Bimodal Brain-Machine Interface for Motor Control of Robotic Prosthetic

Shalom Darmanjian[1,2] oxygen@cnel.ufl.edu
Sung Phil Kim[1] phil@cnel.ufl.edu
Michael C. Nechyba[1,2] nechyba@mil.ufl.edu
Scott Morrison[1] scott@cnel.ufl.edu
Jose Principe[1] principe@cnel.ufl.edu
Johan Wessberg[3] wessberg@neuro.duke.edu
Miguel A. L. Nicolelis[3] nicoleli@neuro.duke.edu

[1]Computational NeuroEngineering Laboratory
[2]Machine Intelligence Laboratory
Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611-6200

[3]Department of Neurobiology, 327E Bryan Research Building, Box 3209 DUMC, Durham, NC 27710

ABSTRACT

In earlier work on mapping multi-channel neural spike data (recorded from multiple cortical areas of an owl monkey) to corresponding 3d monkey arm positions during reaching tasks, we observed that continuous function approximators (such as artificial neural networks) have difficulty in jointly estimating 3d arm positions for two distinct cases — namely, when the monkey's arm is stationary and when it is moving. Therefore, we propose a multiple-model approach that first classifies neural spike data into two classes, corresponding to two states of the moneky's arm: (1) stationary and (2) moving. Then, the output of this classifier is used as a gating mechanism for subsequent continuous models, with one model per class. In this paper, we first motivate and discuss our approach. Next, we present encouraging results for the classifier stage, based on hidden Markov models (HMMs), and also for the entire bimodal mapping system. Finally, we conclude with a discussion of the results and suggested areas for future research.

1. INTRODUCTION

The University of Florida is part of a four-university, multi-year effort to develop direct brain machine interfaces; a concept diagram of the overall system is shown in Figure 1. As part of this research, we seek to develop models that map multi-channel neural firing data recorded from multiple cortical areas of an owl monkey to corresponding 3d arm positions when the animal reaches for food and eats it.[1] To date, our research group has implemented and investigated a number of continuous models for this reaching task, including various feedforward and recurrent artificial neural networks [2]. It has been observed that one of the largest sources of errors in these models occurs when the monkey's arm is stationary (i.e. not moving); that is, while the trained models track the monkey's arm closely during movement, they do not "turn off" for periods of time when the monkey's arm is not moving.

To deal with this problem, we propose to develop an HMM-based input classifier that classifies incoming neural data into two classes, corresponding to two states of the monkey's arm: (1) stationary and (2) moving. Once developed, this classifier can then serve a gating mechanism for multiple continuous models, the task of each individual model now simplified. The development of such a classifier and evaluation of its performance in a bimodal mapping structure is the main topic of this paper.

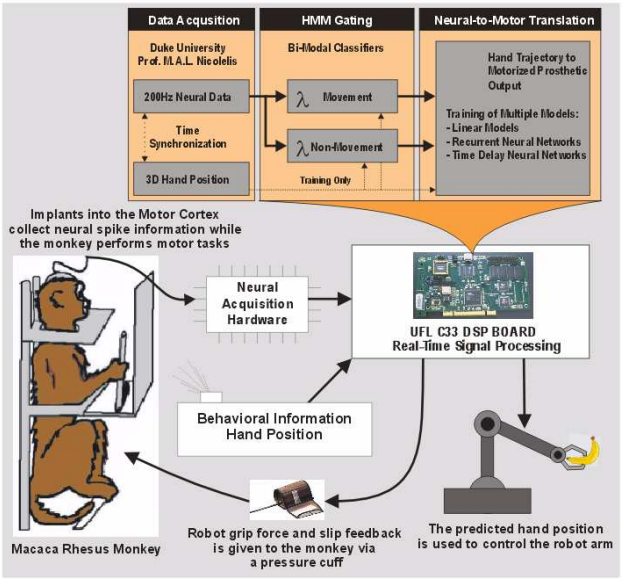

Fig. 1: Overall system diagram.

In this paper, we first discuss our approach to the segmentation problem and subsequent bimodal mapping of neural data to 3d motor movements. We then present experimental results from neural data recorded for an owl monkey. Finally, we discuss our results and suggest further avenues of future research.

2. APPROACH

In this section, we first describe the experimental data and our segmentation of the experimental data into two broad categories (movement and non-movement). We then provide a system-level overview of the neural data classifier, and continue with further details about the classifier. Finally, we describe how we use the classifier output to map the neural activity in the motor cortex of the monkey to 3d arm position.

1. These data have been provided to us by researchers at Duke University who are using micro-wire arrays to simultaneously record the firing patterns for 104 cortical neurons [1].

2.1 Experimental data

Among neural scientists, there is an on-going, unresolved debate whether the motor cortex encodes the arm's velocity, position or both in the neural firing patterns [10,11]. Therefore, we conduct segmentation experiments with two differently segmented data sets, one based on velocity, the other based on displacement. Below, we describe each of these in turn.

For the velocity hypothesis, ideally, the first group should contain data where the arm appears to be stationary, while the second group should contain data where the monkey's arm appears to be moving. We use a simple threshold to achieve this grouping: if the instantaneous velocity of the arm is below the noise threshold of the sensor (determined by inspecting the velocity data visually), the corresponding neural data are classified as *stationary*; otherwise, the neural data are classified as *moving*. In Figure 2, we plot the instantaneous velocity of the monkey's arm for a 500 second segment of the data, where the monkey is repeatedly performing a reaching task. Based on this plot, we choose 4 mm/sec as the noise threshold for the above procedure.

For the position hypothesis, we seek to classify the monkey's arm movements based on displacement. We use Figure 3 below to illustrate this concept. In Figure 3(a), we plot a sample feeding session for the monkey, where the three colored trajectories represent displacement along the three Cartesian coordinates, as the monkey is moving its arm from rest to the food tray, from the food tray to its mouth and, finally, back to the rest position. Figure 3(b) indicates the segmentation of this data into two distinct displacement classes: *rest* and *active*, which are analogous to the *stationary* and *moving* classes in the velocity-based segmentation above.

From Figure 3(c) and (d), where we plot the velocity of the trajectories in Figure 3(a) and (b), we see that this segmentation is not, in fact, the same as the velocity-based segmentation. Note from the yellow-lines (indicating the velocity threshold previously described) that some of the *active* class in the displacement-based segmentation is classified as *stationary* in the velocity-based segmentation. The question, remains, of course, which type of segmentation is likely to be more biologically plausible and, consequently easier to learn. While we will defer our thoughts on this question until Section 3, we do note that keeping an arm stationary (1) at rest, or (2) in extension requires different muscle actions. In the first case, muscles can be relaxed, while in the second case, at least some muscles must be tensed. Therefore, the velocity-based segmentation may actually cause two entirely different sets of neural encodings to be lumped into the same broad category, potentially complicating the segmentation task.

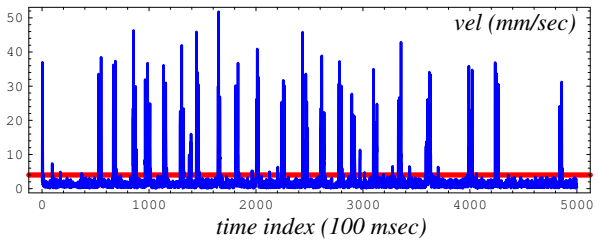

Fig. 2: Instantaneous velocity of monkey's arm movement.

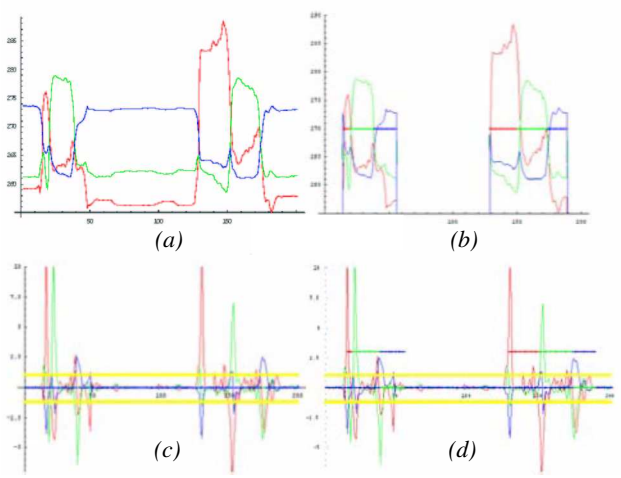

Fig. 3: (a) Cartesian trajectory of monkey's arm during feeding session; (b) displacement-based segmentation; (c) & (d) comparison to velocity-based segmentation.

2.2 Classifier overview

We now want to train two statistical models corresponding to the two classes of data in the previous section. Based on previous statistical analyses [3], these statistical models should capture the temporal statistical properties of neural firings that characterize the monkey's arm movement or lack thereof. One such statistical model, the *hidden Markov model* (*HMM*), enforces only weak prior assumptions about the underlying statistical properties of the data, and can encode relevant temporal properties observed in the data. For these two important reasons, we choose to model the two classes of neural data (stationary *vs.* moving[1]) with HMMs. This choice follows a long line of research that has applied HMMs in the analysis of stochastic signals, such as in speech recognition [4, 5], modeling open-loop human actions [6], and analyzing similarity between human control strategies [7].

Although continuous and semi-continuous HMMs have been developed, discrete-output HMMs are often preferred in practice because of their relative computational simplicity and reduced sensitivity to initial parameter settings during training [4]. A discrete Hidden Markov Model consists of a set of n states, interconnected through probabilistic transitions, and is completely defined by the triplet, $\lambda = \{A, B, \pi\}$, where A is the probabilistic $n \times n$ state transition matrix, B is the $L \times n$ output probability matrix with L discrete output symbols, and π is the n-length initial state probability distribution vector. For an observation sequence O of discrete symbols, we can locally maximize $P(\lambda|O)$ (i.e.probability of model λ given observation sequence O) using the Baum-Welch Expectation-Maximization (EM) algorithm. We can also

1. From this point forward, we will refer to the two classes as stationary *and* moving*, regardless of segmentation approach (velocity vs. displacement). While this is, strictly speaking, not correct for displacement-based segmentation, where we previously used the terms* rest *and* active*, it does significantly simplify notational considerations.*

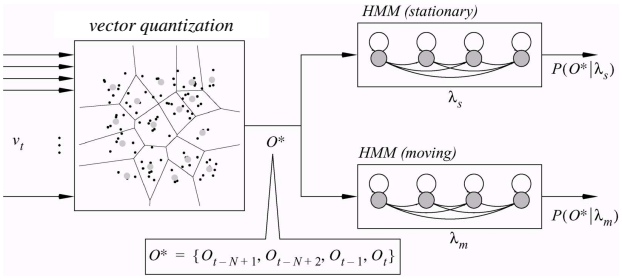

Fig. 4: Stationary/moving classifier structure.

evaluate $P(O|\lambda)$ through the efficient Forward-Backward algorithm.

Figure 4 illustrates then the overall structure of our HMM-based classifier. The classifier works as follows:

1. At time index t, a neural firing vector v_t is first converted to a discrete symbol O_t in preparation for discrete-output HMM evaluation. The method of signal-to-symbol conversion will be discussed in greater detail below.
2. Next, we evaluate the conditional probabilities $P(O^*|\lambda_s)$ and $P(O^*|\lambda_m)$, where,

$$O^* = \{O_{t-N+1}, O_{t-N+2}, O_{t-1}, O_t\}, N > 1, \quad (1)$$

and λ_s and λ_m denote HMMs that represent the two possible states of the monkey's arm (stationary *vs.* moving). In Figure 4, it is assumed that the two HMMs were previously trained on the neural spike data using the Baum-Welch algorithm; details of the training procedure will be discussed in greater detail below.

3. Finally, we classify the state of the monkey's arm according to the two observation probabilities $P(O^*|\lambda_s)$ and $P(O^*|\lambda_m)$.

In step 3 above, a simple decision rule will decide that the monkey's arm is stationary if,

$$P(O^*|\lambda_s) > P(O^*|\lambda_m) \quad (2)$$

and is moving if,

$$P(O^*|\lambda_m) > P(O^*|\lambda_s).^1 \quad (3)$$

The decision rule in equations (2) and (3) is, however, relatively simplistic in that it does not optimize for overall segmentation performance, and does not account for possible desirable performance metrics. For example, it may be very important for the overall modeling scheme to err on the side of predicting arm motion (i.e. action). Therefore, we will investigate a more general decision boundary defined by,

$$P(O^*|\lambda_s)/P(O^*|\lambda_m) = \gamma \quad (4)$$

where γ now no longer has to be strictly equal to one. Note that by varying the value of γ, we can essentially choose our poison — that is, we can tune segmentation performance to fit our particular requirements for such a classifier. Moreover, optimization of the classifier is no longer a function of the individual HMM evaluation probabilities, but rather a function of overall segmentation performance.

1. Note that this segmentation decision implicitly assumes equal prior probabilities for the two classes (stationary vs. moving).

2.3 Signal-to-symbol conversion

Our particular dataset contains 23,000 discrete binned firing counts for the 104 neural channels (each binned count corresponds to the number of firings per 100ms). In order to use discrete-output hidden Markov models, we must first convert the multi-dimensional neural spike data to a sequence of discrete symbols. At a minimum, this process involves vector quantizing the input-space vectors v_t. We choose the well-known LBG VQ algorithm [8], which iteratively generates vector codebooks of size $L = 2^m$, $m \in \{0, 1, \ldots\}$, and can be stopped at an appropriate level of discretization, as determined by the amount of available data. For example, Figure 5 illustrates the LBG VQ algorithm on some synthetic, two-dimensional data (pink/dark area). By optimizing the vector codebook on the neural spike data, we seek to minimize the amount of distortion introduced by the signal-to-symbol conversion process.

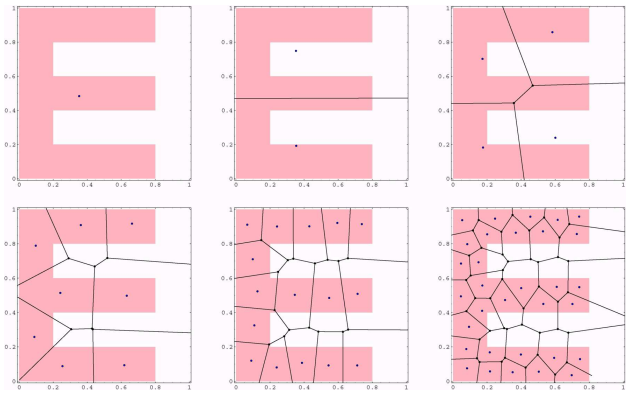

Fig. 5: LBG VQ algorithm on synthetic two-dimensional data.

2.4 HMM structure and training

For this application, we choose the left-to-right (or Bakis) HMM model structure; in this structure non-zero probability transitions between states are only allowed from left to right, as depicted in the HMMs in Figure 4. Given that we expect the monkey's arm movement to be dependent not only on current neural firings, but also on a recent time history of firings, we train each of the HMM models on observation sequences of length N. Since the neural spike data used in this study is binned at 100 msec, $N = 10$, for example, would correspond to neural spike data over the past one second [see equation (1) above]. During run-time evaluation of $P(O^*|\lambda_s)$ and $P(O^*|\lambda_m)$, we use the same value of N as was used during training.

2.5 Bimodal structure and training

The final step in our bimodal mapping structure is to take the outputs from the HMM-based classifier and generate an overall mapping of neural data to 3d arm position. To establish a baseline for this approach — namely the prior segmenting of neural data at the input into multiple classes — we assign a single MA (moving average) model to each class, and train each of the MA models only on that data that corresponds to its respective class, as shown in Figure 6 below. Each MA model adapts its weights using normalized least mean square (NLMS) [12]. After training, test inputs are fed first to the HMM-based classifier, which acts as a

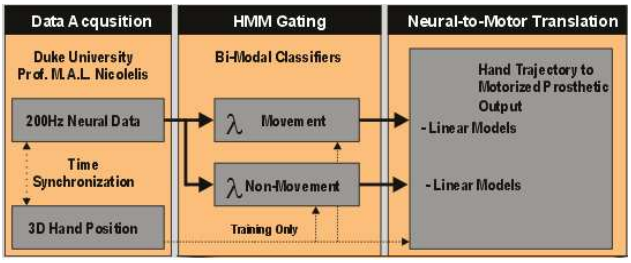

Fig. 6: Bimodal mapping of neural data to arm movement.

gating function for the MA models. Based on the relative observation probabilities produced by the two HMMs, as given in equation (4), one of the two MA models is selected to generate the continuous 3d arm position. With properly trained HMMs, the bimodal system should be able to estimate hand positions with reasonably small error variances.

Given 104 neuron channels, each MA model is defined with 10 time delays (1 sec), and 3 outputs so that its weight vector has 3,120 elements. The MA models were trained on a set of 10,000 consecutive bins (1,000 sec.) of data with a NLMS learning rate of 0.03. Weights for each MA model were adapted for 100 cycles.

After training, all model parameters were fixed and 2,988 consecutive bins of test neural data were fed to the model to predict hand positions. The results of the experiments were then evaluated in terms of short-time correlation coefficients and the short-time signal-to-error ratio (SER) between actual and estimated arm position. For each measure, the short-time window was set to 40 bins (4 sec) since a typical hand movement lasts approximately four seconds.

Of course, a correlation coefficient value of 1 indicates a perfect linear relationship between the desired (actual) and predicted (system) trajectories, while 0 indicates no linear relationship. The second measure, the SER, is defined as the power of the desired signal divided by the power of the estimation error. Since a high correlation coefficient cannot account for biases in the two trajectories, the SER complements the correlation coefficient to give a more meaningful measure of prediction performance. Finally, all our bimodal-system results are compared with two other approaches, namely, a recurrent neural network (RNN) [2] and a single MA model.

3. RESULTS

In this section, we first report results for the HMM-based classifier, and then report results for the complete bimodal system. Furthermore, we compare the bimodal approach with two others — namely, a recurrent neural network (RNN) [2] and a single MA model.

3.1 Classification results

Given our classifier structure in Figure 4 and our decision rule in equation (4), there are a number of design parameters that can be varied to optimize segmentation performance:

$$L = \text{number of prototype vectors in VQ codebook;} \quad (5)$$

$$N = \text{length of observation sequences } O; \quad (6)$$

$$n = \text{number of states in HMM;} \quad (7)$$

$$\gamma = \text{classifier bias.} \quad (8)$$

Table 1: Subsets of neural channels

Subset number	*Explanation*
1	all 104 neural channels
2-5	different combinations of neural channels with statistically significant correlations between arm movement/ non-movement [3]
6	neural channels determined to be significant in ANN mappings of neural activity to 3d arm movement [2]
7	51 neural probe channels [1][1]

[1] *The 104-channel neural data were obtained through spike sorting of data from 51 neural probes.*

In addition, we have determined previously through first- and second-order statistical analysis that some of the 104 recorded neural channels contribute only negligibly to arm movement prediction [3]. Therefore, in our experiments we not only vary the four parameters listed above, but also the subset of neural channels that are used as temporal features in the segmentation process. Table 1 above lists the seven different subsets of neural channels that were used in our experiments.

In Tables 2 and 3, we report experimental results for different combinations of the four parameters in (5) through (8) and subsets of neural channels. These tables present representative segmentation results for a large number of experiments. The L parameter (no. of prototype vectors) was varied from 8 to 256; the N parameter (observation length) was varied from 5 to 10; and the n parameter (no. of states) was varied from 2 to 8.

The two tables differ in how the data was split into training and test sets. In the "leaving-k-out" approach (Table 2), we took random segments of the complete data, removed them from the training data, and reserved them for testing; care was taken that no overlap occurred between the training and test data. In the second approach (Table 3), we split the data sequentially into training and test data in equivalent fashion to the MA training/testing procedure already described. On the one hand, the advantage of the first testing approach is that we can repeat the procedure an arbitrary number of times, leading to more test data, and hence, more statistically significant results. On the other hand, the advantage of the second testing approach is that it uses test data in a manner more likely to be encountered in an eventual BMI system, where a period of training would be followed by a subsequent period of testing.

At this point, we make some general observations about the results in Tables 2 and 3. First, the displacement-based segmentation results are substantially better than the velocity-based segmentation results. Second, we note that the results in Table 2 are marginally better than those in 3. We suggest that one reason for this is that the neural encoding of the small population of 104 neurons is non-stationary to some extent. If the data is non-stationary, we should expect the first testing approach to produce better results, since test data in the "leaving-k-out" approach is taken from within the complete data set, while the second testing approach takes the test data from the tail end of the complete data set.

Table 2: "Leaving-*k*-out" testing

Velocity-based segmentation					
Subset number	*stationary*	*moving*	*L*	*N*	*n*
1	81.5%	83.8%	16	10	6
4	84.0%	75.1%	64	10	4
5	81.0%	82.0%	16	10	3
6	80.4$	81.2%	32	10	6
7	83.2%	82.9%	32	10	3
Displacement-based segmentation					
Subset number	*stationary*	*moving*	*L*	*N*	*n*
1	87.0%	89.2%	16	7	4
4	84.4%	86.6%	64	10	3
5	86.8%	90.3%	32	10	4
6	83.7%	87.8%	32	7	4
7	87.3%	90.0%	32	10	3

Table 3: Sequential testing

Velocity-based segmentation					
Subset number	*stationary*	*moving*	*L*	*N*	*n*
1	82.1%	81.6%	8	7	7
4	81.1%	74.3%	128	10	7
5	81.0%	75.7%	16	10	3
6	80.9%	75.2%	128	10	7
7	81.7%	83.3%	128	10	6
Displacement-based segmentation					
Subset number	*stationary*	*moving*	*L*	*N*	*n*
1	82.1%	85.6%	8	7	7
4	83.5%	87.5%	128	10	6
5	84.0%	87.5%	256	10	6
6	75.6%	81.3%	256	10	4
7	87.3%	86.1%	256	10	5

Finally, from our extensive experiments, which are not all presented here for space reasons, we make one more observation. In general, increased temporal structure leads to better segmentation performance; that is, arm motion is correlated not only with the most recent neural firings, but a short-time history of neural firings.

3.2 Bimodal mapping results

In this section, we report results for neural-to-motor mappings of a single MA model, a recurrent neural network (RNN) and the bimodal system. Since the segmentation results are better for the displacement-based segmentation, we use these HMMs in the first stage of the bimodal system.

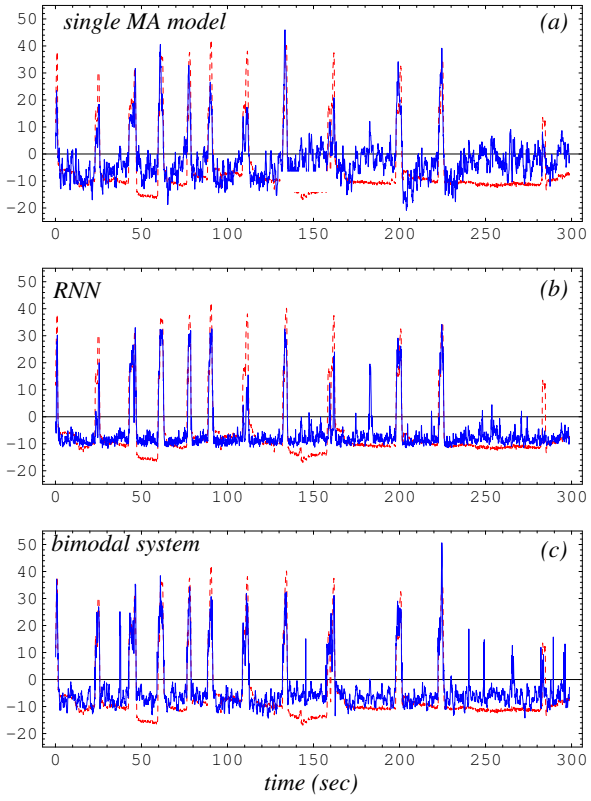

Fig. 7: 3d predicted (blue) and actual arm trajectory (dashed red) for (a) single MA model, (b) RNN and (c) bimodal system over the test data.

In Figure 7, we plot the predicted hand trajectories of each modeling approach, superimposed over the desired (actual) arm trajectories for the test data; for simplicity, we only plot the trajectory along the z-coordinate. Qualitatively, we observe that the bimodal system performs better than the others in terms of reaching targets; this is especially evident for the first, second, and the seventh peaks in the trajectory. Overall, prediction performance of the bimodal system is approximately similar to the RNN, and superior to the single MA model, as evidenced by the empirical cumulative distribution function of L_1 -norms of error vectors, plotted in Figure 8 below. Figure 8 shows that the population distribution functions of L_1 -norms of error vectors of the bimodal system and the RNN are similar, and significantly better than the single MA model.

In Figure 9 we examine the SER for a 100-second long segment of data; part (a) plots the desired Cartesian trajectory for that 100 seconds, while part (b) plots the short-time SERs for all three models over the same 100 seconds. Note that the short-time SER of the single MA model is significantly lower than the other two models especially when the arm is not moving. Note also that there is a sharp drop in the SER of the bimodal system around 40 seconds due to erroneous segmentation of the neural input data. (The consequences of this erroneous segmentation in stage 1 of the bimodal system is also evident at the 40 second mark in Figure 7(c) above.)

The overall correlation coefficients over the whole test set, averaged over all three output dimensions, are 0.64, 0.75, and 0.80

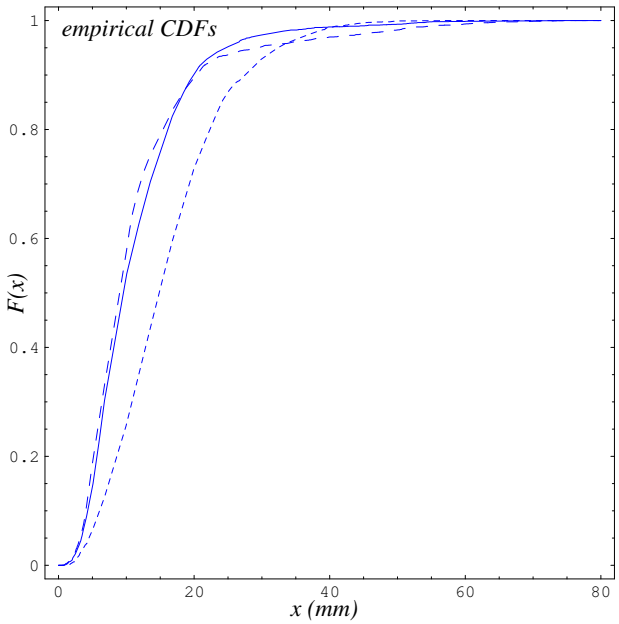

Fig. 8: Empirical CDF of L_1 norms of error vectors for single MA model (short-dashed), RNN (long-dashed) and bimodal system (solid).

for the single MA model, the RNN, and the bimodal system, respectively. The mean of the SER averaged over all dimensions for the single MA model, the RNN, and the bimodal system are $-20.3\text{dB} \pm 1.6$ (standard deviation), $-12.4\text{dB} \pm 16.5$, and $-15.0\text{dB} \pm 18.8$, respectively, while the maximum SERs achieved by each model are 10.4dB, 30.1dB and 24.8dB, respectively.

3.3 Discussion

We note that the prediction performance of the proposed bimodal system is very similar to that of the RNN, and superior to that of the single MA model. Clearly, the use of multiple models improves prediction performance over a single MA model, at some additional computational cost. Compared to the RNN model, the bimodal system produces comparable performance and may lead to a modeling approach that scales better for multiple movements.

Despite these encouraging results, we believe that substantial improvements in performance for the bimodal system are possible. As evidenced by Figures 7 and 9, one of the largest sources of error for the bimodal system is false segmentation in the HMM stage of the mapping. We hypothesize two large contributing factors to that segmentation error. First, given that we are trying to segment arm motion with a very small number of neurons (compared to the total number of neurons in the motor cortex) at the input, it is certainly possible that all the necessary information to perform the segmentation task is simply not available in the measured neural spike data. Second, in the signal-to-symbol conversion of the 104-channel data, we introduce a substantial loss of available information. Even for 256 prototype vectors, the consequent distortion (uncertainty) in the symbol data is substantial, since the 104-channel data does not appear to form tight clusters in the 104-dimensional input space.

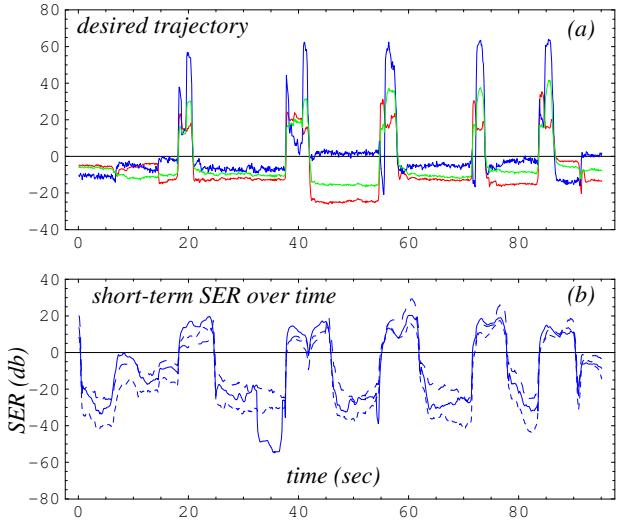

Fig. 9: (a) 100 sec. of desired trajectory (RGB Cartesian coordinates); and (b) corresponding short-time SER for single MA model (short-dashed), RNN (long-dashed) and bimodal system (solid).

Work is on-going by Duke scientists to address the first factor through the development of micro-arrays of electrodes capable of measuring a larger number of neurons in the motor cortex. Thus, we will focus on the second factor, namely altering or eliminating the vector quantization process in our segmentation scheme. While this is an on-going process, we believe that single-channel analysis may suggest possible improvements.

Our initial experiments along these lines have revealed some interesting features of the neural spike data. First, we have observed that different neurons appear to be sensitive to different stages in arm movement. For example, in Figure 10 we plot the ratio $P(O^*|\lambda_m)/P(O^*|\lambda_s)$ for a short segment of data and several different single-channel HMM classifiers. Note that the first neuron appears to be sensitive to the beginning of an arm movement (i.e. reaching for food), while the third neuron plotted appears to be more sensitive to the end of an arm movement. This level of detail may well be obscured by the distortion introduced in the vector quantization process for multiple-channel HMMs.

Second, the segmentation performance of single-channel classifiers for certain neurons is surprisingly good. For example, in Table 4 we report segmentation results for five single neuron classifiers. It may well be possible to combine segmentation results of these individual classifiers to build a better classifier based on all neurons, yet introduce no vector quantization.

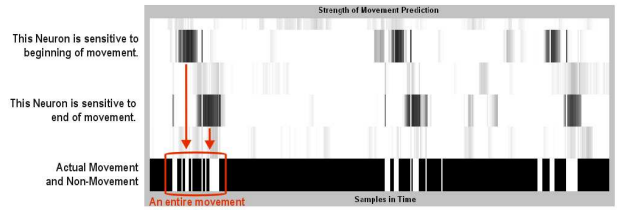

Fig. 10: Different neurons are sensitive to different stages of arm movement.

Table 4: Single-neuron segmentation experiments

Neuron #	*stationary*	*moving*
23	83.4%	75.0%
62	80.0%	75.3%
8	72.0%	64.7%
29	63.9%	82.0%
72	62.6%	82.6%

Finally, while we have focussed our discussion primarily on the first stage of the bimodal system, we also anticipate future improvements in the second stage of the system. Just as a more sophisticated model, such as an RNN, outperforms a simpler model, such as a single MA model, the same may be true in the multiple-models approach. Thus, we are currently working towards replacing the MA models in the second stage of the bimodal system with more expressive modeling frameworks.

4. CONCLUSION

In this paper, we have proposed a bimodal system for mapping neural spike data to 3d arm movement. This system consists of an initial classifier stage and a second mapping stage. We have shown that this system performs better than a single MA model, and comparable to an RNN model, with reduced training and evaluation complexity. Finally, we have suggested some avenues of future research that we hope will improve performance of the bimodal system further.

5. ACKNOWLEDGEMENTS

The work described in this paper was supported by DARPA grant# N66001-02-C-8022.

6. REFERENCES

[1] M. A. Nicolelis, *et al.*, "Simultaneous Encoding of Tactile Information by Three Primate Cortical Areas," *Nature Neuroscience*, vol. 1, no. 7, pp. 621-30, 1998.

[2] J.C. Sanchez, S.P. Kim, D. Erdogmus, Y.N. Rao, J. Principe, J. Wessberg and M. A. Nicolelis, "Input-Output Mapping Performance of Linear and Nonlinear Models for Estimating Hand Trajectories From Cortical Neuronal Firing Patterns," *Proc. of NNSP' 02* pp. 139-148, Martigny, Switzerland, 2002.

[3] S. Darmanjian, "Elementary Statistical Analysis of Neural Spike Data," Internal Technical Report, CNEL, University of Florida, August 2002.

[4] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257-86, 1989.

[5] X. D. Huang, Y. Ariki and M. A. Jack, *Hidden Markov Models for Speech Recognition*, Edinburgh Univ. Press, 1990.

[6] J. Yang, Y. Xu and C. S. Chen, "Human Action Learning Via Hidden Markov Model," *IEEE Trans. Systems, Man and Cybernetics, Part A*, vol. 27, no. 1, pp. 34-44, 1997.

[7] M. C. Nechyba, *Learning and Validation of Human Control Strategies*, CMU-RI-TR-98-06, Ph.D. Thesis, The Robotics Institute, Carnegie Mellon University, 1998.

[8] Y. Linde, A. Buzo and R. M. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Trans. Communication*, vol. COM-28, no. 1, pp. 84-95, 1980.

[9] S. P. Kim, "Modeling The Relation From Motor Cortical Neuronal Firing To Hand Movements Using Competitive Linear Filters/MLP," Internal Technical Report, CNEL, University of Florida, July 2002.

[10] E. Todorov, "On the Role of Primary Motor Cortex in Arm Movement Control," to appear in *Progress in Motor Control III*, 2003.

[11] "One Motor Cortex, Two Different Views," Letters to the Editor, *Nature Neuroscience*, vol. 3, no. 10, October 2000.

[12] S. Haykin, *Adaptive Filter Theory*, Prentice-Hall, Upper Saddle River, NJ, 1996.

[13] J. Wessberg, R. Stambaugh, J.F. Kralik, P.D. Beck, M. Laubach, J.K. Chapin, J. Kim, J. Biggs, M.A. Srinivasan, M.A. Nicolelis, "Real-Time Prediction of Hand Trajectory by Ensembles of Cortical Neurons in Primates," *Nature*, vol. 408, no 6810, pp. 361-365, 2000.