136	87.50	106.9 ± 51.0	8.201	93.75	105.6 ± 70.0
F	17.720	100.00	60	25.833	100.00	38.1 ± 9.6	32.359	100.00	28.4 ± 5.8
G	1.057	62.50	120	2.116	75.00	164.5 ± 108.1	2.551	81.25	196.6 ± 68.0
H	9.080	81.25	48	12.380	93.75	66.6 ± 17.4	13.713	93.75	59.1 ± 22.7
I	16.331	87.50	48	27.212	100.00	35.6 ± 12.8	30.851	100.00	30.3 ± 14.7
J	20.592	93.75	36	38.288	100.00	22.4 ± 4.8	43.873	100.00	18.3 ± 7.5
Avg.	12.206	85.625	-	18.984	92.5	-	22.790	96.875	-
s.d.	10.968	15.604	-	13.990	0.092	-	14.889	0.061	-

Table 1: Experimental results of the three control methods. PBR stands for practical bit rate.

when the practical bit rate hits its maximum value. The π_{random} control used a maximum of 10 flashes for each row and column, hence 120 is the maximum number of flashes per target.

A one-way within-subject ANOVA test on the practical bit rate was conducted; Mauchly's test indicated that the sphericity assumption had been violated (p-value $\approx 0.000 < 0.05$), thus the degree of freedom was corrected according to the Greenhouse-Geisser estimates of the sphericity ($\epsilon = 0.537$). There was a significant difference among the three control methods ($F(1.073, 9.659) = 15.601$, p-value $= 0.003 < 0.05$). A post-hoc test using Bonferroni indicated that the π_{pomdp} was significantly better than the π_{random} (p-value $= 0.027 < 0.05$). Also, $\pi_{\text{bigram-pomdp}}$was again significantly better than π_{pomdp} (p-value $= 0.001 < 0.05$) as well as π_{random} (p-value $= 0.007 < 0.05$).

In terms of the average practical bit rate, π_{pomdp} achieved a 55% performance improvement over π_{random}, and $\pi_{\text{bigram-pomdp}}$ achieved a 86% performance improvement over π_{random} and 20% over π_{pomdp}.

Conclusion

We have present POMDP-based stimulus control methods in the P300 speller system that significantly outperform the conventional random control method. POMDP-based control methods achieve a high performance in the communication bandwidth by stimulating the rows and columns that have high probability containing the target letter. In addition, POMDP-based control methods determine when to stop and claim the target letter. In contrast, the conventional random control method simply stimulate every row and column in a random order, and the stopping condition is typically implemented ad-hoc by the system expert, e.g., stimulating for a prescribed number of times.

Although we have described the POMDP approach to the P300 speller system using visual stimulus, our work is general enough to be applied to BCIs with other stimulus paradigms such as auditory, olfactory, or even multi-modal stimulus. It would be also possible to apply the approach to newer state-of-art equipments such as fMRI or near-infrared spectroscopy (NIRS).

Acknowledgments

The authors would like to thank Sungho Jo for providing the EEG equipment and helpful comments for the early version of the work. This work was supported by National Research Foundation of Korea (grant # 2009-0069702) and KAIST-Microsoft Research Collaboration Center.

References

Doshi, F., and Roy, N. 2008. The permutable POMDP: fast solutions to POMDPs for preference elicitation. In *Proc. of AAMAS*.

Fan, R.-E.; Chang, K.-W.; Hsieh, C.-J.; Wang, X.-R.; and Lin, C.-J. 2008. LIBLINEAR: A library for large linear classification. *J. of Machine Learning Research* 9.

Farwell, L. A., and Donchin, E. 1988. Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials. *Electroencephalography and Clinical Neurophysiology* 70.

Fazel-Rezai, R. 2007. Human error in P300 speller paradigm for brain-computer interface. In *Proc. of IEEE EMBS*.

Hill, J.; Farquhar, J.; Martens, S.; Bießmann, F.; and Schölkopf, B. 2009. Effects of stimulus type and of error-correcting code design on BCI speller performance. In *Proc. of NIPS*.

Hoffmann, U.; Vesin, J. M.; and Ebrahimi, T. 2006. Spatial filters for the classification of event-related potentials. In *Proc. of ESANN*.

Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. R. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101.

Kim, K.-E. 2008. Exploiting symmetries in POMDPs for point-based algorithms. In *Proc. of AAAI*.

Park, J.; Kim, K.-E.; and Jo, S. 2010. A POMDP approach to P300-based brain-computer interfaces. In *Proc. of ACM IUI*.

Pineau, J.; Gordon, G.; and Thrun, S. 2006. Anytime point-based approximations for large POMDPs. *J. of Artificial Intelligence Research* 27.

Townsend, G.; LaPallo, B. K.; Boulay, C. B.; Krusienski, D. J.; Frye, G. E.; Hauser, C. K.; Schwartz, N. E.; Vaughan, T. M.; Wolpaw, J. R.; and Sellers, E. W. 2010. A novel P300-based brain-computer interface stimulus presentation paradigm: Moving beyond rows and columns. *Clinical Neurophysiology* 121.

Wolpaw, J. R.; Birbaumer, N.; McFarland, D. J.; Pfurtscheller, G.; and Vaughan, T. M. 2002. Brain-computer interfaces for communication and control. *Clinical neurophysiology* 113.