2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Remote Teleoperation of an Unmanned Aircraft with a Brain-Machine Interface: Theory and Preliminary Results

Abdullah Akce, Miles Johnson, and Timothy Bretl

Abstract— **This paper presents an interface that allows a human pilot to remotely teleoperate an unmanned aircraft flying at a fixed altitude with input only from an electroencephalograph (EEG), which is used in this case to distinguish between left- and right-hand motor imagery in the brain. The approach is to construct an ordered symbolic language for smooth planar curves and to use these curves as desired paths for the aircraft. The underlying problem is then to design a communication protocol by which the pilot can, with vanishing error probability, specify a string in this language using a sequence of bits sent through a binary symmetric channel in the presence of noiseless feedback. Such a protocol is provided by the combination of arithmetic coding as a method of lossless data compression with posterior matching as a capacity-achieving channel code. Preliminary hardware experiments demonstrate the feasibility of this approach.**

I. INTRODUCTION

A brain-machine interface is a feedback interconnection between a human user and a prosthetic device that mimics the interaction of the central nervous system (including the brain and the spinal cord) with peripheral biomechanics (including the musculoskeletal system). Measurements of neural activity take noisy signals from the user to the prosthetic; sensory receptors take noisy signals back from the prosthetic to the user. The measurements may come from non-invasive methods like an electroencephalograph (EEG) that can observe the gross electrical activity of many neurons, or they may come from invasive methods like intracortical sensors that can observe ensemble spiking of individual neurons. The feedback may be provided by a graphical display, by observation of the prosthetic (visual or through physical coupling), or by direct cortical stimulation.

There has been a sharp increase in work on brain-machine interfaces over the past decade, leading to incredible results like enabling primates to move robotic arms and tetraplegic human patients to move a cursor and read email (e.g., [1], [2], [3], [4], [5], [6], [7], [8], [9], and many others). Most of this work has originated within the neuroscience community and has focused on enriching the measurement and interpretation of neural activity, in particular on developing a better understanding of how ensembles of neurons encode information and control sensorimotor function. Much less work has focused on improving the prosthetic device (e.g., the robot arm or the cursor) and in particular on the importance of its control architecture. What is the appropriate level for task specification? What is the optimal dynamic response in the presence of uncertainty (in this case, largely about the task)? What type of sensory feedback should be provided to the user? These choices have a significant impact on the overall performance of a brain-machine interface, and consequently present an exciting opportunity for roboticists, since they are much the same choices we make when engineering other robotic systems.

We have drawn considerable inspiration from the recent success of Rebsamen et al. [7] and Iturrate et al. [8] in developing brain-machine interfaces that enable a human user with impaired sensorimotor function to drive a robotic wheelchair indoors on level ground using EEG signals. Both of these interfaces are based on the P300 evoked potential, which is an involuntary response to visual stimuli that can be measured reliably with EEG and that indicate, for example, preference for a particular item in a menu (e.g., "turn left," "turn right", or "stop") or for a particular location on a map. These high-level commands are carried out by a fully automated navigation system that handles details like localization and mapping, motion planning, obstacle avoidance, etc. Both systems show impressive levels of performance, which has been validated by experimental results. However, both systems are also complex enough that it is difficult to understand which design choices were most critical to performance, and consequently it seems difficult to generalize their results. In particular, we are interested in enabling the control of vehicles and other prosthetic devices that move at high speed and that have complex dynamics—can the same interface be used for these systems?

Motivated by this question, we take a different approach to the design of brain-machine interfaces. We view the interface as the means by which a user may communicate their intent (i.e., specify a desired task) to the prosthetic. Based on this view, we make system design choices that allow us to formally model the task as a string in an ordered symbolic language, the neural sensor (e.g., EEG) as a communication channel, and the prosthetic as a means of providing feedback to this channel. We do not claim that our approach necessarily gives better performance than [7], [8]; its appeal is that, given the design choices we have made, there is a provably optimal communication protocol that can be derived using tools from information theory. Our approach also admits a strong link to other interfaces that have been previously designed for text entry—this link, which we will make explicit in Section II-A, gives us some hope that our approach may be more broadly generalizable.

As a case study, in this paper we present a brain-machine interface that allows a human pilot to remotely teleoperate

A. Akce is with the Department of Computer Science and M. Johnson and T. Bretl are with the Department of Aerospace Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA {aakce2,mjohns50,tbretl}@illinois.edu

an unmanned aircraft flying at a fixed altitude using EEG signals. We construct an ordered symbolic language of smooth planar curves (Section II-B) and use these curves as desired paths for the aircraft, tracked with an onboard control system. We use a binary classifier to distinguish between left- and right-hand motor imagery in the brain based on the EEG signals, and use a graphical display to provide visual feedback to the pilot. The underlying problem is then to design a communication protocol by which the pilot can, with vanishing error probability, specify a string in the language of curves using a sequence of bits sent through a binary symmetric channel in the presence of noiseless feedback (Section II-C). Such a protocol is provided by the combination of arithmetic coding as a method of lossless data compression with posterior matching as a capacity-achieving channel code (Section III). Preliminary hardware experiments demonstrate the feasibility of this approach (Section IV).

It should be emphasized that we do not believe an EEG is necessarily a practical input device for flying an aircraft, although we do think it serves as a useful demonstration of what can be done even with a minimally-invasive neural sensor. Instead, we hope that the unique challenges presented by this system will push the state-of-the-art in brain-machine interface design, possibly opening doors to new and more practical applications that have yet to be defined.

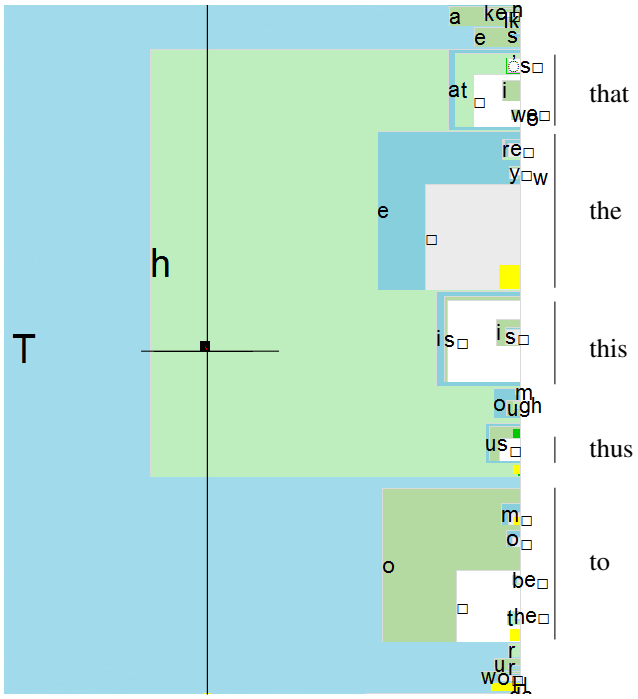

Fig. 1. The Dasher text-entry interface. Each colored box represents a character (alphabetical top to bottom, sequentially nested left to right), with width proportional to its conditional probability. After driving the cursor toward the character "t" and then "h", it becomes easier to select common words like "the" and "this" as a result of the probabilistic language model.

II. PROBLEM FORMULATION

A. Model Interface for Text Entry

Our approach is inspired by an interface for text entry that is shown in Fig. 1, in which users drive a cursor to select characters from a menu [10]. Experienced users can type at 35 words/minute with a mouse, at 25 words/minute with an eye tracker, at 16 words/minute with a breath mouse (driven by lung volume), and at 1-2 words/minute with EEG [5]. Our work with this interface has shown similar performance even with a binary input device driven by EMG [11].

Text has three key properties that make it possible to implement this interface:

a) An ordered symbolic language: Text is made up of symbols $\sigma_i \in \Sigma$ from the alphabet $\Sigma = \{a, b, \ldots, z\}$, perhaps augmented to include punctuation. It has a sequential structure, in the sense that we write it one symbol at a time. It also has an intuitive lexicographic ordering, in the sense that we know the word *rambo* comes "before" the word *robot*. Order is particularly important, because it makes clear which direction the user should move the cursor.

b) A probabilistic language model: The portion of the menu occupied by each character is determined by its conditional probability. This probability comes from a language model, which assigns $P(\sigma_1\sigma_2\cdots\sigma_n)$ to each sequence of symbols in Σ^*.

c) A natural way to display likely words.: The conditional probability of a given character increases the longer the cursor moves toward it. As the corresponding interval grows large, it is recursively subdivided into a second menu of characters, then a third, etc. In other words, rather than symbol-by-symbol coding (e.g., Huffman coding), the interface uses arithmetic coding to place characters in the menu. Rather than select one character at a time, the user continuously "zooms in" on the complete sentence that they want to type.

B. Symbolic Language Model for Smooth Planar Curves

It is possible to construct an ordered symbolic language that gives smooth planar curves the same three properties listed above. Any smooth two-dimensional curve $\gamma\colon [0, L] \to \mathbb{R}^2$ of arbitrary length L can be described by $x' = \cos\theta$, $y' = \sin\theta$, $\theta' = -\kappa$, where s is the arc-length, $\theta(s)$ is the angle of the tangent to the curve at $(x(s), y(s))$, and $\kappa(s)$ is the curvature. The curve is straight when $\kappa = 0$, turns left when $\kappa < 0$, and turns right when $\kappa > 0$. Our language models a subset of curves for which κ is piecewise constant on intervals of length d. We further restrict these curves by choosing each $\kappa_i \in \{c_1, \ldots, c_m\}$ from a finite set. We associate a symbol σ_i with the arc generated by κ_i. We define an alphabet $\Sigma = \{\sigma_1, \ldots, \sigma_m\}$ so that curves can be described concisely as strings of symbols from Σ (see Figs. 2 and 3). It is like a *chain code* [12], but with curvatures rather than cardinal directions as symbols.

The sequential structure of our language allows a user to "spell" curves, one symbol at a time, just like he or she would spell text. Our language also admits an intuitive lexicographic ordering. For two arcs $\sigma_i, \sigma_j \in \Sigma$, we say that $\sigma_i < \sigma_j$ if and only if $i < j$. This definition corresponds

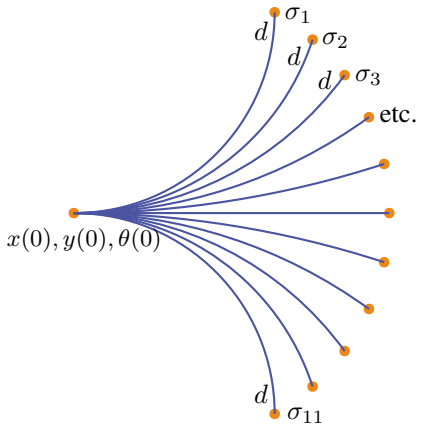

Fig. 2. An example alphabet used in our language of smooth curves. Each symbol σ_i is a circular arc with its length given by d and its curvature given by κ_i for a starting configuration $x(0), y(0), \theta(0)$. κ_i values were chosen such that the change in orientations are evenly spaced in $[-\pi/2, \pi/2]$.

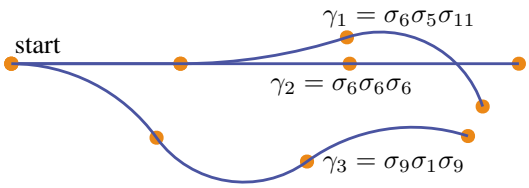

Fig. 3. Three smooth curves composed from the symbols of Fig. 2. These curves exhibit the ordering $\gamma_1 < \gamma_2 < \gamma_3$.

to the notion that σ_i turns left more sharply than σ_j. For two curves $\gamma_i = \sigma_{i_1} \cdots \sigma_{i_N}$ and $\gamma_j = \sigma_{j_1} \cdots \sigma_{j_M}$ we say that $\gamma_i < \gamma_j$ if and only if $i_k < j_k$, where k is the minimum index for which $\sigma_{i_k} \neq \sigma_{j_k}$. This definition corresponds to the notion that γ_i turns left at the first point at which it differs from γ_j (see Fig. 3). This ordering allows a user to "alphabetize" curves just like he or she would alphabetize strings of text. It also allows the user to search for curves with a binary input, just like he or she would search for words in a dictionary by turning pages forward or back. Finally, our language allows the interface to perform inference. We can think of the sequence as being generated by a Markov process. From offline or online data, we can compute a Markov model that assigns a conditional probability to each symbol.

C. Interface Design as a Communication Problem

Our goal is to design a brain-machine interface that allows a human pilot to remotely teleoperate an unmanned aircraft flying at a fixed altitude using EEG signals. We may cast this as a communication problem by making the following three choices: (1) to use smooth planar curves from our symbolic language as desired paths for the aircraft, tracked with an onboard control system; (2) to use a binary classifier to distinguish between left- and right-hand motor imagery in the brain based on EEG signals; and (3) to use a graphical display to provide visual feedback to the pilot. We will say exactly how these choices are implemented in Section IV, but these details affect neither our problem formulation nor our solution approach.

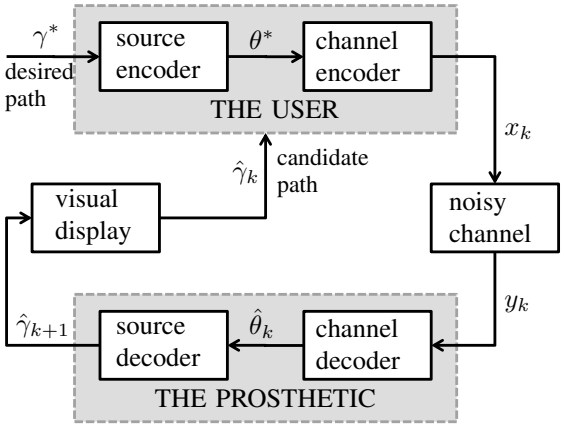

Fig. 4. Information-theoretic approach to the design of brain-computer interfaces. The desired path γ^* from our symbolic language Σ is conveyed through a noisy discrete memoryless channel (in our case, a binary symmetric channel) with noiseless feedback provided by a visual display. At time step k, the display shows a candidate path $\hat{\gamma}_k$. The user, modeled as the encoder, attempts to give the input x_k to the channel by comparing the order of $\hat{\gamma}_k$ to γ^*. $x_k = 0$ or "turn left" if γ^* is to the left of $\hat{\gamma}_k$, otherwise $x_k = 1$ or "turn right." The prosthetic, modeled as the decoder, generates a new candidate path from the noisy channel output y_k. The protocol is designed in two parts, a source code and a channel code.

Our first choice (use curves as desired paths) allows us to define the pilot's objective as specifying a string in an ordered symbolic language with finite cardinality. Our second choice (use a binary classifier to interpret EEG signals) allows us to model the EEG sensor as a noisy discrete memoryless channel, in particular as a binary symmetric channel (BSC) with some crossover probability ϵ. The input to this channel is $x_k \in \{0, 1\}$, where we associate $x_k = 0$ with the command to "turn left" and $x_k = 1$ with the command to "turn right." The output of this channel is $y_k \in \{0, 1\}$, where $P(y_k|x_k) = \epsilon$ if $y_k \neq x_k$ and $P(y_k|x_k) = 1 - \epsilon$ otherwise. Our third choice (use a graphical display) allows us to assume that the BSC can provide causal noiseless feedback, in this case expressed as a candidate path.

With this abstraction, we can reformulate the problem of interface design as the problem of constructing an optimal communication protocol (see Fig. 4). We decompose this protocol into a *source code* and a *channel code* in the usual way [14]. The purpose of the source code is to provide a compact and canonical representation of the desired sequence $\gamma^* = \sigma_1 \cdots \sigma_n$. The result is a *message point* denoted by $\theta^* \in [0, 1)$ to be transmitted over the channel. The purpose of the channel code is to specify the sequence of inputs (in this case, bits) that the user should generate in order to ensure a vanishing probability of error in transmitting $\theta^* \in [0, 1)$. Because we assume noiseless feedback, the channel code should be viewed here as a closed-loop feedback policy rather than a standard error-correcting code. Good choices for both a source code and a channel code will be derived in the following section. Keep in mind that "good" in this context means not only that codes result in high performance but also that they can actually be implemented by a human.

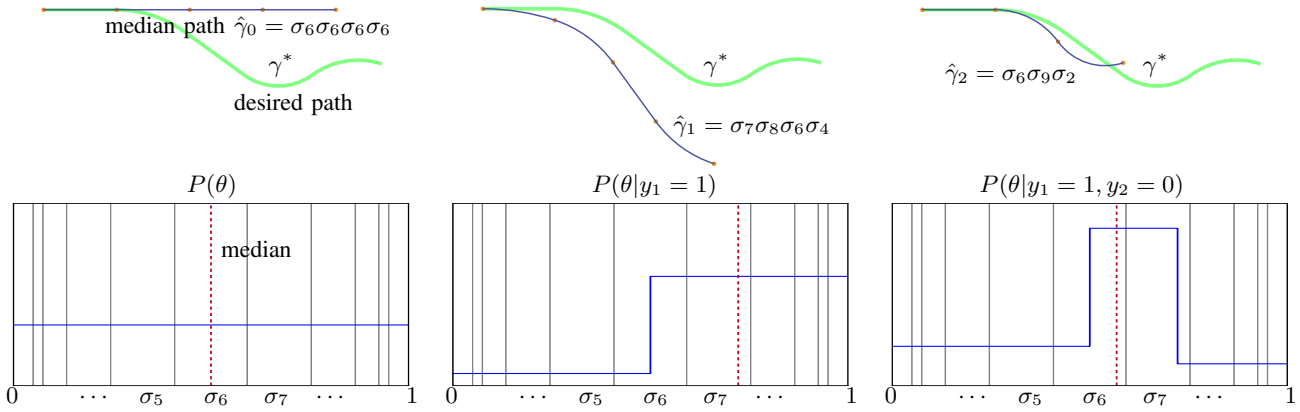

Fig. 5. The communication protocol between the user and the brain-computer interface. The interface generates smooth planar curves from the alphabet of circular arcs given in Fig. 2. The user commands binary inputs to specify the desired path γ^*. The interface maintains a posterior distribution over the unit interval. The graphical display shows the path corresponding to the median of this distribution. This correspondence is determined by arithmetic coding with respect to a statistical model of symbols. In this example, the model is assumed to be a unigram model as depicted in the figures by the size of the subintervals. In the initial step, shown in the first frame, the posterior is uniform and its median is decoded into straight arcs (see $\hat{\gamma}_0$). Upon an observation of a "turn right" command, the likelihood of paths to the right of the median are increased. As a result, the displayed path moves to the right of $\hat{\gamma}_0$, as shown in the second frame. Similarly, the third frame shows the new posterior and the corresponding path after an observation of "turn left."

III. SOLUTION APPROACH

In the previous section, we formulated the problem of brain-machine interface design as the problem of constructing an optimal communication protocol consisting of a source code and a channel code. In this section, we derive these codes using tools from feedback information theory.

A. Source Code

Our goal here is to construct a mapping from random sequences $s = (\sigma_1\sigma_2\cdots)$ distributed according to a statistical language model $P(\sigma_i|\sigma_1\cdots\sigma_{i-1})$ to points $\theta(s) \in [0,1]$ distributed uniformly. We do this using a method of lossless data compression called arithmetic coding [13]. This method is both optimal and has the useful property that it preserves lexicographic ordering, so that $s_1 < s_2$ if and only if $\theta(s_1) < \theta(s_2)$. Given a string $s = (\sigma_1\cdots\sigma_i)$, we denote the *prefix* of s by $\rho = (\sigma_1\cdots\sigma_{i-1})$. An arithmetic code maps each string s to a subinterval $[l_s, r_s) \in [0,1)$ that is computed recursively from the one $[l_\rho, r_\rho) \supset [l_s, r_s)$ associated with its prefix. If the alphabet Σ is of length m, then $[l_\rho, r_\rho)$ is divided into m subintervals arranged in the same order as Σ and with size proportional to $P(\sigma_i|\rho)$. The recurrence is

$$l_s = l_\rho + (r_\rho - l_\rho)\sum_{\sigma<\sigma_i} P(\sigma|\rho)$$
$$r_s = l_\rho + (r_\rho - l_\rho)\sum_{\sigma\le\sigma_i} P(\sigma|\rho),$$

where in the base case we assign the empty prefix $\rho = (\,)$ to the entire unit interval. Note that any real number in $[l_\rho, r_\rho)$ corresponds to a string of infinite length but with prefix ρ. In particular, by a similar recursion we may recover the prefix of length n that corresponds to an estimate $\hat{\theta}_k$ of the message point after k user inputs. At each step $i = 1,\ldots,n$ of this recursion, we add the symbol σ_i given by

$$\sigma_i = \min\left\{\sigma\in\Sigma \,\middle|\, \sum_{\sigma_j\le\sigma} P(\sigma_j|\rho) \ge \frac{\hat{\theta}_k - l_\rho}{r_\rho - l_\rho}\right\} \qquad (1)$$

to the prefix ρ decoded so far. The intuition behind arithmetic coding is that the number of bits required to specify a point in a particular subinterval decreases with the size of this subinterval, and that more likely strings are mapped to larger subintervals. In particular, the average number of bits matches the entropy of the source (a lower bound), as given by the statistical language model [14].

B. Channel Code

Our goal here is to ensure a vanishing probability of error in transmitting a message point θ^* that is assumed to be drawn uniformly at random from the interval $[0,1) \subset \mathbb{R}$. In the presence of feedback, it is possible to significantly reduce the complexity of encoding and decoding and significantly increase the rate at which error probability decreases (even though the capacity remains constant). This is achieved by a posterior matching scheme that admits exponential decay in the probability of error in transmitting θ^* with increasing k [15]. For a binary symmetric channel, this posterior matching scheme has a very simple interpretation. Assume that after time step k, the decoder has computed the posterior distribution $P_{\Theta|Y^k}(\theta|y_1\ldots y_k)$. First, the median of this distribution is selected as the estimate $\hat{\theta}_k$ and given as feedback to the encoder. Then, the encoder selects the next input x_{k+1} as 1 if $\theta^* \ge \hat{\theta}_k$ or as 0 otherwise. Finally, if $y_{k+1} = 1$ (the case $y_{k+1} = 0$ is analogous), then the decoder applies Bayes' rule to update the posterior distribution as

$$P_{\Theta|Y^{k+1}}\left(\theta\middle|y_1\ldots y_{k+1}\right) = \eta\cdot\begin{cases}(1-\epsilon)\cdot P_{\Theta|Y^k}\left(\theta\middle|y_1\ldots y_k\right) & \text{if } \theta\in[\hat{\theta}_k,1)\\ \epsilon\cdot P_{\Theta|Y^k}\left(\theta\middle|y_1\ldots y_k\right) & \text{otherwise,}\end{cases} \qquad (2)$$

where η is a normalizing constant. The new estimate $\hat{\theta}_{k+1}$ is just the median of this distribution. Remarkably, this scheme is not only optimal but also easy for a human user—the "encoder"—to implement. Assume a graphical display shows the user the candidate string $\hat{\gamma}_k = (\sigma_1\sigma_2\cdots)$ that corresponds to the estimate $\hat{\theta}_k$, manifested as a smooth planar curve. Then, the user only has to decide if the desired curve appears lexicographically to the left ("think left") or to the right ("think right") of the candidate (see Fig. 5).

IV. PRELIMINARY HARDWARE EXPERIMENTS

Proof-of-concept experiments show that our approach can be used to remotely teleoperate a real unmanned aircraft flying at a fixed altitude using EEG signals. We selected one able-bodied subject who had several hours of experience in EEG motor imagery from prior studies. The subject was connected to the EEG sensor in the Beckman Institute at the University of Illinois at Urbana-Champaign and the aircraft was flying over a field approximately five kilometers away.

A. Interface to the Human Pilot

We used an EEG sensor to measure brain activity. Eight electrodes were positioned on the scalp near the motor cortex. Output voltages were amplified, low-pass filtered, and sampled at 400Hz. We used the classification algorithm described in [16] to distinguish between left- and right- hand motor imagery. From labelled training data, the classifier uses common spatial analytic pattern (CSAP) to extract discriminative signals that capture large disparities for each class by viewing it as a blind-source separation problem. These signals are processed at 15Hz by a hidden markov model (HMM) to perform classification by belief propagation. This classification occurs at a variable rate, happening whenever the belief probability exceeds a threshold.

The statistical model used by the source code is a fixed unigram model given by discrete gaussian kernel centered on the symbol σ_6 (see Fig. 2), corresponding to the notion that a straight arc has highest probability. This model could also have been generated from training data, for example using prediction by partial matching. The BSC crossover probability assumed by the channel code was $\epsilon = 0.1$.

The graphical display shows onboard video from the aircraft as well as an overhead map of the environment. The map is annotated with the current position and heading of the aircraft and the candidate path $\hat{\gamma}$ (see Figs. 6-7). The pilot modifies the candidate path by modulating the activity in his or her motor cortex. When the aircraft approaches the end of its reference path, the interface sends the first path segment σ_1 of the current candidate path $\hat{\gamma}$ to the aircraft.

B. Remotely Teleoperated Unmanned Aircraft

Our flight system consists of a commercially available glider airframe, the Multiplex Cularis, equipped with a custom autopilot, GPS-aided inertial sensor, and wireless video camera (Fig. 8). The aircraft communicates over a wireless link to a ground station. The ground station communicates with the pilot over a wired network connection—it receives

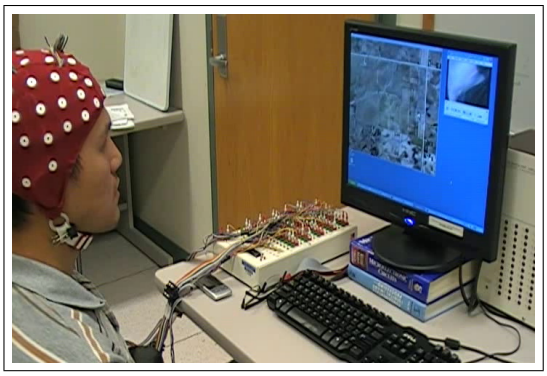

Fig. 6. The pilot remotely teleoperating the aircraft with input only from the EEG sensor. The graphical display shows the state of the system and the live video streamed from the aircraft.

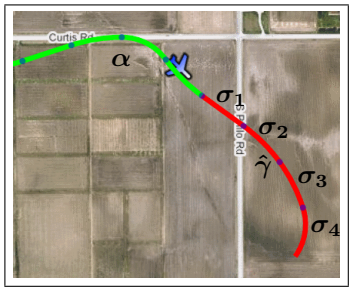

Fig. 7. A zoomed-in snapshot of the graphical display. It shows the map of the environment, the path communicated by the pilot to the BMI, and the state of the aircraft. The path is composed of the candidate path $\hat{\gamma}$ given by the sequence $(\sigma_1\sigma_2\sigma_3\sigma_4)$ and the transferred path α that the aircraft is tracking using its onboard control system.

desired path segment commands from the pilot, and sends telemetry and streaming video back to the pilot.

The onboard computer performs two navigation tasks: low-level stabilization and path following. Low-level stabilization is done using a standard autopilot PID controller. This autopilot is designed to regulate altitude, airspeed, and bank angle. For the purposes of path following, we model the result as a nonholonomic unicycle moving in a horizontal plane and controlled by specifying a desired turning rate. An onboard path following controller using standard "Helmsman Behavior" regulates cross-track and heading errors to zero [17], [18]. The controller does not take into account perturbations due to wind.

C. Task Description and Results

The pilot was asked to perform perimeter surveillance in our preliminary experiments. A large region with a closed boundary (3km) was defined and annotated on the overhead map. The pilot's goal was to fly the aircraft so that the perimeter of this region was covered by the onboard video.

The pilot continuously provided left- and right- motor imagery to specify a desired path to be followed by the aircraft. The task was completed in 5 minutes, and the resulting commanded and flown trajectories are shown in Fig. 9. Over the course of the experiment the subject generated an average of 1.5 binary commands every second. The minimum and

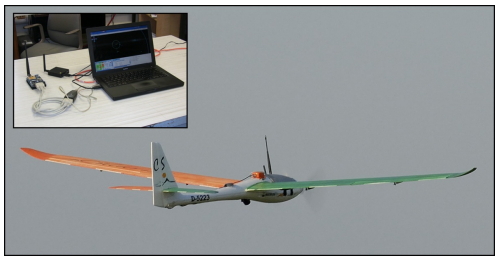

Fig. 8. Aircraft equipped with an autopilot, a GPS device, attitude stabilization sensors, and a wireless video camera. The inset shows the ground station computer with telemetry radio and video receiver.

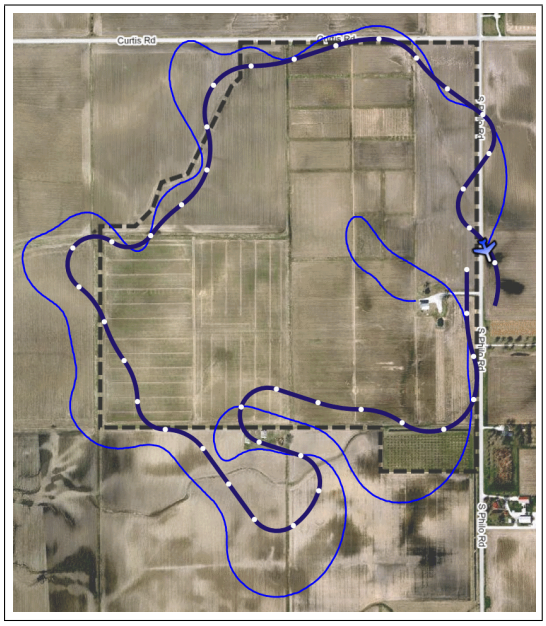

Fig. 9. The results of the perimeter surveillance task. It shows the path specified by the pilot (thick segmented curve) and the actual trajectory of the aircraft (thin solid curve). The dashed lines mark the perimeter of the territory.

maximum times between two consecutive commands were 0.14 and 2.5 seconds respectively. The resulting path was 40 symbols (arc segments) long, specified at a rate of 8 symbols per minute. The real-time behavior of the subject and aircraft can be observed in the accompanying video submission.

V. CONCLUSION

In this paper, we presented a brain-machine interface that has enabled remote teleoperation of an unmanned aircraft using a physical interface that consists only of an electroencephalograph (EEG) to provide input and a graphical display with annotated video from an onboard camera to provide feedback. Our approach was to construct an optimal communication protocol that says exactly how user inputs and sensory feedback should be generated in order to convey intent as quickly and robustly as possible. Despite errors in path following and a heuristic statistical language model, the subject was able to accomplish the task without intensive training. This work provides the foundation for performing more experiments to characterize inter-subject performance, including learning a subject-dependent statistical language model from training data. Future work also includes improving the onboard control, extending our interface to multiple inputs and 3D trajectories, and developing an integrated augmented reality graphical display.

VI. ACKNOWLEDGMENTS

We gratefully acknowledge T. Coleman, E. Maclin, O. Dantsker, N. Kim, A. Aranake, J. Honcharevich, A. Ortiz, and R. Ma for their support and contributions. This research was supported by awards NSF-CNS-0931871 and NSF-CMMI-0956362-EAGER and by a NASA-ISGC Seed Grant.

REFERENCES

[1] D. M. Taylor, S. I. H. Tillery, and A. B. Schwartz, "Direct cortical control of 3d neuroprosthetic devices," *Science*, vol. 296, pp. 1829–1832, 2002.

[2] J. M. Carmena, M. A. Lebedev, R. E. Crist, J. E. O'Doherty, D. M. Santucci, D. F. Dimitrov, P. G. Patil, C. S. Henriquez, and M. A. L. Nicolelis, "Learning to control a brain–machine interface for reaching and grasping by primates," *PLoS Biol*, vol. 1, pp. 193–208, 2003.

[3] R. A. Andersen, S. Musallam, J. W. Burdick, and J. G. Cham, "Cognitive based neural prosthetics," in *Int. Conf. Rob. Aut.*, 2005, pp. 1908–1913.

[4] L. R. Hochberg, M. D. Serruya, G. M. Friehs, J. A. Mukand, M. Saleh, A. H. Caplan, A. Branner, D. Chen, R. D. Penn, and J. P. Donoghue, "Neuronal ensemble control of prosthetic devices by a human with tetraplegia," *Nature*, vol. 442, pp. 164–171, July 2006.

[5] E. Felton, N. L. Lewis, S. A. Wills, R. G. Radwin, and J. C. Williams, "Neural signal based control of the Dasher writing system," *IEEE EMBS Conf. Neural Engr.*, pp. 366–370, May 2007.

[6] 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*, vol. 5, no. 2, pp. 214–220, 2008.

[7] B. Rebsamen, E. Burdet, C. Guan, H. Zhang, C. L. Teo, Q. Zeng, C. Laugier, and M. H. Ang Jr., "Controlling a wheelchair indoors using thought," *Intelligent Systems, IEEE*, vol. 22, pp. 18–24, 2007.

[8] I. Iturrate, J. Antelis, A. Kubler, and J. Minguez, "A noninvasive brain-actuated wheelchair based on a p300 neurophysiological protocol and automated navigation," *Robotics, IEEE Transactions on*, vol. 25, no. 3, pp. 614–627, June 2009.

[9] N. G. Hatsopoulos and J. P. Donoghue, "The science of neural interface systems," *Annu. Rev. of Neurosci.*, vol. 32, no. 1, pp. 249–266, 2009.

[10] D. J. Ward, A. F. Blackwell, and D. J. C. MacKay, "Dasher: A gesture-driven data entry interface for mobile computing." *Human-Computer Interaction*, vol. 17, no. 2/3, pp. 199–228, 2002.

[11] C. Omar, M. Johnson, T. Bretl, and T. P. Coleman, "Policies for neural prosthetic control: initial experiments with a text interface," in *American Control Conference*, Seattle, WA, Jun. 2008.

[12] H. Freeman, "On the encoding of arbitrary geometric configurations," *IRE Trans. Electron. Comput.*, vol. EC-10, pp. 260–268, 1961.

[13] I. H. Witten, R. M. Neal, and J. G. Cleary, "Arithmetic coding for data compression," *Communications of the ACM*, vol. 30, no. 6, pp. 520–540, 1987.

[14] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. Wiley-Interscience, July 2006.

[15] O. Shayevitz and M. Feder, "Optimal feedback communication via posterior matching," *Arxiv preprint 0909.4828*, 2009, submitted to IEEE Transactions on Information Theory.

[16] M. McCormick, R. Ma, and T. Coleman, "An Analytic Spatial Filter and A Hidden Markov Model for Enhanced Information Transfer Rate in EEG-based Brain Computer Interfaces," in *ICASSP*, Dallas, TX, March 2010, to be presented.

[17] R. Rysdyk, "Unmanned aerial vehicle path following for target observation in wind," *Journal of Guidance Control and Dynamics*, vol. 29, no. 5, pp. 1092–1100, 2006.

[18] D. Jung and P. Tsiotras, "Bank-to-turn control for a small uav using backstepping and parameter adaptation," in *International Federation of Automatic Control (IFAC) World Congress*, 2008.