**Simulated performance using model-based trajectory extrapolation (Kalman filter)**

A model-based strategy that determines bin choices in the trajectory extrapolation task would rely on statistical estimation of the trajectory parameters in making predictions about the object’s motion according to an internal dynamics model. We employ the Kalman filter, a standard algorithm used in modeling this trajectory estimation (e.g., 30-33).

Given a choice of polynomial (dynamics) model and available visual information, the Kalman filter recursively estimates the state of the moving dot (i.e., the trajectory parameters) at each time step using an internal representation:

 (S2)

The equation represents the linear extrapolation (prediction) from the internal representation of the object  at time *t-1* to the predicted state at the next moment in time, . The transition matrix *A* defines and represents the specific extrapolation strategybyencoding assumptions about the object’s dynamics process.

Here, we focus on local polynomial expansions of 2nd degree (i.e. locally quadratic). Locally quadratic extrapolation is defined by the *A* matrix:

 (S3)

which involves state estimates of position, velocity, and acceleration in x and y, respectively, and captures the curvature involved in the trajectory of an object undergoing constant acceleration in a gravitational field and has been previously found to underlie static contour extrapolation through occlusion[4-5].

The term  in Equation (S2) represents the process uncertainty (noise) with  with *Q* being a covariance matrix. The process noise represents the subject’s own estimate of model error.

At each time step, the current state, , and error covariance, , are projected to the next time step, via the *prediction* step, using the following formulas:

 (S4)

 (S5)

Then the *observation* of the true object state,  occurs, following Equation (S6):

 (S6)

The equation represents the linear relationship between the object’s true state,  and the observable parameters of that state (i.e., from the visible trajectory), defined by the matrix . We assume that subjects’ measurements are dominated by the velocity of the object’s path. As such, the *H* matrix is defined as:

**** (S7)

The observations are corrupted by sensory measurement uncertainty. The term  in Equation (S6) represents this sensory uncertainty with *p(v)~N(0,R*) with *R* being a covariance matrix. This term represents the perceptual sensitivity of the subject to the observable trajectory.

An estimate of the error between the prediction and the observation is computed, along with an estimate of the error covariance:

 (S8)

 (S9)

The error covariance is used to compute the Kalman gain, , which acts as a blending factor, that weights the model’s estimates against the observations:

 (S10)

The Kalman gain serves to minimize the *a posteriori* error covariance in the computations of the *a posteriori* state and covariance estimates,  and , respectively, using the following formulas:

 (S11)

 (S12)

Keeping in mind that the latter portion of the motion trajectory is occluded in our task, no measurements can be taken during this period. The prediction continues, however, using the process above, with the only omission being the  term from the state update equation (Eq. S11).

In modeling human performance, we provide the Kalman filter with x,y coordinates of the same trajectories that we provide the Q-learning agent (note that the precise parameters do not factor into the Q-learning agent’s performance). The y-coordinates are given a small amount of random Gaussian jitter to simulate sensory noise. The process and sensory measurement uncertainty (Eqns (S2) & (S6)) parameters are set to levels consistent with human performance in preliminary work. The Kalman filter runs through the prediction-observation cycle for the duration of the visible trajectory, and then continues its predictions through the duration of occlusion without feedback (observation). Once the Kalman filter has extrapolated the visible trajectory to the occluder’s edge, the process is terminated. We compute the polar angle of the extrapolated trajectory’s endpoint and determine which bin the trajectory is thus expected to reemerge within—this is recorded as the Kalman filter’s choice for that “trial”. Percent correct is once again computed as the percentage of correct choices (number of trials in which the choice bin = the “true” bin) in the 2000 trials.