A Comparative Analysis of Neural Decoders for Hand Trajectory Estimation from M1 Activity

Mohammed Abusadeh*, Erik Garcia Oyono*, Virginia Greco*,
Helena Kosovac Godart*, Anna Pahl*
*Department of Bioengineering, Imperial College London, London, UK
{mohammed.abusadeh24, erik.garcia-oyono24, virginia.greco24, helena.kosovac-godart24, anna.pahl24}@imperial.ac.uk

***Abstract*—This report evaluates three neural decoding methods—Kalman Filter, Support Vector Machine (SVM) with angle-specific Kalman, and Long Short-Term Memory (LSTM) networks—for predicting hand trajectories from monkey neural spike data. Models were implemented on a standard dataset and compared using root mean square error (RMSE), runtime, and residual analysis. The Kalman filter offered efficient but inflexible predictions. Incorporating SVM-based angle classification improved directional accuracy at the cost of higher computation. The LSTM decoder achieved the lowest RMSE and most consistent results but required significantly longer training and inference times. Trade-offs among performance, flexibility, and computational cost are discussed, along with potential improvements in model design and preprocessing.**

***Index Terms*—Brain–Machine Interface, Neural Decoding, Kalman Filter, LSTM, SVM**

I. Introduction

Brain-Machine Interfaces (BMIs) translate Central Nervous System (CNS) activity into artificial outputs to enhance, restore, or replace motor function [1]. Commonly used in neurorehabilitation, BMIs record brain signals and use computational decoders to map neural activity to motor outputs [2]. This report focuses on neural decoders—models that predict movement trajectories by estimating kinematic variables such as velocity and position.

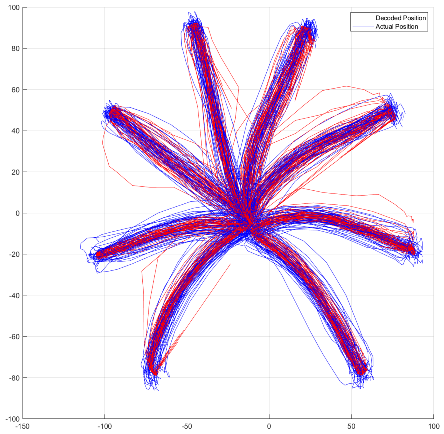

Fig. 1. Hand position estimation using LSTM network. The red line indicates the predicted trajectory; the blue line shows the actual trajectory.

The neural data have been generously provided by the laboratory of Prof. Krishna Shenoy at Stanford University. The code developed for this study is available at https://github.com/eg424/BCI

A. Kalman Filter

The Kalman Filter (KF) is widely used in neuroscience for estimating kinematic variables (e.g., hand position) from noisy observations [3] [4]. In hand tracking, its recursive, real-time nature makes it suitable for smoothing motion trajectories and compensating for sensor noise.

The KF assumes a linear dynamical model with Gaussian noise, allowing for optimal state estimation through two iterative steps:

1) Prediction Step: The next state—hand velocity and position—is predicted from the previous state using Eq. 1:

$$x_k = Ax_{k-1} + Bu_k + w_k \tag{1}$$

Where A is the state-transition matrix, Bu_k is the control input, and w_k is zero-mean Gaussian process noise with covariance Q—. The state vector x_k denotes hand position and velocity.

2) Update Step: The prediction is corrected using a measurement z_k and the Kalman gain to combine the predictions with observations, as explained by Eq. 2:

$$x_k = x_k^{pred} + K(z_k - Hx_k^{pred}) \tag{2}$$

Where K is the Kalman gain, which balances the influence of the prediction and the measurement; H is the observation matrix, mapping the state to the measurement space; and z_k is the measurement derived from neural features.

The effectiveness of KF has been demonstrated in numerous studies. In [5], the authors developed a KF framework where hand position, velocity, and acceleration were treated as the system state, while the firing rates of spike trains served as the observations. Their results showed that this approach not only provided higher accuracy than linear filters but also required less computational effort.

Another study, [6], explores the potential of KF for neural decoding. The authors compare several models for estimating hand trajectories, finding the KF particularly effective due to its dynamic, adaptive framework that models hand position and velocity as system states and neural features as observations. This approach improves trajectory reconstruction accuracy compared to static regression models, making it well-suited for real-time neural decoding applications.

B. Support Vector Machine

Support Vector Machines (SVMs) are well-suited for BMI applications due to their effectiveness in high-dimensional, small-sample, and noisy environments [7]. SVMs identify the optimal hyperplane that maximises the margin between data classes using a subset of training samples known as support vectors to improve generalisation— even with limited training data.

For non-linear classification tasks, kernel functions map inputs to higher-dimensional spaces, enabling both linear and non-linear decision boundaries [7].

Their practicality in BMIs has been demonstrated for real-time motor intention detection using minimal EEG training data [8]. Shevchenko *et al.* [9] found SVMs to be competitive with Linear Discriminant Analysis (LDA) and deep learning models, while offering greater computational efficiency.

Given these strengths, SVMs are widely used in BMIs for discrete classification tasks. In this work, SVMs are used as a discrete angle classifier, serving as the initial stage in a hybrid decoder that combines classification with continuous trajectory estimation using angle-specific KFs.

C. Long Short-Term Memory Network

Deep learning models, particularly recurrent neural networks (RNNs), have gained popularity in neural decoding due to their ability to model temporal dependencies in sequential data [10]. In an RNN, each input X_t is projected via weights w_X into a hidden state H_t, which then recurrently connects to itself through weights w_H over time:

$$H_{t+1} = f(w_H \cdot H_t + w_X \cdot X_t) \tag{3}$$

$$Y_t = g(w_Y \cdot H_t) \tag{4}$$

Where $f(\cdot)$ and $g(\cdot)$ are non-linear activation functions, and Y_t is the output prediction at time t.

Among RNNs, Long Short-Term Memory (LSTM) networks are designed to capture long-range dependencies through gated units that regulate memory flow and update mechanisms, outperforming traditional decoding methods [11].

LSTMs have been applied to BMI tasks with notable success. For instance, Wang *et al.* [12] used LSTMs to achieve robust classification of motor imagery EEG signals. Park and Kim [13] demonstrated that LSTMs could outperform KFs in predicting hand movement direction by simultaneously modeling speed and direction.

Motivated by these successes, this study develops and evaluates an LSTM network for neural decoding.

II. Methods

A. Kalman Filter Decoder

During training, KF processes neural spike data and corresponding hand trajectories to learn a predictive model.

Spike timings and hand positions are preprocessed to extract features including position, velocity, firing rates, their temporal derivatives, and Fano factors. The state transition matrix (A) is estimated using least-squares regression while the process noise covariance (W) is computed from the variability in the training set. The observation model (H) maps extracted features to hand position, with observation noise covariance (Q) derived from the residuals of this mapping.

During testing, hand position is estimated from new spike data. The process starts with initialization, where the initial hand position is obtained from the test data. Firing rates and their first derivatives are then computed using the same method as in training, ensuring consistent feature extraction.

B. SVM with Angle-Specific Kalman Filtering

This hybrid decoder uses a multi-round SVM angle classifier with direction-dependent KFs for trajectory estimation.

During training, spike and hand position data are preprocessed to extract kinematic and neural features. Firing rates are calculated using a 395 ms moving window over cumulative spike counts, along with their temporal derivatives and trial-averaged activity. Hand position is used to estimate x- and y-velocities via finite differencing. Each movement direction is modeled separately, with observation matrices constructed from firing rates, rate changes, and average neural activity to learn angle-specific Kalman parameters (A, W, H, Q).

In parallel, five SVM classifiers are trained to predict movement direction using features from increasing time windows (300–380 ms in 20 ms steps). Each classifier uses a high-dimensional feature vector that includes spike count statistics, Fano factor, total spike count, inter-spike interval metrics, and time-binned firing rates. The features are normalized across trials and encoded using one-vs-all classification.

Testing applies SVMs sequentially until a stable direction is predicted. The corresponding angle-specific Kalman filter is then initialized using the test start position and early velocity estimate, and used to decode ongoing hand movement.

C. LSTM Decoder

The LSTM decoder is designed to model temporal dependencies in neural activity and estimate continuous hand trajectories and movement angles.

1) Data Preparation: Spike data are preprocessed in line with the other decoders. For each trial, movement direction is computed by applying the arc-tangent function to hand position differences over time, yielding angular velocity. Trials are sorted by sequence length to minimize padding, enhancing training efficiency and preserving temporal structure [14].

2) Model Architecture: The LSTM network includes two hidden layers with 200 units each, followed by a dropout layer for regularization. A fully connected output layer maps the hidden states to three outputs: x- and y-positions, and movement angle (θ). The network is trained using the Adam optimizer to minimize mean squared error (MSE) between the predicted and true kinematic values.

3) Prediction: Test spike data are reshaped into the required input format (Channels, Time, Batch) and converted into a deep learning array. The trained model outputs estimates for x, y, and θ, which are interpreted as decoded hand trajectories.

D. Statistical Analysis

To assess model performance, the Kolmogorov–Smirnov test was applied to per-trial RMSE distributions to test for normality. Later, the Friedman test was used to detect overall differences across the three decoders under repeated measures. Post-hoc pairwise comparisons were conducted using Wilcoxon signed-rank tests with Bonferroni–Holm correction to identify pairwise differences.

III. Results

The three decoding models were evaluated on a dataset comprising 50 trials across eight reaching angles. Each model was trained and tested on the same data splits to ensure consistency. Performance was assessed using root mean square error (RMSE), total runtime, average time per prediction, and a weighted performance rank: $0.9 \times \text{RMSE} + 0.1 \times \text{Total Runtime}$.

TABLE I
Performance Metrics for Each Decoder

Decoder	RMSE	Total Time (s)	Time/Pred (s)	Weighted
Kalman	24.6087	39.5083	0.0059	26.0987
SVM + K	20.8070	132.6286	0.0198	31.9891
LSTM	14.5916	155.8600	0.0233	28.7184

The Kalman filter demonstrated the lowest computational cost but had the highest RMSE. Incorporating angle-specific filtering and progressive SVM-based angle classification reduced the RMSE in the hybrid model, although increased runtime. The LSTM decoder achieved the lowest RMSE overall but required the longest runtime and highest average prediction time per sample. As a result, its weighted rank was higher than that of the original KF.

Normality of per-trial RMSE distributions was assessed using the Kolmogorov–Smirnov test. Results indicated that KF errors were normally distributed ($p = 0.7045$), while SVM + Kalman ($p = 0.0352$) and LSTM ($p = 0.0021$) violated normality assumptions. The Friedman test indicated a statistical difference in RMSE among the three decoders ($p = 4.49 \times 10^{-9}$).

Post-hoc Wilcoxon signed-rank tests with Bonferroni–Holm correction confirmed all pairwise differences between the models were statistically different ($p < 0.001$), demonstrating that each decoder's accuracy differs statistically (Fig. 2).

IV. Discussion

In this study, we compared three neural decoding frameworks—the Kalman Filter (KF), a hybrid Support Vector Machine with angle-specific Kalman filtering (SVM+KF), and a Long Short-Term Memory network (LSTM)—evaluating their performance on hand trajectory estimation from primary motor cortex activity. Our findings reveal distinct trade-offs between accuracy, computational cost, and model flexibility.

A. Performance Trade-offs

The basic KF demonstrated the highest efficiency, requiring minimal computational resources and achieving real-time decoding speed. However, as noted by Shevchenko et al. [9], linear models can underperform in dynamic, non-linear tasks with more complex state transitions. This shows KFs linear-Gaussian assumptions and constant-velocity model limited its ability to capture complex or non-linear movement trajectories, resulting in the highest RMSE.

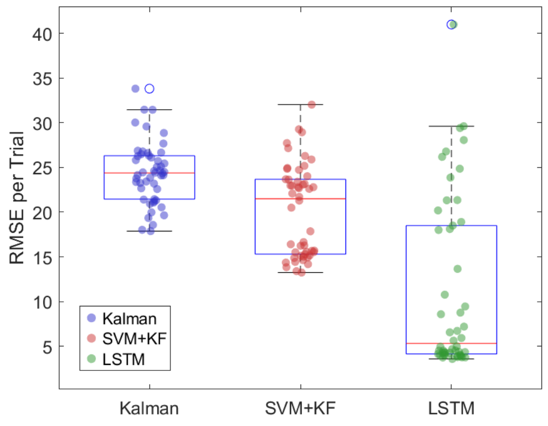

Fig. 2. Comparison of RMSE per trial distribution for Kalman Filter (KF), SVM with Angle-Specific Kalman Filter (SVM+KF), and Long Short-Term Memory (LSTM), with individual trial points overlaid for each model. The colors represent the three different decoder types: blue for Kalman, red for SVM+KF, and green for LSTM.

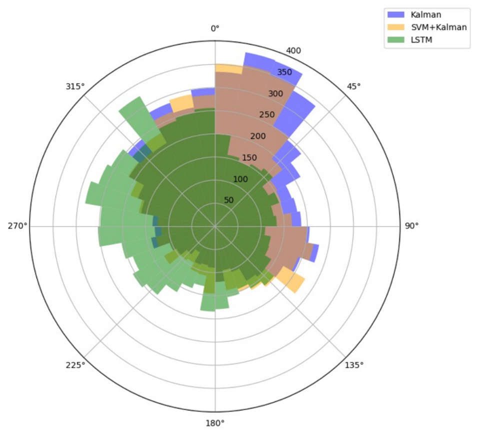

Fig. 3. Rose plot of residual error angles to analyze the systematic directional bias in prediction errors. The angles of the residual error vectors were computed for each model using the arc-tangent of the X and Y errors and binned into 10 degree intervals. Each bar in the rose plot represents the frequency with which the residual errors are pointed in a specific direction, where a higher bar indicates more errors in that direction.

SVM+KF introduced classification angle-specific KFs to better capture direction-dependent patterns. However, the five rounds of classification significantly increased computational cost. Although reclassification improved angle prediction from 87.5% to 93%, the runtime burden remained high, and overall accuracy was still limited by misclassified trials, which led to large trajectory errors and elevated RMSE.

The LSTM model achieved the lowest RMSE and most uniform residual distribution across movement directions (Fig. 3), showing precise modeling of temporal dependencies and non-linear relationships. However, at high computational cost.

B. Statistical Analysis

Kolmogorov–Smirnov confirmed that only the KF error distribution was normally distributed, whereas both the SVM+KF and LSTM errors exhibited non-normal characteristics. The Friedman test detected significant overall differences in RMSE across decoders, and post-hoc Wilcoxon signed-rank tests (with Bonferroni–Holm correction) validated that all pairwise comparisons were statistically significant. These results confirm that the LSTM's lower RMSE represents a true performance gain, and that the SVM+KF hybrid also provides statistical improvement over the base KF.

C. Limitations and Future Directions

Despite LSTM's accuracy, its high computational cost limits its scalability and suitability for rapid prototyping. Future work could explore faster architectures such as Gated Recurrent Units (GRUs), Temporal Convolutional Networks (TCNs) [15], or dimensionality reduction (e.g., PCA) and improved preprocessing techniques to enhance efficiency [10]. Moreover, systematic hyperparameter tuning (e.g., Bayesian optimization [16]) and parallel architectures may further improve prediction accuracy [17].

All models would benefit from optimized feature extraction. As shown in [18], adaptive spike detection thresholds based on target firing rates can improve feature quality with minimal computational cost, directly boosting accuracy. Conventional pipelines often rely on time-averaged spike binning, which risks losing critical temporal information. [8] introduces a sequential Monte Carlo (SMC) framework using a linear-nonlinear Poisson model to probabilistically reconstruct kinematics from raw spike trains. Unlike Gaussian-based models such as the Kalman filter, SMC preserves the full posterior distribution, enabling greater real-time decoding accuracy.

D. Implications for Real-Time BMI Systems

Our comparative analysis discusses that decoder selection must carefully balance complexity, accuracy, and computational cost. While refining deep learning or SVM-based decoders may improve performance, further gains likely depend on enhancing data quality and decoding pipeline structure.

For closed-loop, real-time BMI applications where rapid feedback is needed, the KF or a streamlined hybrid may be preferable. Conversely, in offline decoding or systems with Graphics Processing Unit (GPU) acceleration, deep learning approaches may be a better choice.

V. Conclusion

Our results highlight a trade-off between model accuracy and computational cost. Kalman is efficient but limited; SVM+Kalman improves accuracy modestly with higher cost, while LSTM provides the best predictions with the highest resource demands. Future work should focus on optimizing model efficiency and enhancing feature extraction to improve real-time decoding performance.

VI. Contributions

All authors contributed equally to this work. Detailed contributions are available in the project repository.

References

[1] J. Wolpaw and E. W. Wolpaw, *Brain-Computer Interfaces: Principles and Practice*. OUP USA, Jan. 2012.

[2] W. Q. Malik, W. Truccolo, E. N. Brown, and L. R. Hochberg, "Efficient decoding with steady-state kalman filter in neural interface systems," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 19, no. 1, p. 25–34, Feb. 2011.

[3] S.-P. Kim, J. D. Simeral, L. R. Hochberg, J. P. Donoghue, G. M. Friehs, and M. J. Black, "Point-and-click cursor control with an intracortical neural interface system by humans with tetraplegia," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 19, no. 2, p. 193–203, Apr. 2011.

[4] J. D. Simeral, S.-P. Kim, M. J. Black, J. P. Donoghue, and L. R. Hochberg, "Neural control of cursor trajectory and click by a human with tetraplegia 1000 days after implant of an intracortical microelectrode array," *Journal of Neural Engineering*, vol. 8, no. 2, p. 025027, Mar. 2011.

[5] W. Wu, A. Shaikhouni, J. Donoghue, and M. Black, "Closed-loop neural control of cursor motion using a kalman filter," in *The 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, vol. 4. IEEE, 2004, p. 4126–4129. [Online]. Available: https://doi.org/10.1109/iembs.2004.1404151

[6] H. G. Yeom, W. Hong, D.-Y. Kang, C. K. Chung, J. S. Kim, and S.-P. Kim, "A study on decoding models for the reconstruction of hand trajectories from the human magnetoencephalography," *BioMed Research International*, vol. 2014, p. 1–8, 2014.

[7] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez, "A comprehensive survey on support vector machine classification: Applications, challenges and trends," *Neurocomputing*, vol. 408, pp. 189–215, 2020.

[8] H. J. Choi, S. Das, S. Peng, R. Bajcsy, and N. Figueroa, "On the feasibility of eeg-based motor intention detection for real-time robot assistive control," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[9] O. Shevchenko, S. Yeremeieva, and B. Laschowski, "Comparative analysis of neural decoding algorithms for brain-machine interfaces," *bioRxiv*, Dec 2024, available online: https://doi.org/10.1101/2024.12.05.627080.

[10] J. A. Livezey and J. I. Glaser, "Deep learning approaches for neural decoding across architectures and recording modalities," *Briefings in Bioinformatics*, vol. 22, pp. 1577–1591, 3 2021.

[11] M. Śliwowski, M. Martin, A. Souloumiac, P. Blanchart, and T. Aksenova, "Decoding ecog signal into 3d hand translation using deep learning," *Journal of Neural Engineering*, vol. 19, p. 026023, 4 2022.

[12] P. Wang, A. Jiang, X. Liu, J. Shang, and L. Zhang, "Lstm-based eeg classification in motor imagery tasks," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 26, pp. 2086–2095, 11 2018.

[13] J. Park and S.-P. Kim, "Estimation of speed and direction of arm movements from m1 activity using a nonlinear neural decoder," in *2019 7th International Winter Conference on Brain-Computer Interface (BCI)*, 2019, pp. 1–4.

[14] MathWorks, "Long short-term memory networks," 2023. [Online]. Available: https://uk.mathworks.com/help/deeplearning/ug/long-short-term-memory-networks.html

[15] J. I. Glaser, A. S. Benjamin, R. H. Chowdhury, M. G. Perich, L. E. Miller, and K. P. Kording, "Machine learning for neural decoding," *eneuro*, vol. 7, pp. ENEURO.0506–19.2020, 7 2020.

[16] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," *Advances in Neural Information Processing Systems*, vol. 25, pp. 2951–2959, 6 2012.

[17] X. Ma, S. Qiu, C. Du, J. Xing, and H. He, "Improving eeg-based motor imagery classification via spatial and temporal recurrent neural networks," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 7 2018, pp. 1903–1906.

[18] D. M. Abdullah and A. M. Abdulazeez, "Machine learning applications based on svm classification: A review," *Qubahan Academic Journal*, vol. 1, no. 2, pp. 34–44, 2021.