Appendix

A Phosphene Model

A.1 Methods

This section describes the phosphene model used to simulate patient's perception resulting from stimulation. The model takes in a stimulus vector $\boldsymbol{s} \in \mathbb{R}^{n_e \times 3}$ specifying the frequency ($freq$), amplitude (amp), and pulse duration ($pdur$) of a biphasic pulse train on each electrode. In addition, the model also takes in a vector of patient-specific parameters $\boldsymbol{\phi}$ (see Table 1). We break these parameters down into implant parameters (x, y, rot), global parameters (ρ, λ, ω, OD_x, OD_y), and stimulus-related parameters (a_0-a_4); all explained below.

Exact implant locations vary patient-to-patient. The three implant parameters allow our model to account for these changes. We used a simulated implant inspired by designs of real epiretinal implants [3, 56] and those used in previous simulation studies [14]. It consists of 225 disk electrodes (radius $75\,\mu\text{m}$ arranged onto a square, 15×15 grid with $400\,\mu\text{m}$ spacing, initially centered over the fovea. The three implant-related parameters translate and rotate the initial implant to be centered at (x, y), and to be rotated by rot degrees. The implant used is depicted in Figure A.2, overlaid on top of a simulated map of axon nerve fiber bundles [57].

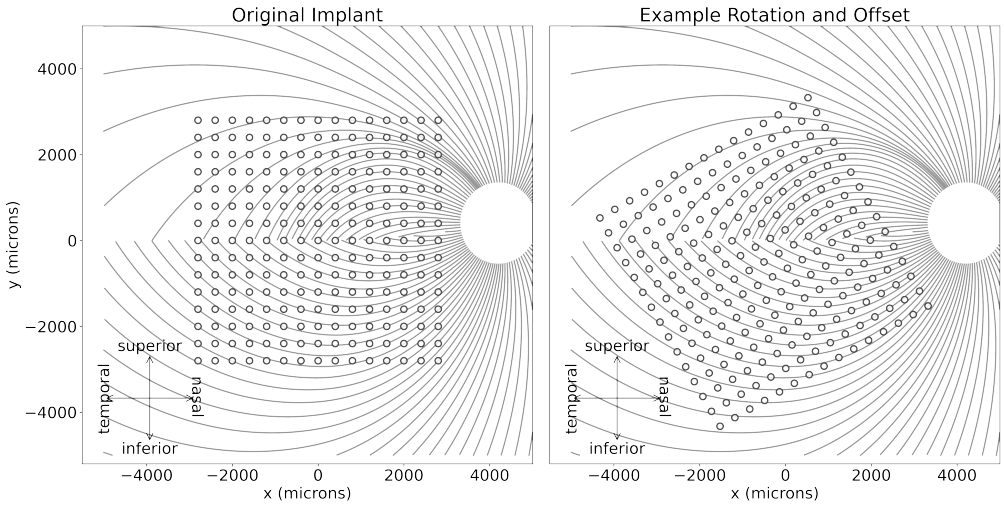

Figure A.1: The implant used for optimization, and an example implant after rotation and translation based on patient-specific parameters ϕ. The white circle on the right is the optic disc. Arced lines depict simulated axon nerve fiber bundles.

The remaining model parameters are inspired by various psychophysical and electrophysiological studies [8, 10, 18, 19, 58], and are summarized in the following list:

- ρ : Average phosphene size. This will be modified locally based on stimulus parameters.
- λ : Average phosphene eccentricity (a measure of phosphene elongation; not to be confused with retinal eccentricity). This will be modified locally based on stimulus parameters.
- ω : Orientation scaling factor. The orientation of phosphenes will be the orientation of the underlying axon bundle, scaled by ω (Eq. 12).
- OD_x, OD_y: The x and y location of the patient's optic disc, into which axon nerve fiber bundles terminate.
- a_0-a_2: Coefficients to modulate phosphene brightness with stimulus parameters (Eq. 9).
- a_3 : Coefficient to modulate phosphene size with stimulus parameters (Eq. 10).
- a_4 : Coefficient to modulate phosphene eccentricity with stimulus parameters (Eq. 11).

Each electrode's location on the retina can be determined from the implant parameters. The corresponding location in the visual field (μ_e) is determined using the retinotopic map described in

Watson *et al.* [59]. Each electrode's phosphene orientation is then $\theta_e = \omega\theta_{axon}$, where θ_{axon} is the orientation of the axon nerve fiber bundle (NFB) underlying the cell (pixel). Axon NFBs are modeled as spirals originating at the optic disc and terminating at each simulated cell. These spirals follow a simulated axon map [47] based on tracings of axon trajectories in 55 human eyes. In summary, phosphene size, eccentricity, brightness, and orientation are modulated based on stimulus parameters and implant location according to the following equations:

$$b_e = a_0(amp_e)^{a_1} + a_2(freq_e) \tag{9}$$

$$\rho_e = \rho * a_3 * amp_e \tag{10}$$

$$\lambda_e = \lambda \left(\frac{pdur}{0.45}\right)^{a_4} \tag{11}$$

$$\theta_e = \omega * \theta_{axon} \tag{12}$$

The phosphene for each electrode is a multivariate Gaussian blob, centered at the electrodes location in visual field (μ_e), and with covariance matrix Σ_e constructed such that the resulting phosphene will have brightness b_e, size ρ_e, eccentricity λ_e, and orientation θ_e, as shown in the following equations (repeated from main text for convenience):

$$b(x, y) = 2\pi b_e \det(\mathbf{\Sigma_e})\, \mathcal{N}([x, y]^\top | \mu_e, \mathbf{\Sigma_e}), \tag{13}$$

The covariance matrix $\mathbf{\Sigma_e} = \mathbf{R}\mathbf{\Sigma_0}\mathbf{R}^T$ is calculated from the eigenvalue matrix $\mathbf{\Sigma_0}$ and a rotation matrix $\mathbf{R}$:

$$\Sigma_0 = \begin{bmatrix} \alpha_e^2 & 0 \\ 0 & \beta_e^2 \end{bmatrix}, \qquad R = \begin{bmatrix} \cos\theta_e & -\sin\theta_e \\ \sin\theta_e & \cos\theta_e \end{bmatrix}.$$

The eigenvalues α_e and β_e depend on the intended phosphene area (ρ_e) elongation (λ_e), and a constant ϵ (set to e^{-2}):

$$\alpha_e^2 = -\frac{\rho_e\sqrt{1-\lambda_e^2}}{2\pi \ln \epsilon}, \qquad \beta_e^2 = -\frac{\rho_e}{2\pi \ln \epsilon \sqrt{1-\lambda_e^2}}.$$

This formulation guarantees that the Gaussian blob, when thresholded using ϵ, will have the intended area, orientation, eccentricity, and brightness.

Blobs from individual electrodes are summed into a global percept. This linear summation is supported by recent studies, which have shown that percepts from multi-electrode stimulation are often linearly related to the percepts from stimulation on the individual electrodes [36]. Although the sum across electrodes is linear, modulating the size and eccentricity of phosphenes makes the final result a nonlinear function of stimulus parameters, preventing analytic inversion.

A.2 Evaluation

Our model is motivated by similar anatomical and psychophysical phenomena as the previous state-of-the-art model for epiretinal prostheses [11], but its formulation allows for favorable computational properties. In comparison, our model is on average 45x faster to run, and consumes about 120x less GPU memory. These computational benefits are the main reason a new model was necessary, and enables a more advanced deep stimulus encoder by allowing training with larger encoder models, longer training duration, and larger batch sizes.

Nonetheless, we also verified that the model produces state-of-the-art predictions, as described in Section 4.1. Despite its similar design, our model achieves much better scores on the Beyeler *et al.* [8] evaluation for phosphene shape. This is likely because our formulation allows much tighter control of phosphene shape attributes (e.g., size, eccentricity), allowing (for the first time) positive R^2 on shape descriptors for held-out electrodes. Our model performs similarly to the previous state-of-the-art model on the Granley *et al.* [11] evaluation, which is to be expected given that the equations modulating phosphene appearance with stimulus parameters in both models are very similar.

Fig. A.2 reproduces the plots from Figures 4A-C and 5 of [11], but with our proposed model included. These figures show the brightness or size rating from Argus II patient(s) as stimulus parameters vary [19, 37], in subjective units. For brightness, '10' means the same as the reference pulse, '20' means twice as bright, etc. For size, '1' means the same as reference pulse, '2' means twice as large (notation matching [11, 19, 37]).

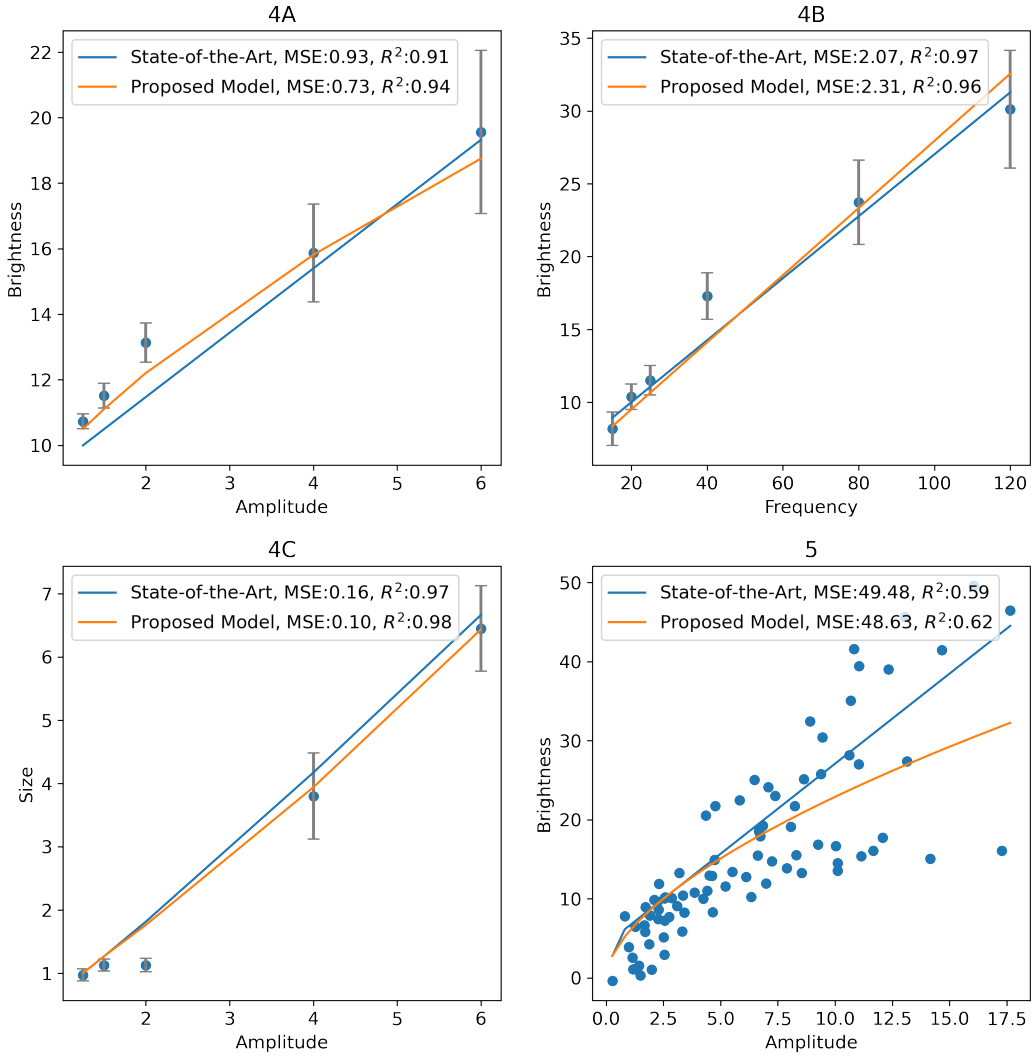

Figure A.2: Evaluation of phosphene brightness and size as stimulus parameters vary. Reproduced from [11], but with our proposed phosphene model included. State-of-the-art denotes the phosphene model from [11]. Units are subjective, in comparison to a reference pulse (i.e. brightness of 20 means twice as bright, size of 3 means 3 times as bright) [19]

B Deep Stimulus Inversion

B.1 Architecture

The encoder architecture described in Section 3 is illustrated in Fig. B.1.

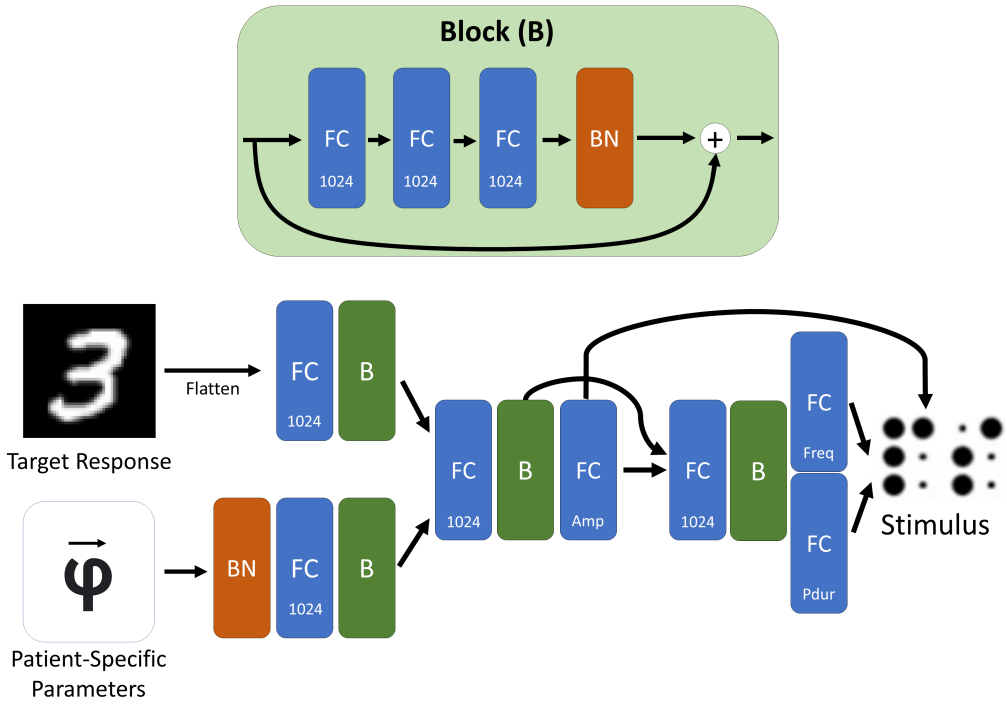

Figure B.1: Deep stimulus encoder architecture. FC: fully connected layer, BN: batch normalization layer, B: block of layers (shown at top). Two arrows merging into one fully connected layer denotes concatenation. 45M total parameters.

C Human-in-the-Loop Optimization

C.1 Kernel Selection and Hyperparameters

This section gives more details on fitting the hyperparameters for the Gaussian process (GP) kernel used in preferential Bayesian optimization (PBO). As stated previously, the performance of PBO crucially depends on the GP kernel and its hyperparameters, which encode our prior assumptions about the latent preference function. To select hyperparameters, we used a transfer learning strategy, selecting hyperparameters that generalized best within a small validation group of simulated patients (see Appendix C.2 for discussion).

We simulated 600 random duels (ϕ_1 and ϕ_2 chosen randomly) on each of 10 simulated patients. For each patient, we fit four commonly used kernels (Squared Exponential, Squared Exponential with Automatic Relevance Determination (ARD), Matérn 3/2, and Matérn 5/2) and inferred hyperparameters using type II maximum likelihood estimation [60]. The bounds for each hyperparameter were $[\exp(-10), \exp(10)]$. For each of these candidate kernel-hyperparameter pairs, we fit a GP with the corresponding kernel and hyperparameters to 50 training duels for each of the other 9 patients. Then, the performance of the candidate GP was evaluated on the remaining 550 data points using Brier score [43], a commonly used metric measuring the accuracy of probabilistic predictions:

$$BS = \frac{1}{n}\sum_{i=1}^{n}(y_{true} - y_{pred})^2, \tag{14}$$

where y_{true} is the true duel outcome (1 or 0) as decided by the simulated patient, and y_{pred} is the probability of ϕ_1 being selected as the winner (corresponding to outcome of 1), as predicted by the Gaussian process.

The kernel and hyperparameters with the lowest Brier score, averaged across all 9 other patients, were selected (Matérn 5/2). To verify that this kernel performed well, we also ran human-in-the-loop optimization (HILO) for 20 random simulated patients, using the best hyperparameters for each of the four previously mentioned kernels. The results are shown in Figure C.1. The Matérn 5/2 kernel performed slightly better than the Matérn 3/2 kernel, and significantly better than the ARD kernel. While performance was similar to the Squared Exponential kernel, we ultimately selected Matérn 5/2 due to its lower Brier score.

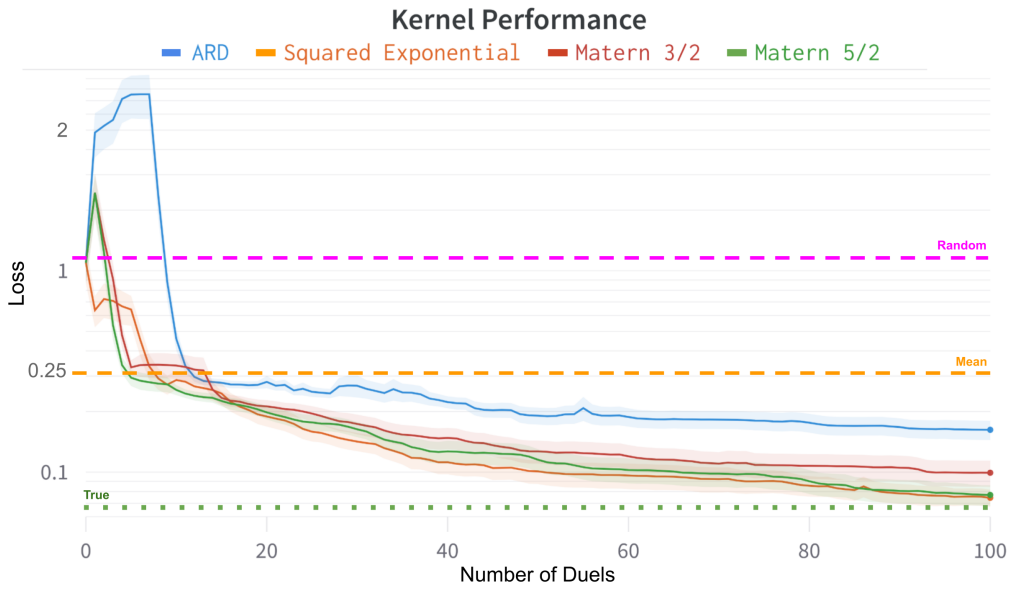

Figure C.1: Joint perceptual loss (y axis, log scale) plotted throughout HILO with different Gaussian process kernels. Error bars denote SEM.

C.2 Hyperparameter Optimality

The Gaussian process kernel hyperparameters selected with our transfer learning strategy performed well in our simulations, leading to higher-quality patient-specific stimulus encodings (Figure 4). This transfer learning approach was chosen to match a clinical setting, where limited human data availability. However, it is certainly possible that better performing hyperparameters exist. To investigate the optimality of our hyperparameters, we examine two other strategies: an ideal case where 'patient-optimal' hyperparameters are used for each patient, and an online strategy where hyperparameters are updated during optimization with each patient,

Patient-optimal Hyperparameter Selection In this strategy, hyperparameters were precomputed for each patient by simulating 200 random duels. Kernel parameters were again fit using type II maximum likelihood estimation [60]. These 'patient-optimal' hyperparameters were then used during HILO for the patient.

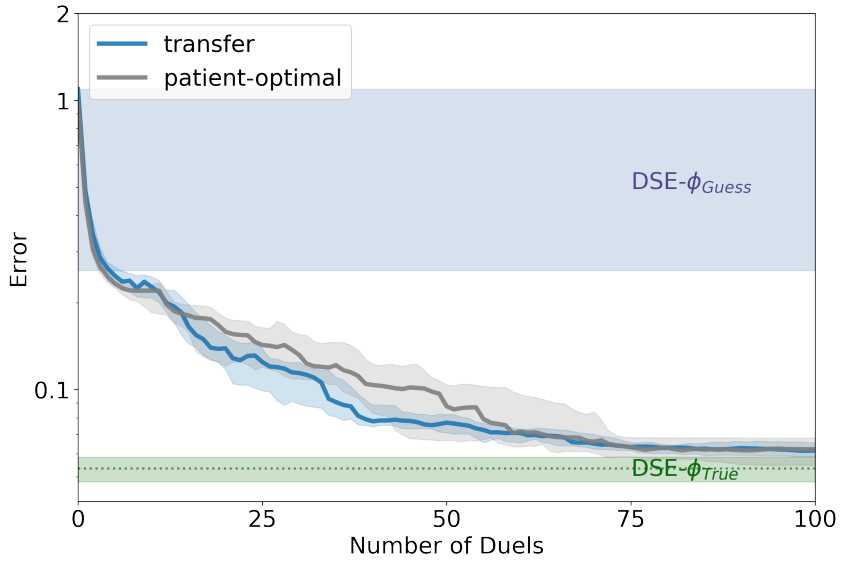

Figure C.2: Reconstruction error using Gaussian process kernel hyperparameters selected via transfer learning and 'patient-optimal' strategies throughout HILO.

Optimization results for 20 random patients are shown in Figure C.2. Both the transfer and the patient-optimal settings led to similar performance, and there was no significant difference between the final reconstruction errors ($p > .05$, two-sided paired t-test). Thus it seems the transfer learning strategy indeed selected hyperparameters that generalize well for new patients. Note that in general the transfer learning selection strategy is more practically applicable than the patient-optimal strategy since it does not require a long calibration period, but the patient-optimal strategy is an alternative that could be used for the first human subjects when no human data (only simulated data) is available.

Online hyperparameters selection While it is common to keep kernel hyperparameters constant during optimization [33, 61, 62], online optimization is an alternative, where kernel parameters are periodically re-fit to the patient data during optimization. We tested online optimization where hyperparameters were recalculated with an update period of 1, 5, 10, or 20 duels, or were never updated. In all settings, the initial hyperparameters were chosen using the transfer learning strategy.

The results for online hyperparameters optimization are shown in Figure C.3. All update periods eventually converged to similar performance as with the transfer learning strategy (*i.e* never updating hyperparameters), but since recalculating hyperparameters is costly, online hyperparameter updates increased the optimization time required to reach a desired performance (Figure C.3, *right*). This suggests that just using the transfer learning hyperparameters is a better strategy than online updates.

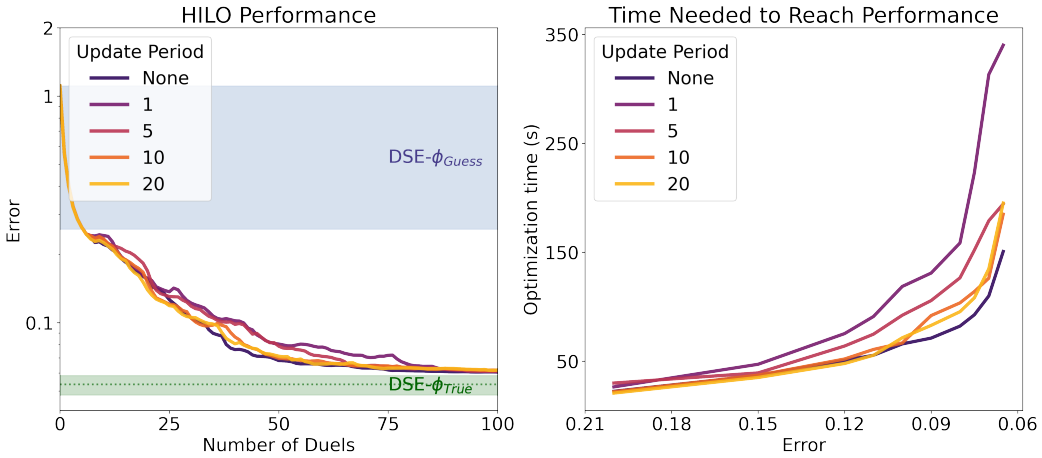

Figure C.3: Reconstruction error (left) and Bayesian optimization time required to reach a specified reconstruction error (right) for HILO with online updates of GP hyperparameters.

C.3 Acquisition function

The acquisition function is responsible for choosing ϕ_1 and ϕ_2 for each duel, and must balance exploration of the search space with exploiting values of ϕ that are expected to work well. The Maximally Uncertain Challenge (MUC) [42] acquisition presented in the main text was initially compared against 2 other top-ranking [42] acquisition functions: Bivariate Expected Improvement (Bivariate EI) [63], and Dueling Upper Credibility Bound (Dueling UCB) [64]. In addition, we also compare against a baseline acquisition, where ϕ_1 and ϕ_2 are chosen randomly for each duel. We ran HILO for the same 20 random patients for each acquisition.

The joint perceptual loss throughout optimization for each acquisition is presented in Figure C.4. All of the tested acquisition functions dramatically outperformed the random baseline, which converged to a value near the mean DSE without HILO. Although MUC and Dueling UCB performed similarly, we ultimately selected MUC due to its slightly lower final loss.

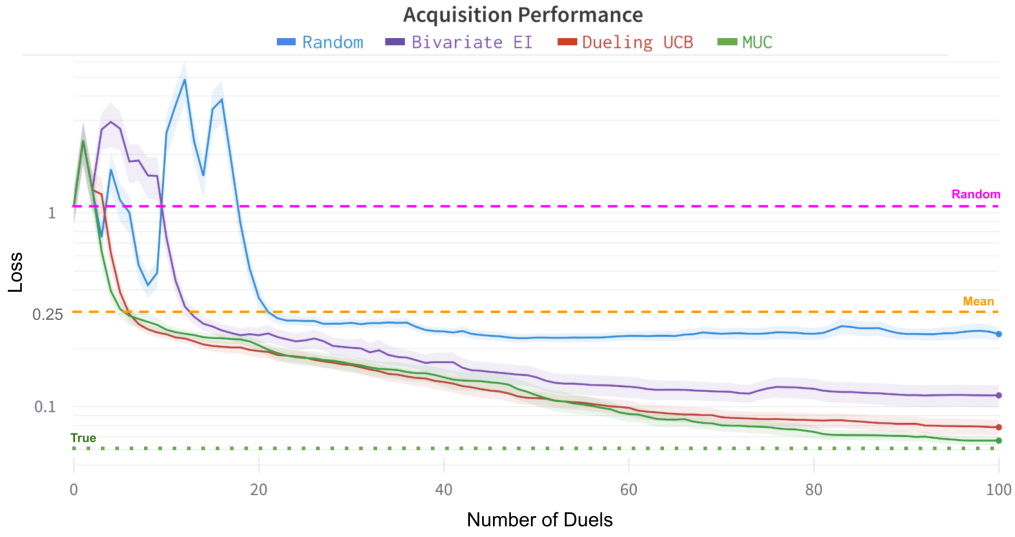

Figure C.4: Joint perceptual loss (y axis, log scale) plotted throughout HILO with different acquisition functions. Error bars denote SEM.

C.4 Batch Bayesian Optimization

While the proposed Bayesian optimization strategy of sequentially presenting the user with stimuli would likely not be prohibitively time consuming (about 17 minutes with 100 duels, 10 seconds per duel), it is possible that batch Bayesian optimization could speed up optimization. We consider two formulations of batch Bayesian optimizations: 1) the user selects their preference from a batch of stimuli in each iteration, and 2) each duel still has only two options, but batches of duels are precomputed to save on updated the Gaussian process posterior and the acquisition function time between duels.

Option 1 is theoretically more ideal, since more information acquired in each comparison would hopefully allow for fewer comparisons. However, this would require the patient to remember an entire batch of stimuli before making a comparison, and in practice phosphenes can be very difficult to discriminate [7]. It is difficult to evaluate the effect of this cognitive burden on simulated patients, and we thus leave it to future work to consider whether the parallelization of data acquisition would make up for the increased difficulty of the task.

Therefore, option 2 was tested by precomputing batches of 1 (*i.e. original*), 3, 6, or 10 duels at a time. We used a batch variant of the maximally-uncertain challenge acquisition function [42]. Results for 20 simulated patients are shown in Figure C.5. Precomputing batches reduced the optimization time required to reach a desired performance, but at the cost of requiring more duels. In practice, the optimal batch size could be determined by balancing the time required per duel with the time required for optimization. Faster acquisitions (*e.g.* KernelSelfSparring [65]) could further reduce optimization time.

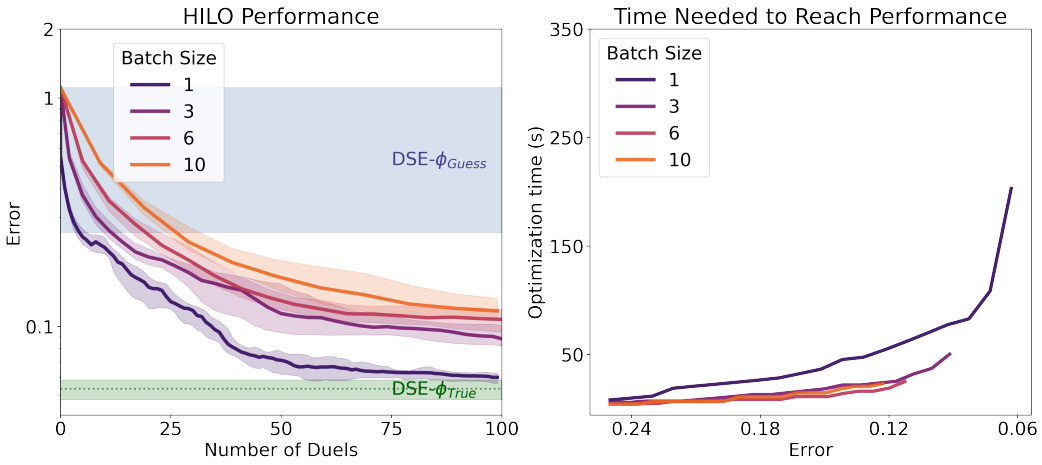

Figure C.5: Reconstruction error (left) and Bayesian optimization time required to reach a specified reconstruction error (right) for HILO with different batch sizes.

C.5 Baselines and Robustness

In this section, we provide further details for the naive encoder we compared against, and the robustness experiments.

Naive Encoder The naive encoder is the encoding strategy currently used in commercial epiretinal prostheses [3]. This encoder operates under the assumption that each electrode can be thought of as a pixel in an image. The optimal stimulus under this assumption is therefore simply a downsampled version of the target image. The frequency and pulse duration are constant across all electrodes. This naive encoder has been previously shown to be suboptimal [14], but we still include it as a comparison to the currently used encoding strategy.

Robustness In section 4.4 we evaluate the robustness of human-in-the-loop optimization (HILO) to misspecifications in the forward model. Here, we provide specific details on the implementation of these misspecifications, and how we adapted the baseline encoders to the misspecified scenarios.

- **Axon Trajectories**: The simulated axon map from [47] has two parameters, β_{inf} and β_{sup}, which control the axon trajectories in the inferior and superior retina, respectively. [47] also reports the observed ranges for these parameters: $\beta_{sup} \in [-2.5, -1.3]$ and $\beta_{inf} \in [0.1, 1.3]$. The unmodified model uses the centers of these ranges. Under misspecification, we randomly set both β_{sup} and β_{inf} to one of these bounds for each patient.
- **Thresholds**: Threshold is the amplitude at which a phosphene becomes visible to a patient $50\,\%$ of the time. In epiretinal prostheses, thresholds are notoriously noisy, and vary significantly across electrodes, patients, and over time [48]. While some progress has been made towards predicting these thresholds [66], most state-of-the-art models assume that thresholds are known.

  With this misspecification, the assumed threshold on each electrode was modified by a random but systematic amount. Specifically, the threshold on each electrode was randomly selected to be between $\frac{1}{2}$x and 2x its original value for the 100% condition and between $\frac{1}{4}$x and 4x its original value for the 300% condition.
- **Out of Distribution**: It is also possible that a new patient does not fall within our assumed ranges. Thus, we tested a variant where the true patient-specific parameters ϕ were sampled from outside the ranges in Table 1. Specifically, each parameter was sampled to be 0-50% above or below the specified range (some parameters were clipped to stay within defined ranges, *e.g.*, λ cannot be outside of $[0, 1)$).

  During HILO, the acquisition functions have specified bounds that constrain candidate ϕ. We therefore tested two variants, one where PBO was allowed to expand the bounds, and another where it was confined to within its original bounds. The end results were similar in terms of DSE performance, so the variant with its original bounds is presented in the main text.

DSE-ϕ_{Guess} is our best approximation of what a DSE would guessed patient-specific parameters would perform, and is bounded by the performance of a DSE with mean ϕ, and random ϕ from the ranges in Table 1. For each of the misspecifications, the mean and random ϕ baseline DSEs are still encoded with the same ϕ as in the unchanged patient, but since the phosphene model for the patient is changed, the resulting loss is different. DSE-ϕ_{True} is still parameterized with the patients true ϕ, however, the true ϕ are no longer a perfect description of the misspecified patient. This is shown by the fact that HILO surpasses the true encoders performance: under the misspecified model, there exists some other ϕ which leads to percepts with improved perceptual quality when decoded using the misspecified phosphene model, compared to those encoded with the true ϕ. This highlights the robustness of optimization based on user preferences.