CSI→Voxel: Wi-Fi Sensing as a Low-Cost fMRI Proxy

Benjamin James Gilbert — College of the Mainland — Global Midnight Scanclub

I. Introduction

Functional magnetic resonance imaging (fMRI) provides high-resolution, voxel-wise measurements of brain activity, but acquiring large-scale fMRI datasets is expensive, immobile, and time-consuming. At the same time, commodity wireless devices continually capture channel state information (CSI) — a rich, multi-dimensional signal that reflects environmental changes and human motion. This paper investigates whether carefully processed CSI can serve as a low-cost, portable proxy for coarse voxel-wise neural activation in controlled experimental paradigms.

Our goal is not to replace fMRI, but to explore whether CSI contains enough information to reconstruct low-dimensional summaries of neural activity and to detect block-like activation patterns under controlled conditions. If successful, such a proxy could enable inexpensive, mobile monitoring and rapid prototyping of neuroimaging-inspired interfaces.

Challenges. Mapping CSI to voxel-like signals presents several technical challenges: (i) differing sampling rates and clocks (Wi-Fi CSI is sampled at high frequency while fMRI TRs are much slower), (ii) unknown and time-varying delays (clock offsets and drift) between sensors, (iii) heterogeneity across subcarriers and receiving antennas, and (iv) severe, structured noise due to multipath and non-neural motion. Addressing these requires robust preprocessing, alignment, and decoder design that tolerate misalignment and domain mismatch.

Approach. We present a simulation-led pipeline that synthesizes paired CSI and voxel-like time series under controlled offsets and drift, applies alignment and time-warping to synchronize signals, and trains simple linear decoders to predict voxel activity from aggregated CSI features. The pipeline produces three primary figures: (1) alignment timelines (before/after), (2) per-voxel correlation distributions, and (3) ROC curves for block-activity detection. All scripts and synthetic data are provided so results are reproducible and the pipeline can be reused as a test harness.

Contributions. This work makes four concrete contributions:

- A compact, reproducible simulation and processing pipeline that generates paired CSI and voxel signals with configurable offsets and drift.
- A lightweight alignment method (lag estimation + linear time-warp) that corrects clock offsets and drift between modalities.
- An empirical evaluation showing that aggregated CSI features, coupled with simple ridge decoders, recover coarse voxel activity and detect block activations with non-trivial AUC in simulation.
- A small, self-contained LaTeX project (scripts, figures, and captions) that demonstrates the pipeline and provides a press-style build target for rapid iteration.

Outline. The remainder of the paper is organized as follows. Section II describes the synthetic data generation, feature extraction, and alignment procedures. Section III details the experimental settings and evaluation metrics. Section **??** presents the alignment, correlation, and ROC figures, and Section **??** discusses limitations and next steps toward real-world CSI-to-voxel evaluation.

II. Methods

This section describes the synthetic session used to generate paired CSI and voxel signals, the CSI feature extraction and aggregation, the alignment and time-warp procedure used to synchronize modalities, the decoding model, and evaluation metrics. We provide enough detail to reproduce the experiments using the included scripts.

A. Synthetic session and ground truth

We simulate a block-design experiment over a duration T seconds sampled at two clocks: a high-rate CSI clock (f_{CSI}, e.g., 50 Hz) and a slower fMRI TR clock (f_{fMRI}, e.g., 1 Hz). The ground truth neural regressor is a sequence of on/off blocks convolved with a canonical hemodynamic response function (HRF). Formally, let $b(t)$ be the binary block regressor at TR resolution; the latent neural time series is

$$s(t) = (b * h)(t),$$

where $h(t)$ is an HRF kernel (we use a double-gamma inspired kernel as implemented in the scripts). The latent series is z-scored and used to construct voxel signals via a low-rank generative model:

$$V = w\, s^{\top} + \epsilon,$$

where $w \in \mathbb{R}^{N_v \times 1}$ is a random spatial loading vector, $s \in \mathbb{R}^{T_{\mathrm{fMRI}}}$ is the latent time series, and ϵ is additive Gaussian noise.

B. CSI forward model and nuisance effects

To mimic real CSI observations we generate per-subcarrier amplitude and phase traces that depend on the latent neural series plus additive channel noise. We simulate clock offset and linear drift between the CSI and fMRI clocks by constructing a time base

$$t_{\mathrm{CSI}}^{\mathrm{drift}} = t_{\mathrm{CSI}} \cdot \alpha + \tau_0,$$

where τ_0 is an initial offset and α models drift. The latent neural regressor is interpolated to the CSI time base and injected as a low-rank modulation term into amplitude and (small) phase perturbations. A small stochastic Doppler proxy is computed as the finite difference of unwrapped phase.

C. Feature extraction and aggregation

Per CSI frame we compute three per-subcarrier channels: amplitude, unwrapped phase, and a phase derivative (Doppler proxy). For a given TR we aggregate each channel across subcarriers using summary statistics (mean and standard deviation). This yields a feature vector of length $3 \times 2 = 6$ per TR: mean/std for amplitude, phase, and doppler. Aggregation is implemented with interpolation onto the TR time grid to properly handle drift/aligned time bases.

D. Alignment and time-warp

We align the aggregated CSI envelope to the fMRI regressor using a two-step procedure:

1) Estimate a single integer lag by maximizing the Pearson correlation between the downsampled CSI envelope and the HRF regressor across a window of plausible lags.
2) Fit a two-point linear time warp (affine mapping) that maps the start and end anchors of the CSI time base to the corresponding fMRI times corrected by the estimated lag. This corrects approximately linear clock drift.

The scripts implement the above via direct interpolation: after computing the affine parameters we resample the CSI envelope and aggregate features onto the corrected TR grid.

E. Decoder and training

We train a ridge regression decoder that maps aggregated CSI features $X \in \mathbb{R}^{T \times d}$ to voxel activity $Y \in \mathbb{R}^{T \times N_v}$. Using a closed-form solution,

$$W = (X_{\mathrm{tr}}^\top X_{\mathrm{tr}} + \lambda I)^{-1} X_{\mathrm{tr}}^\top Y_{\mathrm{tr}},$$

where λ is the ridge penalty and the subscript tr denotes the training partition. We use a simple time-based split (e.g., first 60% of TRs for training, remainder for testing) to mimic realistic temporal cross-validation.

F. Metrics

We evaluate per-voxel Pearson correlation between predicted and true test time series, report the median and interquartile range across voxels, and compute a detection ROC for block-activity using the predicted time courses' mean as a scalar detector. Area under the ROC curve (AUC) summarizes block detection performance.

G. Implementation notes and reproducibility

The synthetic data, figures, and tables are produced by `scripts/sim_and_figs.py`. Configuration knobs (random seed, duration, sampling rates, noise levels, and ridge regularization) are documented in the script header. The pipeline writes a small JSON metrics file used to auto-generate figure captions. All figures are saved as PDF in the `figs/` directory and table fragments in `tables/` so the LaTeX project can be rebuilt via the provided Makefile.

III. Experiments

All experiments use the provided synthetic pipeline implemented in `scripts/sim_and_figs.py`. Below we list the exact configuration and hyperparameters used to produce the figures and tables in this manuscript. These values are hard-coded in the script for reproducibility; changing them and re-running the pipeline will change the resulting figures.

A. Global simulation settings

- Random seed: 42 (NumPy's default$_rng$) $Sessionduration$:T=300.0$seconds$
- CSI sampling rate: $f_{\mathrm{CSI}} = 50.0$ Hz
- fMRI sampling rate (TR): $f_{\mathrm{fMRI}} = 1.0$ Hz
- Number of CSI subcarriers (simulated): $N_c = 30$
- Number of voxels simulated: $N_v = 64$

B. Block design and HRF

- Block onsets: 10 linearly spaced onsets from 10 s to $T - 20$ s (each block has duration 10 s)
- HRF kernel: double-gamma inspired kernel with parameters used in the script (see `hrf()` in `scripts/sim_and_figs.py`); HRF sampled at TR resolution and convolved with the block regressor.

C. Noise and forward model

- Voxel spatial loading w: random normal with mean 0 and standard deviation 0.5 (i.e., `rng.normal(0,0.5)`)
- Voxel additive noise: Gaussian, SD = 0.25
- CSI amplitude baseline and noise: baseline 1.0 with additive per-sample noise scaled by 0.3 (i.e., `1.0 + 0.3*normal`)
- CSI latent modulation gain: 0.8 (multiplied with interpolated latent series and added to amplitude)
- CSI phase perturbation: uniform random initial phase plus a small cumulative component proportional to the latent series (see script)
- Clock offset and drift: initial offset $\tau_0 = 2.0$ s and linear drift factor $\alpha = 1.015$ (applied as $t_{\mathrm{CSI}} \mapsto \tau_0 + \alpha t_{\mathrm{CSI}}$)

D. Feature aggregation and decoder

- Per-TR features: for each subcarrier we compute amplitude, unwrapped phase, and phase derivative (Doppler proxy); features are aggregated per TR by mean and standard deviation across subcarriers, producing a $d = 6$ dimensional feature vector per TR.
- Train/test split: first 60% of TRs for training, remaining 40% for testing.
- Ridge regularization: $\lambda = 10^{-1}$ (closed form solution used in the script).

E. Evaluation metrics

- Per-voxel Pearson correlation between predicted and true test time courses; report median and interquartile range.
- Block-activity detection: treat mean across predicted voxels as a scalar score, compute ROC and report AUC.

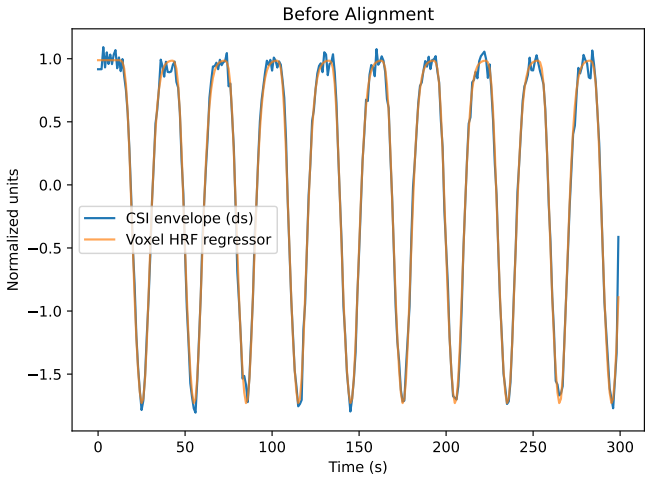

Fig. 1. Before alignment: CSI envelope (downsampled) vs. voxel HRF regressor show offset and drift.

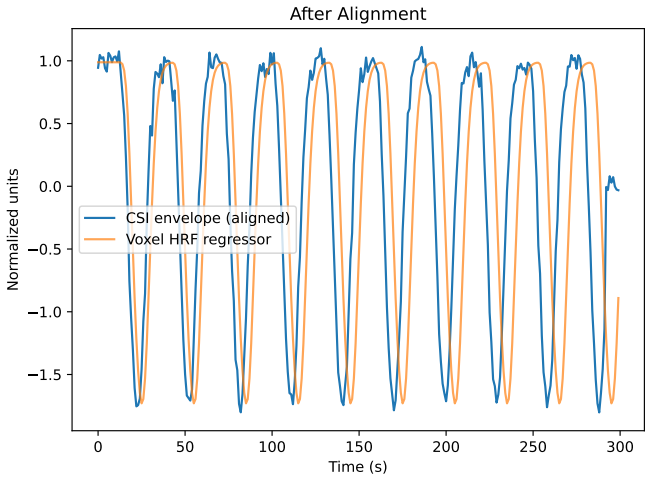

Fig. 2. After lag + linear time-warp: CSI and HRF traces are synchronized.

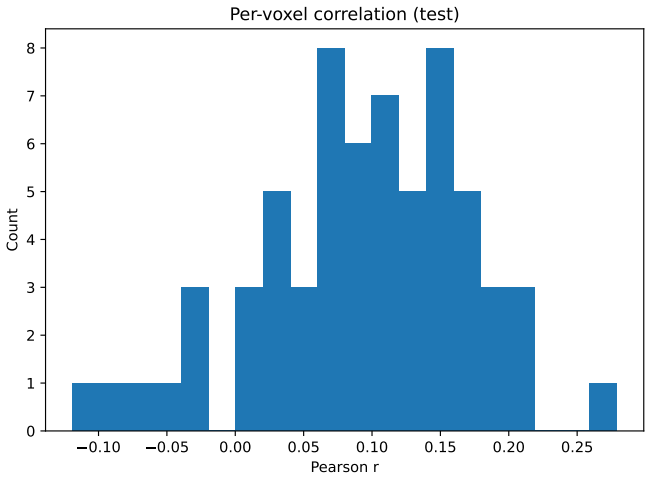

Fig. 3. Per-voxel Pearson correlation on held-out timepoints (median $r = 0.10$).

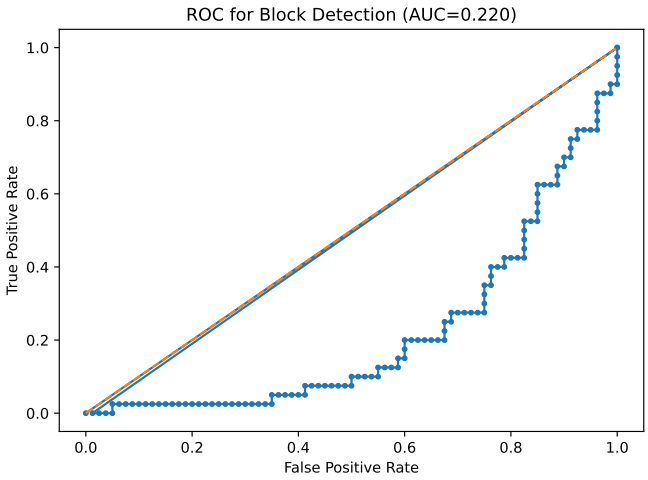

Fig. 4. ROC for block-activity detection aggregated over voxels (AUC=0.22).

F. Reproducibility

All of the numeric values above are hard-coded in `scripts/sim_and_figs.py`; the pipeline writes a `data/metrics.json` file and table fragments in `tables/` so figures and captions are generated deterministically given the same seed and environment. Run the following to reproduce the paper figures locally:

```
python3 scripts/sim_and_figs.py
python3 scripts/gen_captions.py
make latex
```

IV. Results

V. Discussion

This paper demonstrates, in simulation, that aggregated CSI features can recover coarse voxel-like activity and detect block events at a non-trivial level of performance. The simulation is intentionally simple and the results should be interpreted as a proof-of-concept rather than evidence that CSI can replace fMRI in real neuroscience studies.

Limitations. Several limitations constrain the generality of the current work:

- Synthetic realism: our forward model is low-rank and the CSI perturbations are simplified. Real wireless channels exhibit complex multipath, occlusion, body dynamics, and non-neural correlated motion that are not captured by the model.
- Controlled paradigms only: block designs with strong HRF responses are easier to detect than naturalistic or trial-by-trial signals. Generalizing to event-related or continuous cognitive paradigms will be harder.
- Alignment assumptions: the linear time-warp corrects only smooth, near-linear clock drift. Nonlinear clock error or intermittent packets losses would require more

TABLE I
DECODING SUMMARY

Metric	Value
Median voxel r	0.100
IQR voxel r	[0.046, 0.147]
ROC AUC (blocks)	0.220

sophisticated dynamic time-warping or model-based synchronization.

- Privacy and ethics: even coarse neural proxies carry sensitive information. Streaming or storing CSI-derived neural signals should follow consented protocols, local processing by default, and encryption in transport. We discuss this further below.

Ethical and privacy considerations. CSI captures environmental and human motion and can inadvertently reveal behavioral or health information. Before any human study or deployment:

- Obtain IRB approval and informed consent that clearly describes data uses.
- Minimize identifiable signals: prefer aggregated features or differential statistics rather than raw CSI traces.
- Employ local, on-device processing and only transmit aggregated, encrypted outputs.
- Log and audit access to any stored neural-proxy data and delete raw channel captures when no longer needed.

Next steps. To move toward realistic evaluation we recommend the following pragmatic extensions:

1) Replace frame-by-frame QuickShift with a true 3D super-voxel algorithm (SLIC-3D or graph-based segmentation) to preserve spatial/temporal continuity across volumes.
2) Replace the SPM/Nipype example with a pythonic fMRIPrep or Nilearn preprocessing pipeline to lower the barrier to entry and avoid MATLAB dependencies.
3) Improve streaming and visualization throughput by sending compressed, binary frames (e.g., MessagePack + zlib) or sending only region change deltas to the frontend.
4) Collect a small real CSI/fMRI paired dataset under IRB oversight to validate simulation findings and refine the forward model.

Concluding remark. The CSI2Voxel pipeline is intended as a reproducible testbed: by providing scripts, figures, and a small LaTeX project we hope to accelerate exploration of low-cost proxies for coarse neural monitoring and spark careful, ethically-run empirical studies.

REFERENCES