Lee, Byungjoo; Kim, Sunjun; Oulasvirta, Antti; Lee, Jong-In; Park, Eunji

Moving Target Selection

Published in:

DOI:
10.1145/3173574.3173804

Published: 01/01/2018

Please cite the original version:
Moving Target Selection: A Cue Integration Model

Byungjoo Lee¹, Sunjun Kim¹,², Antti Oulasvirta², Jong-In Lee¹,², and Eunji Park¹
¹Graduate School of Culture Technology, KAIST  ²Aalto University, Finland
{byungjoo.lee, yi-jong-in, tracy1829}@kaist.ac.kr, {sunjun.kim, antti.oulasvirta}@aalto.fi

ABSTRACT
This paper investigates a common task requiring temporal precision: the selection of a rapidly moving target on display by invoking an input event when it is within some selection window. Previous work has explored the relationship between accuracy and precision in this task, but the role of visual cues available to users has remained unexplained. To expand modeling of timing performance to multimodal settings, common in gaming and music, our model builds on the principle of probabilistic cue integration. Maximum likelihood estimation (MLE) is used to model how different types of cues are integrated into a reliable estimate of the temporal task. The model deals with temporal structure (repetition, rhythm) and the perceivable movement of the target on display. It accurately predicts error rate in a range of realistic tasks. Applications include the optimization of difficulty in game-level design.

ACM Classification Keywords
H.5.m. Information Interfaces and Presentation (e.g. HCI) : Miscellaneous

Author Keywords
Temporal pointing; player modeling; moving target selection; level of difficulty; game balancing; cue integration.

INTRODUCTION
This paper presents novel data, analysis, and a model of an interactive task common in many popular games and music applications. In moving target selection, a displayed target moving rapidly must be selected with a brief input movement (such as a button pressing) while it is within some selection region. This is a special case of a general task called temporal pointing [23], and to press the button at the right time, the user must perceive information about both how long to wait to perform input execution (i.e., temporal distance) and how precisely the input should be performed immediately after the wait (i.e., temporal width). We are interested in this encoding and decision process. Unique to moving target selection, this process is expected to happen rapidly. Prior to our study, it was unknown how it is affected by the characteristics of the target (such as speed), the characteristics of the input device being used, and so on.

The model presented here contributes to the understanding on how to design tasks that require temporal precision. According to our survey,¹ for 23 of the top 100 free games in the Google Play market and 19 in the iOS App Store’s, selection of moving targets is an essential part of the game. Traditionally, the difficulty of levels is designed via heuristics [1, 3, 10, 11, 19, 21, 29]. Heuristics are effective for rapid design, but they are not predictive and are unsuitable for fine-tuning design parameters. For example, the HEP (Heuristics for Evaluating Playability) set [10] provides 43 heuristics. Several are related to the level of difficulty, but all are mute about how design choices are linked to player performance. Therefore, tuning of level parameters on the basis of heuristics requires iteration and costly empirical testing.

Our objective is a practical but accurate model that can be used to analyze moving target selection tasks and to predict the effects different design decisions bring to bear on user performance [16, 18]. Our model is based on the principles of probabilistic motor control. According to the cue integration theory [14], people’s performance in sensorimotor tasks is informed by priors expressing how likely different events are. The key to the theory is that such priors, learned over time, render it possible to make an estimate for very rapidly moving or ambiguous stimuli. We hypothesize that in the case of moving target selection, two types of cues are available to form such priors: (1) temporal structure (pace, rhythm, etc.) and (2) visually perceivable movement of the target toward the selection region.

Our model uses maximum likelihood estimation (MLE) to integrate information from each cue into one estimation of the target. Cue integration theory makes the assumption that people consider all information available to them optimally. The model thus yields an upper bound of performance achievable under favorable conditions. We present data suggesting that this optimality assumption provides surprisingly accurate predictions in realistic tasks. Also, being derived from a clear cognitive mechanism, our model has higher explanatory power than the black box models obtained by machine learning techniques [8].

To our knowledge, this is the first application of the cue integration theory in human–computer interaction (HCI). While cognitive psychology has timing tasks, such as moving target

¹Data retrieval date: June 27, 2017
Our study modeled the process by which a user perceives $W_t$ and $D_t$ in the important case of moving target selection. When the target is moving on a display, there are multiple cues that allow users to perceive $D_t$ and $W_t$. We describe the process by which such cues are perceptually encoded and integrated by the user. This makes it possible to represent more realistic HCI tasks that were not possible with previous models.

Studies of Temporal Tasks

Research on timing performance in psychology comes close to our topic. The most closely related tasks include reaction, moving target interception [35, 36], anticipation-coincidence [5, 24], and sensorimotor synchronization tasks [31, 32]. While these tasks are closely related to moving target selection, none can directly explain it.

First, in the reaction task, participants should respond as soon as possible to an unknown stimulus in time. The onset of the response cannot be anticipated, unlike in temporal pointing, because no prior information is given. Second, in the moving target interception task, participants must capture a fast-moving object through a hand or another end-effector. Intercepting a moving target requires gross spatial movement of an end-effector (hand or foot) [4, 9], which is outside the scope of temporal pointing. Also, interception is a very general idea that subsumes many instances [37]: pursuit, head-on, receding, and perpendicular. No single model exists for the general interception task. Finally, anticipation-coincidence and sensorimotor synchronization tasks deal with anticipatory timing performed by small movements such as tapping a finger with a metronome. Unlike temporal pointing, the task does not judge an input as success or failure. The stimulus is simply an impulse signal [12, 27, 31]. Some have studied stimuli with long durations, rather than impulse, but did not report or model users’ error rate [4, 27, 38, 40].

Although past studies did not directly model moving target selection, they provide many clues to understand it. In particular, the reliability of the temporal structure cue can be explained from the scalar property of the internal clock [17, 7]. The scalar property means that as the period of externally given temporal rhythm becomes longer, reliability of timing estimation decreases proportionally. Second, the drift-diffusion model [30], a model of reaction processes, can explain the precision of timing estimation obtained from the visually perceivable motion of the target. We can assume that the user accumulates and encodes the temporal information of the moving target within a limited cue viewing time ($t_c$).

Cue Integration Theory

Several signals are available for anticipating the moment of selecting a moving target. To explain how they are integrated into an actionable representation to guide response, we assume a probabilistic approach, which is natural in the face of stochasticity and noise. We use the cue integration theory to explain the different signals’ combination into a single estimate helping the user to decide when to press down [15, 13, 20]. Cue integration theory assumes that perception is a probabilistic process where each sensory channel conveys a cue about some property of the moving target. However, each
channel is unreliable to some degree. The integrated perception is a construct that represents a “decoded” sensation over all unreliable signals. We use maximum likelihood estimation to obtain a point estimate of the property being estimated. Assuming that a single-cue estimate is unbiased but corrupted by Gaussian noise, the optimal strategy for estimating the integrated perception \( \hat{\rho} \) is a weighted average [13, 15]:

\[
\hat{\rho} = \sum w_i \hat{\rho}_i \text{ where } w_i = \frac{1/\sigma^2_i}{\sum 1/\sigma^2_i} \tag{1}
\]

where \( w_i \) is the weight given to the \( i \)th single-cue estimate and \( \sigma^2_i \) is that estimate’s variance. If two sensations exist for one property, the variance of the integrated perception is \( \sigma^2 = \sigma^2_1 \sigma^2_2 / (\sigma^2_1 + \sigma^2_2) \).

**Overview and Derivation of the Model**

The model predicts error rate \( (E) \) in the moving target selection task. Cues are provided visually as a movement of an object (target), and a user must invoke a selection when the target is within some selection area. Selection error rate is defined as the ratio of the number of failed selections to the total number of selections \( (E = \frac{\# \text{ of failed selections}}{\# \text{ of total selections}}) \).

A key part of the model is about the mental processes of a user perceiving \( D_t \) and \( W_t \) from two different cues: (1) temporal structure and (2) visually perceivable movement of the target (see Figure 2):

- Perception of \( W_t \): A single target moving into one selection region conveys a unique \( W_t \) to the user; it is the duration from the moment the target enters the selection region to the moment it exits (see Figure 1).
- Perception of \( D_t \): \( D_t \) is the time from the user’s subjective now to the moment when the target first contacts the selection region. If the perception of \( D_t \) is accurate, the user can more accurately release a response when the target will contact the selection region.

From the perception of the task requirements (\( D_t \) and \( W_t \)), the user decides when to press the button. We assume that a user has an implicit aim point \( (\mu) \) somewhere between 0 and \( W_t \), where \( t = 0 \) is the moment when the target contacts the selection region for the first time. How precisely the user can perform response execution at this moment is determined by the following factors: (1) the reliability of the perception of \( W_t \) and \( D_t \), (2) the user’s timing ability, and (3) the precision of the user’s response execution. The effects of these three factors are aggregated into one input variance \( (\sigma^2) \) in the model.

The resulting distribution of a user’s responses is represented by the Gaussian distribution over time with mean \( \mu \) and variance \( \sigma^2 \). Then an error rate can be calculated as \( 1 - \int_0^{W_t} \mathcal{N}(\mu, \sigma^2) \). We will now describe these ideas in depth.

**Available Temporal Cues in Moving Target Selection**

Moving targets convey temporal requirements of the response via two main types of cues (see Figure 3). First, the motion of the target often has a specific temporal structure. Consider, for example, a gamer in a sniper role who is waiting for the enemy. If enemy soldiers appear and disappear in front of the sniper at certain intervals, the gamer will be able to predict the next time to fire the gun from the repeated pattern.

Second, visually perceivable movement of the target is available, as the target passes closer to and finally through the selection region. Imagine that a point object of speed \( s \) passes through a selection region of length \( l \) that is distance \( d \) away.

This means that the user must wait \( d/s \) seconds and then perform the input within the following \( l/s \) seconds (see Figure 5). When compared to the temporal structure of the stimuli, encoding of visually perceivable movement is more efficient but is indirect; the process is easily corrupted by noise, depending on the quality or the complexity of the given target movement.

Below we explain how \( W_t \) and \( D_t \) are perceived from each cue and how the mean and reliability of each perception stand out.
Assumptions: Perception
Suppose a property $A$ is perceived and estimated via sensory modality $X$. In this case, the mean value of the percept can be represented as $\hat{p}_X^A$, and the reliability of the perception can be expressed as standard deviation $\sigma_X^A$. Proceeding from previous studies of human timing perception [25, 26], we can assume that perceptions a user can obtain from two cues of a moving target are Gaussian distributions.

We assume that the user perceives only one temporal target – the same $W_t$ but from two or more, different cues. In cue integration theory, this assumption can be expressed as: $\hat{p}_X^W \approx \hat{p}_X^W$. The subscript $ts$ represents the temporal structure cue, and the subscript $vm$ represents the visually perceivable movement cue.

Note that unlike $W_t$, $D_t$ is defined from the user’s “subjective now,” so we cannot set a common “true” value for both cues (see Figure 4). Let $(D_t)_ts$ be the $D_t$ of the temporal structure cue and $(D_t)_vm$ be the $D_t$ of the visually perceivable movement cue. By encoding both $D_t$s from each cue, the user perceives when the target will contact the selection region for the first time. It is assumed that this moment is equally perceived from different cues ($\hat{p}_X^D \approx \hat{p}_X^D$). However, the reliability of the perceptions is assumed to be different for each cue: $\sigma_{ts}^D \neq \sigma_{vm}^D$.

The reliability of two perceptions $\sigma_{ts}^D$ and $\sigma_{vm}^D$ can be expressed as below, from known properties of cognition.

Reliability of $D_t$, Perceived from Temporal Structure Cue
The temporal structure of a moving target means how often a visually perceivable movement cue. For example, a metronome that beeps once every five seconds is harder to follow than a metronome that sounds once a second. This relationship between $P$ and $\sigma_t$ has been verified for a wide range of $P$ values from hundreds of milliseconds to several seconds [31].

In encoding of $D_t$ from the visually perceivable movement of the target, three variables shape the reliability of the perception: (1) the complexity of target movement, (2) the remaining distance between target and selection region, and (3) the time allowed to observe the target movement (cue viewing time $t_c$). Among these factors, we want to express reliability as a function of $t_c$ only. The remaining factors are included in the model through empirically determined free parameters.

Reliability of $D_t$, Perceived from Visual Movement Cue
In encoding of $D_t$ from the visually perceivable movement of the target, three variables shape the reliability of the perception: (1) the complexity of target movement, (2) the remaining distance between target and selection region, and (3) the time allowed to observe the target movement (cue viewing time $t_c$). Among these factors, we want to express reliability as a function of $t_c$ only. The remaining factors are included in the model through empirically determined free parameters.

In drift-diffusion models [30, 6], people are assumed to accumulate evidence from external cues over time. However, because the amount of information that one cue has is fixed, the reliability of the information that a person can obtain is bounded, no matter how long $t_c$ is. That is, given a sufficient amount of cue viewing time ($t_c \rightarrow \infty$), the reliability of the information that can be encoded from a visually perceivable movement cue converges to some specific maximum. In contrast, if the cue viewing time is very short ($t_c \rightarrow 0$), this information is very unreliable and the user’s inputs will (theoretically) have infinite variance. Understanding this mechanism, we can express the reliability of the encoded $D_t$ as an exponential function of the cue viewing time ($t_c$):

$$\sigma_{vm}^D = \frac{1}{e^{\nu t_c} - 1} + \delta$$

where $\nu$ is a coefficient that represents the drift rate of information and $\delta$ is a constant that represents the maximum reliability of the perception that can be achieved when $t_c$ is long enough. These parameters aggregate the effects of other factors of perceivable motion that affect the reliability of the perception, except $t_c$. Note also that, like temporal structure perception, Equation 3 represents only the quality of information the stimulus has, without regard to individual-specific timing capability differences.

Cue Integration
Through the process described above, the user obtains estimates of $W_t$ and $D_t$ from each cue. The mean and reliability of $W_t$ estimates are the same for the two cues, so they do not change after the integration process. In other words, a user’s
We can now compute error rates for a given temporal target.

\[
W_{\text{target}} (\hat{\rho}_D) = W_{\text{target}} (\hat{\rho}_D) = P_{\text{err}}
\]

However, the \( D \) perceived from each cue, or the moment when the target touches the selection region for the first time, has the same mean and different reliability. Through the cue integration process, the final perception has the following mean and reliability:

\[
\hat{\rho} = \hat{\rho}_D = \hat{\rho}_m \quad \text{and} \quad \sigma = \sigma_m = \sigma_m.
\]

User-Specific Factors

From the perceived \( W_t \) and \( D_t \), the user decides on a specific moment to perform the actual input. Due to variance and noise, the input response is distributed over time. The point at which the mean of the distribution is located is called the implicit aim point \( \mu \) of the user. The implicit aim point is expressed with a parameter \( c_\mu \) representing its ratio to the perceived temporal width:

\[
\mu = c_\mu \cdot \hat{\rho} \approx c_\mu W_t
\]

For example, if a user targets the first quarter of the temporal width, the \( c_\mu \) value will be 0.25. In this study, we assumed that the value of \( c_\mu \) indicating the implicit aim point of the user does not change. Several previous studies too have shown that \( c_\mu \) is constant for changes in \( W_t \) [23, 38].

By assuming that \( \sigma_W \) is negligibly smaller than \( \sigma_D \) because \( W_t \) is much shorter than \( D_t \), we determine the variance \( \sigma \) of the input distribution from both the reliability of perceived \( D_t \) and the timing precision of the user. We define a parameter \( c_\sigma \), which aggregates all user-side factors affecting the precision of response execution. This parameter takes into account both the noise in response execution and the effect of the input device. By multiplying \( \sigma_D \) by \( c_\sigma \), we get the standard deviation \( \sigma \) of the user’s input response:

\[
\sigma = c_\sigma \sigma_D
\]

When \( t = 0 \), the target contacts the selection region for the first time. The user’s inputs are represented by the Gaussian distribution over time \( \mathcal{N}(\mu, \sigma^2) \).

Error Rates

We can now compute error rates for a given temporal target. Integrating the input distribution of a user in the interval of temporal width \([0, W_t]\) yields the success rate, and subtracting this from 1 results in an error rate (see Figure 2).

The user input responses \( R \) are distributed as a Gaussian of mean \( \mu \) and variance \( \sigma^2 \) on the time axis:

\[
R(t|\mu, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(t-\mu)^2}{2\sigma^2}}
\]

The error rate is the value obtained by subtracting the area of the successful input from 1. Since the successful input represented in the distribution is from time 0 to \( W_t \), the error rate \( E \) can be expressed as:

\[
E = 1 - \int_0^{W_t} R(t) dt = 1 - \frac{1}{2} \left[ \text{erf}(W_t - \mu \sigma \sqrt{2}) + \text{erf}(\mu \sigma \sqrt{2}) \right]
\]

where \( \text{erf}(\cdot) \), known as the error function, appears in integrating a Gaussian distribution. By substitution from Equation 5, Equation 6, and Equation 7 for \( \sigma \) and \( \mu \), the above equation can be expressed as:

\[
E = 1 - \frac{1}{2} \left[ \text{erf}(\frac{1 - c_\mu}{c_\sigma \sqrt{2} W_t}) + \text{erf}(\frac{c_\mu}{c_\sigma \sqrt{2} W_t}) \right]
\]

where, for simplification, \( D_t \) is used in the denominator in line with this equation:

\[
D_t = P/\sqrt{1+(P/(1/(\nu c) - 1) + \delta)}^2
\]

Here, \( D_t \) is different from \( (D_t)_r \) or \( (D_t)_m \) and can be regarded as the effective temporal distance perceived by the user through the cue integration process. If \( t_c \) is 0, \( D_t \) is simply \( P \), and if \( t_c \) is very long, \( D_t \) converges to \( P/\sqrt{1+(P/\delta)^2} \).

By defining index of difficulty \( ID = \log_2(D_t/W_t) \), which is a single dimensionless variable governing this equation, we can further simplify Equation 10 as follows:

\[
E = 1 - \frac{1}{2} \left[ \text{erf}(\frac{1 - c_\mu}{c_\sigma \sqrt{2} ID}) + \text{erf}(\frac{c_\mu}{c_\sigma \sqrt{2} ID}) \right]
\]

The model can be used no matter the shape of the target, speed, movement pattern, or selection region shape.

PARAMETER EXPLORATION

The model has four free parameters \([\nu, \delta, c_\sigma, c_\mu]\). In this section, we explain how changes in each free parameter affect error rate.

Effect of varying \( c_\sigma \) on error rate: \( c_\sigma \) is a parameter representing the noise generated in the process of response execution. When \( c_\sigma \) becomes larger, the error rate of the user increases. Large changes in \( c_\sigma \) for a single user should not happen without changes in input device or user task [23]. In other words, if the input device or the task changes, this parameter would be re-estimated.

Effect of varying \( c_\mu \) on error rate: By setting \( c_\mu \) to 0.5, a user aims at the center of the target (\( W_t \)). That is the optimal aim point according to this model. However, users typically aim for the beginning of the target with \( c_\mu \) less than 0.5 (i.e., negative mean asynchrony [23, 31]). Because aiming strategy can change during a task, this parameter may change. That said, after learning, large changes are rare.

Effect of varying \( \nu \) on error rate: \( \nu \) is called the drift rate, which means the rate at which the user receives information
from the moving stimulus. Therefore, in a limited cue viewing time condition, a higher ν will lower the user’s error rate. For more continuous changes in t_c, this parameter determines the rate at which the user’s error rate converges to its minimum value (see Figure 6). This parameter is user- and possibly stimulus-specific. It would be re-estimated if the user group or stimulus type changes.

**Effect of varying δ on error rate:** δ is the maximum reliability of the perception a user can encode from a moving target as a result of a sufficiently long cue viewing time. It represents a lower bound on the user error rate. Although many hypotheses can be presented for how δ and ν are determined, our work assumes that the various visual characteristics of a particular moving target (complexity of motion trajectory, color of target, shape of background, etc.) are the main determinants limiting the amount of information the user can obtain.

**STUDY 1: MODEL VALIDATION**

The first experiment evaluated the overall fit of the model with empirical data in a closely controlled task. The experimental setup and apparatus used permit assessment of model predictions against three variables: (1) implicit aim point μ of participants, (2) standard deviation σ in input timing, and (3) error rate E.

**Method**

*Participants:* 18 paid participants (10 females) were recruited from a local university. The average age of participants was 24.8 years (SD = 3.4), and 15 had corrected vision. No participant was colorblind. The 17 participants with private music training averaged 5.36 (SD = 2.96) years of experience.

*Design:* The experiment was a 6 × 2 within-subject design with two independent variables: cue viewing time and target speed. The levels were the following:

- Cue viewing time (t_c): 0, 50, 100, 150, 200, and 250 ms
- Target speed: SLOW condition (1.162 m/s or 167 LEDs/s) and FAST condition (2.324 m/s or 333 LEDs/s)

**Task:** Participants were asked to perform a task of selecting a point-shaped target moving in a straight line (see Figure 7). We define cue viewing time (t_c) as the time to reach the selection region after target appearance (see Figure 7). Timing and error rates are measured according to the change of t_c. The temporal width and the period of the target’s appearance were fixed and unchanged: P = 2.000 ms and W_t = 100 ms. t_c was controlled by hiding a certain portion of the initial target movement. When t_c is zero, the task becomes a blinking target selection, and we can expect an error rate of about 70% from the given fixed P and W_t [23].

**Apparatus:** In order to minimize the effect of display latency, we implemented a customized high-speed 1D moving target selection experiment device. The apparatus consists of four components: an LED strip, an LED driver board, an experiment driver board, and a dummy mouse.

The LED strip consists of 3 meters of Adafruit DotStar module (144 LEDs/m). The center of the selection region is always placed at the 250th LED. At most, 117 LEDs (0.82 m) were used in the fastest and longest (333 LEDs/s, t_c = 250 ms + W_t = 100 ms) condition. The LED driver board (Arduino UNO board, Adafruit DotStar Library with hardware SPI) controls the LED strip to display the selection region, moving target, and success feedback (see Figure 7). The measured frame rate was 333.3 Hz; therefore, the achievable maximum speed without pixel skipping is 2.324 m/s (333 LEDs/s). We choose an opponent color pair (yellow–blue) to maximize the color contrast; a dark blue color (#000001) for the selection region and a moderate yellow color (#202000) for the moving target. For input device, we directly connected the left button of a dummy mouse to the experiment driver board. The mouse button was activated with 55 nN force at 0.4 mm displacement.

The experiment driver board (Arduino 101) sets parameters and records timestamps of input and LED events. Two serial pins and three digital pins interconnect the two driver

---

**Figure 6.** ν and δ are empirical parameters in the model. ν determines the speed at which the user encodes a motion cue. The δ is the minimum temporal distance that can be encoded from the motion cue. As shown in the upper row in the figure, the user’s error rate does not fall below a certain level even if the ν value increases. However, the bottom row shows that when the δ is small, user error may be reduced to near zero if t_c is sufficient.

**Figure 7.** In Study 1, participants performed a task of selecting a 1-dimensional moving target through a button click. The target moved from left to right at a specific speed and the participant was instructed to click the button when the target is in the blue selection region. At the moment of each button click, the target motion stopped and green visual feedback was given if the click was successful.
boards for serial communication and event timing synchronization. We collected the following events: a target appears (T1)-disappears (T0), a target enters (A1)-exits (A0) the selection region, and a button is pressed (B1)-released (B0). Timestamps were added internally to all events in 0.1 ms resolution. Event logs with the corresponding timestamps were transferred to a PC and stored. Hence, the event timings were rigorously maintained to <0.5 ms accuracy (mean error) and <1.5 ms precision (standard deviation).

**Setup and Procedure:** Participants sat on a regular office chair and looked at the front wall 2.8 meters away. On the wall, the LED device was installed horizontally at the eye level of the participants. We aligned the center of the selection region in front of participants. Then the experimenter briefly demonstrated the experiment and each participant was given a practice session until accustomed. Participants were asked to select as many targets as possible without skipping and instructed to take a break after completing one condition.

All task conditions were assigned randomly to the participants. There were 40 trials for one condition, and each condition was given twice. Therefore, participants performed, in all, 960 trials (40 trials × 12 conditions × 2 repeats). It took about an hour to finish. We logged all raw data of each participant. This includes not only the error rates of the input but also the timing of all the events.

**Results**

Of the 40 trials measured for one condition, the last 30 were included in analysis. Since each condition was repeated twice, the following results were analyzed from, in all, 60 trials for each condition. We ran statistical tests for each independent measure on the error rate, using two-way repeated-measure analysis of variance (RM-ANOVA). We applied Greenhouse-Geisser correction when the sphericity assumption was violated. We then compared the empirical µ, σ, and E with the model.

A successful trial means a B1 event positioned between A1 and A0.

**Effect of Target Speed**

The effect of target speed on error rate was not statistically significant (F(1,17)=2.998, p = .101). The average error rate was 0.36 (SD=0.259) for SLOW condition and 0.34 (SD=0.261) for FAST condition. The interaction effect between target speed and cue viewing time on error rate was significant (F(5,85)=3.082, p = .013). According to the pairwise t-test, the difference in error rate with speed was statistically significant at tc = 0.05 s (p = .021) and 0.25 s (p = .007). When tc was 0.05 or 0.25, the average error rate was 0.69 (SD=0.094) and 0.31 (SD=0.171) in the FAST condition and 0.718 (SD=0.113) and 0.259 (SD=0.188) in SLOW, respectively.

**Effect of Cue Viewing Time**

The effect of cue viewing time on error rate was statistically significant (F(2,12,36.0)=122.03, p < .001). As previously reported [23], and as we expected, the error rate was 70.5% (SD=10.3%) when the tc was zero. However, as tc increases to 0.25, the error rate decreases to 28.6% (SD=17.9%) with the aid of a visually perceivable movement cue (see Figure 8).

**Model Fitting**

**Testing Assumptions about σ and μ**

Since the main effect of target speed was not significant and our model can aggregate the effect of speed through v and δ parameters, the results presented in this subsection are averaged over two target speed levels (see Figure 8). All the curve fittings in Study 1 were carried out with the *fitnlm* function provided in MATLAB.

In the derivation process for the model, we assumed roughly that μ is always constant for the same Wt but the actual μ has changed from 0.2 to 0.6 as tc changes. The overall mean value of μ was 0.34 (SD=0.29). Particularly in sensorimotor synchronization studies, participants are known to aim mainly at the stimulus front [23, 31] (i.e., negative mean asynchrony). On the other hand, in our study, when tc was around 0.1 s, participants were starting to aim at the center of the selection region (see Figure 8).

The average value of σ was 0.064 (SD=0.051) and the model (Equation 7) had very accurate fit for σ values (R2 = 0.975; see Figure 8). The σ value is the integration result of σd and σt and from this high fit we showed that there is no problem with our two assumptions about σ: (1) diffusion-drift accounting for reliability of visual cues and (2) maximum likelihood integration of temporal and visual cues.

<table>
<thead>
<tr>
<th>Studies</th>
<th>cm</th>
<th>cm</th>
<th>v</th>
<th>δ</th>
<th>R²</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blinking (CHI'16)</td>
<td>0.169</td>
<td>0.080</td>
<td>N/A</td>
<td>N/A</td>
<td>0.997</td>
</tr>
<tr>
<td>Flappy Bird (CHI'16)</td>
<td>0.070</td>
<td>0.008</td>
<td>N/A</td>
<td>N/A</td>
<td>0.87</td>
</tr>
<tr>
<td>Validation Overall (this paper)</td>
<td>0.295</td>
<td>0.083</td>
<td>20.2</td>
<td>0.366</td>
<td>0.812</td>
</tr>
<tr>
<td>Validation Fast (this paper)</td>
<td>0.31</td>
<td>0.085</td>
<td>20.7</td>
<td>0.354</td>
<td>0.865</td>
</tr>
<tr>
<td>Validation Slow (this paper)</td>
<td>0.28</td>
<td>0.081</td>
<td>19.7</td>
<td>0.377</td>
<td>0.753</td>
</tr>
<tr>
<td>Flappy Bird (this paper)</td>
<td>0.118</td>
<td>0.0316</td>
<td>N/A</td>
<td>0.484</td>
<td>0.961</td>
</tr>
<tr>
<td>Cake Tower (this paper)</td>
<td>0.496</td>
<td>0.186</td>
<td>28.7</td>
<td>0.191</td>
<td>0.886</td>
</tr>
<tr>
<td>Cake Tower (CHI'16)</td>
<td>0.266</td>
<td>0.031</td>
<td>N/A</td>
<td>N/A</td>
<td>0.310</td>
</tr>
</tbody>
</table>

**Figure 8. Model validation for μ, σ (top row), and error rate (bottom row).** At the μ value measured at a specific tc (around 100 ms), a user reactive timing strategy that we did not expect was observed (σ increased to 0.6). However, that did not have much effect on the fit of the model to the error rate (R2 = 0.812). The model fit well with σ values (R2 = 0.975).
The model faithfully predicted the empirical error rates, which were obtained by using highly accurate experimental equipment with minimum system latency.

We also tested the assumptions made by the model, especially on the implicit aim point (μ) and the timing variance (σ) after the cue integration process. At a specific cue viewing time (around 100 ms), we observed a user reactive timing strategy that we did not expect. If \( t_c \) was similar to participants' reaction time, they tried to press the button as soon as the target appeared, instead of anticipating the time the target would reach the selection region via visual information. In that regard, some participants informed us that they selected the target via reaction rather than anticipation for some task conditions, and we suspect that \( t_c \) of 0.1 s is the case. This reactive behavior of a user is not covered by our model, which is something that game designers who use our model should be aware of. However, it did not have much effect on the robustness of the overall model \( (R^2 = 0.812) \).

The values we obtained for the four parameters of the model deserve a comment. First, the \( c_\sigma \) and \( c_\mu \) values were similar to those measured in past blinking target selection studies [23, 22]. This shows the robustness of these model parameters. The \( \nu \) value is measured as 20.2, and at this high drift rate, the user can accumulate all the information that the visually perceivable movement cue has, even if only a \( t_c \) of 0.3 seconds is given. Here \( \delta \) was measured to be 0.366, which is only 18% of target appearing period \( (P=2 \text{ s}) \). This means that the reliability of the visually perceivable movement cue is five times higher than that of the temporal structure cue when \( t_c \) is long enough.

**STUDY 2: REVISITING FLAPPY BIRD**

Flappy Bird is a game where a bird is repeatedly jumped to maintain its altitude and avoid the obstacles that follow. Flappy Bird can be regarded as a moving target selection task in which temporal structure cues and visual motion cues are given at the same time (see Figure 9). A player should repeat the input at specific intervals \( (P) \) to keep the bird at a certain altitude, which is a temporal structure cue to help estimate the timing for the next jump. The player also estimates the moment of the next jump from the trajectory of the bird, the visual motion cue, to the obstacle. One previous study [23] designed the jump period and temporal width of the game as a total of 12 \( P \times W_t \) combinations and let the participants play it (for the exact conditions, see the appendix of [23]).

The experimental method used for data collection is given by the original authors [23]. They used the model developed for the blinking target selection task to analyze the empirical error rate in Flappy Bird. Since the model does not consider visually perceivable motion cues, it achieved significantly worse fit \( (R^2 = 0.87) \) than a task of selecting a blinking target \( (R^2 = 0.997) \).

**Reanalysis with the Cue Integration Model**

We reanalyzed those empirical data [23] with the cue integration model. In Flappy Bird, the target and selection region can be observed by the player for a sufficiently long time, so \( t_c \) should be infinite (or very large). Then the exponential decay function in \( \sigma_{\nu,m} \) (Equation 3) becomes zero and only the three free parameters apart from \( \nu \) are significant. Fitting our model to the data with the \texttt{fitnlm} function provided by \texttt{MATLAB}, we obtained three parameters as in Table 1.

Our model showed a significantly better fit \( (R^2 = 0.961) \) than the previous model \( (R^2 = 0.87) \). The user-specific precision value \( c_\sigma \), which is measured lower (0.0316) than when there is only a temporal cue (typically around 0.08 [23, 22]), reflects the task becoming easier for the user as visually perceivable cues are added. This trend cannot be reflected in the previous model [23]. Instead, the lower error rate due to the addition of the visually perceivable cue is resulted as an unrealistic lower user precision value \( (c_\sigma = 0.008) \). On the other hand, the \( \delta \) was measured similarly to in Study 1 using our model. This lends more support to the model.

**STUDY 3: MODELING CAKE TOWER**

Cake Tower is a commercial game based on moving target selection that is being served to children in a digital activity space in Korea.\(^6\) Cake Tower can be played with just one button and the player’s goal is to stack the cake as high as possible. The way to stack a cake is simple in principle, but it gets harder and harder as the game advances. With a certain temporal pace, a new cake layer appears alternately from the left or right corner of the screen. The new layer is always the same size as the topmost layer of the tower the player has accumulated so far. The new layer flies at a constant speed towards the center of the screen, and the player must drop this layer on top of the cake tower. It is ideal to drop the layer completely on the top of the cake tower, but if it overlaps only partially, the non-covered part is cut off and the size of the layer that the player must stack next time will be reduced. If the player fails to accumulate any more cake, the game ends.

\(^6\)\url{https://envisible.squarespace.com/new-index-3}
and the height of the tower accumulated until then is the score (see Figure 10).

Method

Participants: 18 paid participants (7 female) were recruited from a local university. They were, on average, 27.6 years old (SD=5.88), and 8 had corrected vision. Their experience of receiving music training, excluding public education, averaged 3.26 years (SD=4.92).

Experimental Design: The experiment was a $2 \times 2 \times 3$ within-subject design with three independent variables: period of cake appearance, temporal width, and cue viewing time. The levels were the following:

- Period of cake appearance ($P$): 800 and 1,500 ms
- Temporal width ($W_d$): 80 and 160 ms
- Cue viewing time ($t_c$): 50, 150, and 250 ms

We maintained the initial size ($w$) of the cake throughout the experiment (165 pixels, 42.5 mm). The speed of the cake ($s$) was determined as $2w/W_d$ accordingly. The distance ($d$) from the corner of the screen to the center of the screen where the cake should be placed was $s \cdot t_c$, so the cake was only visible for $t_c$.

Apparatus: The application was implemented with Unity (version 5.6.2) on Mac OS X 10.12.4 and shown on a monitor (Samsung SyncMaster 205BW, measured avg. display lag = 8 ms) with 1680 x 1050 resolution. A MacBook Pro (Retina, 15-inch, early 2013) with 2.7 GHz Intel Core i7 and 16 GB RAM and NVIDIA GeForce GT 650M graphics card was used. Participants performed the task by clicking the left button of a Logitech G9 (1000 Hz) mouse fixed to the desk.

Procedure and Task: After the participants had filled out a demographic questionnaire form, the experimenter showed how to perform the task. When participants said that they understand the rules of the game, they started the experiment without any practice. At the beginning of each trial of the experiment, participants first observed four dummy cakes fly, which were generated with the corresponding period ($P$) value of the condition. This was to allow the temporal structure cue to be generated before actual cake stacking-up begins. Participants were asked to stack as much cake as possible.

The experiment was conducted in a quiet room with the participants sitting at a desk. They were encouraged to click the button with their preferred hand, but all were starting with their right hand. Twenty trials were conducted for each condition, and all conditions were given to each participant in random order. As a result, each participant performed, in total, 240 trials for an hour.

Logging: We logged all the raw data of the task. This includes not only the number of layers of cake but also the task situation when the participant clicked the button.

Monte Carlo Simulation

Cake Tower is a moving target selection task in which $W_d$ changes because the cake requires more precise timing as it gets higher. The model does not offer a closed-form equation predicting how many cake layers a player can stack, only the probability of failing for a given cake piece. We therefore simulate a virtual player with the model. It plays the game over and over again, allowing us to estimate the scores the user might obtain.

Player characteristics can be represented via model parameters $(c, \mu, \nu, \delta)$. Each player performs Gaussian inputs with a standard deviation $\sigma$ centered at $\mu$ for the given $D_i$ and $W_d$ of the current cake layer. This process is repeated several times, similar to the Monte Carlo method. The input generated from the random function is used to determine how much the next cake layer should be reduced in size, which is repeated until the virtual player fails to input. An estimate of a gamer’s score can thus be obtained.

Results and Discussion

The first 5 of 20 trials in each condition were excluded to take into account the learning effect. In the remaining 15 trials, the average height of the final cake tower was obtained. The overall average score of participants was 2.16 (SD=1.83).

Obtaining Model Parameters from Study 1

We first assume a virtual player from the four parameter values measured in Study 2 and compare the score for the simulated Cake Tower with the actual empirical data. The $v$ parameter, which was not measurable in Study 2, was replaced by the measurement in Study 1.

While it was a completely different game, the model could predict the Cake Tower scores reasonably well. Predictions show a high linear correlation with actual score data ($R^2 = 0.86$ (see Figure 11). This result shows that the model can be used to predict gamer performance across games when input device and user sample are relatively similar.

Model Fitting

We want to fit the player’s score predicted through Monte Carlo simulation and the score measured in the actual experiment against each ID condition. The simulation returns the expected cake tower height from the four given free parameters. Then, an error sum of squares between the returned values and the actual measurements is determined as an objective function and the optimal parameters are found through the patternsearch function provided by MATLAB. Pattern
search is an effective optimization technique for minimizing a stochastic objective function such as Monte Carlo simulation. The cue integration model shows a strong correlation with empirical data ($R^2 = 0.89$). With simulation via the previous model [23], model fit was considerably worse ($R^2 = 0.31$; see Figure 11). The previous model has difficulties due to the fact that visual timing estimation is more dominant in Cake Tower than in Flappy Bird. Indeed, the $\nu$ (28.7) and $\delta$ (0.191) measured through the cue integration model demonstrate a more effective visual motion cue in Cake Tower than in Study 1 or Study 2.

The $c_\mu$ value obtained (0.496) indicates that participants mainly aimed at the central point of the temporal target, unlike in previous studies (see Table 1). This is because subjects playing Cake Tower had to stack a target (i.e., flying cake layer) of the same size as the selection region, unlike in Study 1 and 2, where the moving target was assumed to be a point.

APPLICATION: DESIGNING GAME-LEVEL DIFFICULTY

Many studies have been conducted on how to set the appropriate difficulty level of a game [33, 2, 16, 41, 39, 28, 19, 1, 34]. Unlike previous work, the cue integration model offers an analytical–predictive method for resolving parameters like target speeds, cue viewing times, target sizes, and selection region. Three steps are required:

1. The designer first determines the time period ($P$) the player needs to repeat the input. In a music application, $P$ would be the tempo of the music. In the case of Flappy Bird, the time period of the jump, which must be repeated to keep the bird's altitude constant, was $P$.

2. The designer should then understand how much cue viewing time ($t_c$) is provided. The corresponding value $D_t$ can then be computed. At this point, the values of $\nu$ and $\delta$, as listed in Table 1, can be used:

   - $t_c$ is 0 s: The target suddenly appears in a selection region (e.g., a blinking target) $\rightarrow D_t = P$.
   - $t_c$ is from 0 to 0.3 s: The target appears and disappears quickly, or is often blocked by obstacles (e.g., a real-time shooting game or a baseball batting game) $\rightarrow D_t = P\left/\sqrt{1 + (P/(\nu t_c - 1) + \delta)^2}\right.$.
   - $t_c$ is long enough: The size of the target and activation region is small and the player can always observe the movement of the target (e.g., Flappy Bird or Dancing Line, and most temporal pointing games) $\rightarrow D_t = P/\sqrt{1 + (P/\delta)^2}$.

3. Finally, after $D_t$ is obtained, the designer can predict from Equation 10 what error rate players will have for a particular $W_t$ value. After finding of the $W_t$ value that produces the desired error rate, the size of the selection region and the movement speed of the target can be determined.

For example, in a rhythm game, such as Beatmania, let us assume that the designer wants to ramp up error rate to 30%, 50%, and 70% for easy, medium, and hard difficulty levels, respectively. If the BPM (Beats Per Minute) value of the given background music is about 150, the $P$ value is 0.4 seconds. If there is a design constraint that the $t_c$ value should be fixed at 0.1 seconds, $D_t$ becomes 0.3 s by Equation 11, using parameters borrowed from Study 1. Finally, from Equation 10, we can see that the required $W_t$ for the game levels is 58 ms, 35 ms, and 20 ms, respectively. On this basis, the designer can adjust the speed of the target or the size of the selection region.

CONCLUSION

This paper has presented a theory and mathematical model for predicting error rates in moving target selection tasks. The model is derived from a known cognitive theory called cue integration theory. To explain how users form an integrated percept of a rapidly moving target, the theory builds on a probabilistic account of sensorimotor performance. It estimates an upper bound of performance achievable by a perfect decoder. Across three studies, we have shown that this optimality assumption fits empirical data of trained users surprisingly well. This makes the model potentially useful in the design of gaming and music applications, where it can be used to design critical parameters like speeds, selection regions, and viewing times of moving targets.

Acknowledgments

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF), funded by the Ministry of Science and ICT (NRF-2017R1C1B2002101). The research also received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation program (grant agreement 637991).
REFERENCES


