Behavior Research Methods
2010, 42 (1), 226-241
doi:10.3758/BRM.42.1.226

MouseTracker: Software for studying real-time mental processing using a computer mouse-tracking method

Jonathan B. Freeman and Nalini Ambady
Tufts University, Medford, Massachusetts

In the present article, we present a software package, MouseTracker, that allows researchers to use a computer mouse-tracking method for assessing real-time processing in psychological tasks. By recording the streaming *x*-, *y*-coordinates of the computer mouse while participants move the mouse into one of multiple response alternatives, motor dynamics of the hand can reveal the time course of mental processes. MouseTracker provides researchers with fine-grained information about the real-time evolution of participant responses by sampling 60–75 times/sec the online competition between multiple response alternatives. MouseTracker allows researchers to develop and run experiments and subsequently analyze mouse trajectories in a user-interactive, graphics-based environment. Experiments may incorporate images, letter strings, and sounds. Mouse trajectories can be processed, averaged, visualized, and explored, and measures of spatial attraction/curvature, complexity, velocity, and acceleration can be computed. We describe the software and the method, and we provide details on mouse trajectory analysis. We validate the software by demonstrating the accuracy and reliability of its trajectory and reaction time data. The latest version of MouseTracker is freely available at http://mousetracker.jbfreeman.net.

In many domains across experimental psychology and cognitive science, it has proved difficult to assess how the cognitive system processes information in real time. Many of the phenomena in which researchers are interested occur quite rapidly, spanning only several hundreds of milliseconds. To understand how information processing unfolds across fractions of a second requires an online measure with substantial temporal resolution. Eyetracking and event-related brain potential (ERP) measures have been used, but these are expensive and often not suitable for many empirical questions motivating psychologists and cognitive scientists. Alternatively, which is often the case, researchers attempt to infer about processing by using outcome-based measures, such as reaction times (RTs) or error rates. These are inexpensive to record, needing only computer software, but can be limited in their ability to make inferences about what sort of perceptual-cognitive processing is occurring across time—especially how this processing evolves over time.

We believe that these methodological limitations have implications for our perspectives on mental phenomena and the development of theories and models to explain them. Because the fields of psychology and cognitive science have been somewhat unequipped to assess the real-time evolution of psychological responses with sufficient online measures, sometimes researchers have been left to keep the temporal dynamics of mental processes out of the theoretical and empirical spotlight. For instance, in the domain of categorization phenomena, there is now a wide proliferation of theories describing the static representation of category knowledge, but the temporal dynamics of processing leading up to categorical responses have gone virtually unexplored (see Dale, Kehoe, & Spivey, 2007). More broadly, in many niches across psychology and cognitive science, mental processes are often treated as proceeding in abstract discrete sequences rather than as being time dependent and temporally dynamic (Spivey, 2007; Spivey & Dale, 2004, 2006). Part of the story may be researchers' genuine subscription to discrete representational or symbolicist accounts of cognition (see, e.g., Dietrich & Markman, 2003; Newell, 1980; Pylyshyn, 1984), but some of this may be due to the simple fact that the methodological toolbox has not yet seen something practical, inexpensive, and time sensitive enough to flesh out the real-time dynamics of mental phenomena. Although offline methods have often been used to infer about these dynamics (e.g., priming, backward masking), online methods are scarce. In the present article, we will describe an online method to assess real-time mental processing and a graphics-based software package to let any psychologist or cognitive scientist easily harness the power of it.

Hand in Motion Reveals Mind in Motion

One possible way to tap into the real-time processing eventuating in the sorts of responses that psychologists

J. B. Freeman, jon.freeman@tufts.edu

and cognitive scientists care about—a categorization, the recognition of an emotion or spoken word, a lexical or evaluative decision, a social judgment—would be to look at participants' reaching arm movements as they make their way into settling into one of multiple response alternatives. Although motor responses are thought, according to a classical perspective, to be the end result of a feed-forward pipeline from perception → cognition → action (temporal cortex → "association cortex" → premotor areas), there is now abundant evidence that a motor response, such as the trajectory of a reaching arm movement, is continuously updated by perceptual-cognitive processing over time (e.g., Abrams & Balota, 1991; Gold & Shadlen, 2001; Song & Nakayama, 2006, 2008; Tipper, Howard, & Houghton, 1998). For example, as participants make their way toward grabbing an object that happens to move while the arm is still in motion, the trajectory of the arm is continuously adjusted in midair to seamlessly arrive at the object's new point in space (Goodale, Pelisson, & Prablanc, 1986). Similarly, as participants move a finger to point at a target, the finger's trajectory exhibits a temporally dynamic influence from subliminal primes that smoothly alters its curvature (Finkbeiner, Song, Nakayama, & Caramazza, 2008; Schmidt, 2002). These findings and others reviewed elsewhere (Spivey, Richardson, & Dale, 2008) have demonstrated that continuous motor responses, such as the reach of an arm, are not simply the endpoints of sensory and cognitive subsystems. Rather, the dynamics of action are part and parcel with the dynamics of perception and cognition. Thus, fortunately for us, online motor responses that are sampled fast enough may be informative as to the time course of perceptual-cognitive processing.

Computer Mouse-Tracking

Spivey, Grosjean, and Knoblich (2005) initially used online motor responses to understand real-time processing in the context of spoken word recognition by recording the streaming *x*-, *y*-coordinates of the computer mouse. Participants moved the mouse from the bottom center of the screen to either the top left or top right corners, corresponding to the recognition of two different words. For instance, a picture of a candle appeared in the top left corner, and a picture of a candy appeared in the top right corner while participants heard the spoken phrase, "Click the candle." When the distractor object corresponded to a word whose initial phoneme overlapped with the spoken word (e.g., *candy* for the word *candle*), but not when they did not overlap (e.g., *pickle* for the word *candle*), participants' mouse movements showed a conspicuous curvature toward the distractor (e.g., *candy*) before settling into the correct alternative (e.g., *candle*). As participants heard the word "candle," the ongoing accrual of "can . . ." partially activated lexical representations of both "candle" and "candy," causing mouse movements to gravitate somewhat toward "candy" before smoothly settling into "candle." Thus, the acoustic–phonetic input of the spoken word was taken up continuously over time rather than accrued in discrete steps. This had important implications for distinguishing between discrete stage-based and graded constraint-based accounts of spoken word recognition. Using this computer mouse-tracking method, such evidence for continuous temporal dynamics has been extended to other domains, such as semantic categorization (Dale et al., 2007) and syntactic ambiguity resolution (Farmer, Anderson, & Spivey, 2007).

Recently, we have used this mouse-tracking method to explore the temporal dynamics in real-time person perception. In one set of studies, participants were presented with typical and atypical male and female faces and were asked to categorize the target's sex using the computer mouse (Freeman, Ambady, Rule, & Johnson, 2008). The top left and top right corners displayed "Male" and "Female." Atypical faces included, for instance, a man's face that had been feminized using a morphing algorithm, or a woman's face containing a sex-atypical cue (short hair). When categorizing the sex of atypical faces (by moving the mouse from the bottom center to the top left or top right corners), participants' mouse trajectories were continuously attracted toward the opposite sex category before settling into the correct categorical response. For instance, when categorizing the sex of a masculinized female face (relative to a feminized female face), the mouse gravitated more toward the "Male" response before finally clicking on "Female." Thus, participants' social categorizations were the result of temporally dynamic competition, where partially active social category representations (male and female) simultaneously competed over time until gradually settling onto construals of others. We call this a *dynamic continuity account* of person construal (Freeman et al., 2008).

In a subsequent study, we used this method to show that this continuously evolving mixture of social category representations (male and female) during sex categorization may continuously cascade into the partial and parallel activation of associated stereotype knowledge (Freeman & Ambady, 2009). For instance, when deciding which of the two sex stereotypes (*caring*, a female stereotype, or *aggressive*, a male stereotype) better described a man whose face contained feminized cues, participants' mouse trajectories were continuously attracted to the caring alternative before settling into aggressive (Figure 1). We interpreted this as evidence for the partial and parallel activation of stereotypes tied to alternate social categories. Simultaneously and partially active stereotype knowledge belonging to multiple social categories was triggered across ongoing perceptual accrual of the face, and this fluctuating mixture settled over time onto ultimate judgments of others. A mouse-tracking paradigm has also been recently used to explore the time course of race categorization (Freeman, Pauker, Apfelbaum, & Ambady, 2010) and social attitude activation (Wojnowicz, Ferguson, Dale, & Spivey, 2009).

In short, computer mouse-tracking can afford researchers valuable information about the temporal dynamics of a variety of psychological processes. Indeed, reflecting on Spivey et al.'s (2005) aforementioned findings, Magnuson (2005, p. 9996) commented that "the continuous data provided by the mouse-tracking technique . . . [have] the potential to address not only specialized theoretical debates but also some of the biggest questions facing cog-

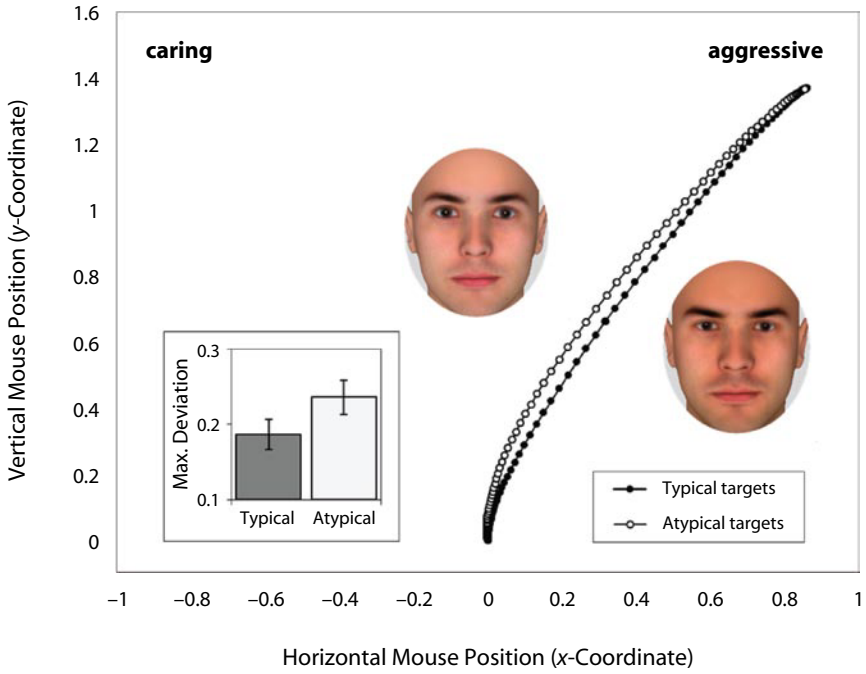

Figure 1. In a prior study, Freeman and Ambady (2009) found that when stereotyping sex-atypical faces (e.g., a man with feminized cues), participants' mouse trajectories were continuously attracted to the stereotype of the opposite sex (e.g., *caring*) before settling into their ultimate response (e.g., *aggressive*). As is indexed by maximum deviation (bar graph), the attraction of trajectories for sex-atypical faces to the opposite sex stereotype was statistically significant. From "Motions of the Hand Expose the Partial and Parallel Activation of Stereotypes," by J. B. Freeman and N. Ambady, 2009, *Psychological Science*, *20*, p. 1185. Copyright 2009 by Wiley-Blackwell Publishing. Adapted with permission.

nitive science." The current methodological standard has been the RT (the moment the mouse clicks on a response). When only RTs are used, however, fine-grained temporal information about how a participant's response evolves over time can be lost. Using mouse-tracking, the real-time development of a participant's ultimate response can be sampled approximately 60–75 times every second (60–75 Hz). Current tasks that measure RTs could easily be tweaked for recording continuous streams of motor output that culminate in those same responses. In addition, novel tasks could be designed to collect these motor streams for answering new empirical questions and endeavoring first interrogations into the real-time dynamics of mental activity in a variety of domains. Below, we will describe a software package that the first author has developed to allow researchers to do this in a comprehensive and graphics-based manner.

MOUSETRACKER

MouseTracker is a software package that allows researchers to measure real-time hand movements from the streaming *x*-, *y*-coordinates of the computer mouse (while behavioral responses are made on the basis of images, letter strings, and/or sounds), and subsequently visualize, process, and analyze them (as was done in the aforementioned studies). It can be installed on computers running Windows XP/Vista/7. For downloading and other information, visit http://mousetracker.jbfreeman.net. It is copyrighted by the first author and protected by an end-user license agreement. The agreement grants users a perpetual and nonexclusive license to use the software, but requires that the users do not modify, reproduce, license, or sublicense the software. It contains three programs: A data collection program (*Runner*) allows researchers to specify stimuli files, timing, and response options, among other parameters, and run participants through studies. A graphics-based program (*Designer*) may be used to set up the visual layout and response options of an experiment. Finally, an analysis program (*Analyzer*) can then import participants' data from such studies and visualize, process, and analyze the mouse movements.

MouseTracker is able to handle many different kinds of multiple-choice decision tasks that are uniquely customized by the researcher. In terms of analysis, the program can handle 1 individual participant's data or aggregate across a whole group of participants' data at the same time. MouseTracker automatically performs space-rescaling on mouse trajectories. It can analyze trajectory data either in normalized time or in raw time. It groups mouse trajectories by specified conditions and visualizes all trajectories within specific conditions for side-by-side visual comparisons. Trajectories can be explored and individually selected for detailed information or exclusion. Mouse-

Tracker generates mean trajectories of conditions and computes by-trial indices of spatial attraction/curvature and complexity. It also conveniently *z*-normalizes these (pooling both across and within conditions) for distributional analyses. If trajectories are analyzed in raw time, MouseTracker can also generate velocity and acceleration profiles. All of these data are then able to be exported into a comma-separated-value (.CSV) file for analyzing in Excel or other software, prepared in a way that is ready for hypothesis testing and further analysis.

Overall Structure

A researcher designs an experiment using a .CSV file in Excel (or some other program), which specifies display and experimental parameters and lists stimuli and responses. The Designer program can be used to edit the .CSV file's display parameters and response options in a graphics-based manner. In all experiments, several things remain constant. On each trial, a "Start" button appears at a designated location on the screen. After a participant presses this button, the trial begins. The participant then moves the mouse from the "Start" button to one of the response alternatives. The "Start" button and response alternatives may be placed anywhere on the screen. Once an experiment .CSV file has been designed, it can be executed by the Runner program, which will run the experiment and record mouse trajectories. Once the participant finishes, mouse movement data undergo several automatic transformations and are subsequently outputted to a MouseTracker data file (.MT). This .MT data file is then loaded into the Analyzer program for a graphics-based analysis of the recorded mouse trajectories. This .MT data file is inherently a .CSV file, which can be manually read in Excel. It contains trial-by-trial information about raw *x*-, *y*-coordinates and other information about transformed data, in addition to standard information about RTs (the moment the mouse clicks on a response), initiation times (the moment the mouse is first moved), and stimuli, among others. Once the .MT file, or multiple .MT files, are loaded into Analyzer, trajectories can be visually explored, averaged, and manually or systematically excluded, and indices of spatial attraction/curvature and complexity may be computed. If trajectories are kept in raw time (rather than in normalized time), velocity and acceleration profiles are also available. In every particular analysis, two conditions (Condition 1 and Condition 2) are examined side by side and compared along with their mean trajectories. Time course data are also available. Finalized data are then exported to an output .CSV file, readable in Excel, which contains all of the data prepared in a way suitable for hypothesis testing and further analysis.

Installation

MouseTracker is installed using the self-extracting executable file obtained from the download Web site: http://mousetracker.jbfreeman.net. Once the user agrees to the end-user license agreement and the installation is complete, several shortcuts are created in the Start Menu: a shortcut to the Runner program, a shortcut to the Designer program, a shortcut to the Analyzer program, a shortcut to complete the free registration, a shortcut to the help file, and a shortcut to the license agreement. The free registration screen appears after the installation has completed, but this can also be launched using the Start Menu shortcut. Users must complete the free registration before proceeding to use the Runner, Designer, and Analyzer software programs.

What Are Mouse Trajectory Data and How Are They Analyzed?

During every trial, the Runner program records the real-time *x*-, *y*-coordinates of the computer mouse. Because the sampling rate is approximately 60–75 Hz (see the Influences of Hardware section for a discussion of the rate variability), about every 13–16 msec during the live portion of a trial, three pieces of information are recorded: raw time (how many milliseconds have elapsed), the *x*-coordinate of the mouse (in pixels), and the *y*-coordinate of the mouse (in pixels). Because these mouse trajectory data are rich—containing between 60 and 75 *x*-, *y*-coordinate pairs every second—there are many ways to carve up the data, various measures that could be computed, and several temporal and spatial scales of possible analysis. Below, we will describe standard analytic approaches, which have been consistently used in prior mouse-tracking studies and are fully implemented in MouseTracker. However, there are an infinite number of ways in which one could analyze mouse trajectories, and a MouseTracker user could conduct these externally using the outputted data.

Space rescaling. First, all trajectories are rescaled into a standard MouseTracker coordinate space, which we have used in prior work (Freeman & Ambady, 2009; Freeman et al., 2008; Freeman et al., 2010). A diagram of this coordinate space appears in Figure 2. The top left corner of the screen corresponds to [−1.00, 1.50], and the bottom right corner corresponds to [1.00, 0.00], leaving the start location of the mouse (the bottom center) with the coordinates [0.00, 0.00]. This standard space thus represents a

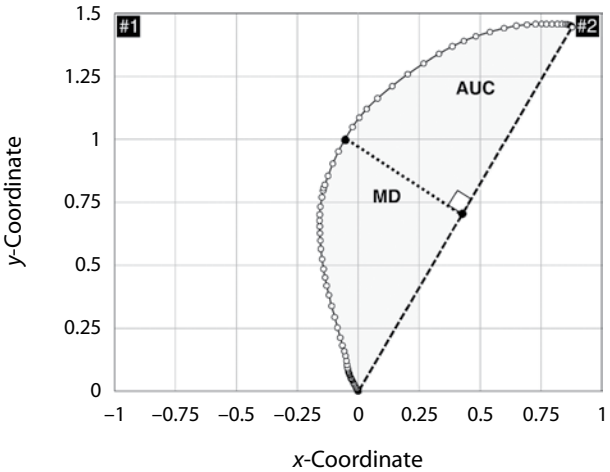

Figure 2. Diagram of the MouseTracker standard coordinate space and the calculation of measures of spatial attraction to the opposite response alternative: maximum deviation (MD) and area under the curve (AUC).

2 × 1.5 rectangle, which retains the aspect ratio of most computer screens. These space-rescaled trajectories may then be submitted to a time-normalization procedure (or retained in raw time).

Time normalization. Time normalization is often conducted because each recorded trajectory tends to have a different length. For instance, a trial lasting 800 msec will contain approximately 56 *x*-, *y*-coordinate pairs, but a trial lasting 1,600 msec will contain approximately 112 *x*-, *y*-coordinate pairs (at 70 Hz). To permit averaging and comparison across multiple trials with different numbers of coordinate pairs, the *x*-, *y*-coordinates of each trajectory are time normalized into a certain number of time steps using linear interpolation (e.g., by default, 101, to permit 100 time-normalized equal spaces between coordinate samples). Thus, the 56 coordinate pairs from the 800-msec trial would be fit to 101 pairs, just as the 112 pairs from the 1,200-msec trial would be fit to 101 pairs. Thus, each trajectory is normalized to have the same number of time steps (e.g., 101), and each time step has a corresponding *x*- and *y*-coordinate.

Averaging. Each participant's mean trajectory for one condition is computed by averaging together all of the *x*-coordinates of trajectories in that condition at each time step, and all of the *y*-coordinates of trajectories in that condition at each time step. If data from multiple participants are being averaged together, the grand mean trajectories are computed by averaging together each participant's mean trajectories.

Measuring spatial attraction. These preprocessed and averaged mouse trajectory data could be used in many ways that depend on the research questions at hand. In many cases, one question is whether the trajectories for one condition travel closer to an unselected alternative relative to those in another condition. For instance, in the aforementioned studies (Freeman et al., 2008), we asked: When categorizing the sex of a masculinized woman's face, are trajectories more attracted to the male category (on the opposite side of the screen) than they are when categorizing a feminized woman's face? One approach would be to use 101 paired-samples *t* tests to compare the *x*-coordinate of participants' mean trajectories for Condition 1 versus Condition 2 at each individual time step. However, the issue of multiple comparisons arises (although a bootstrapping method has been used; see Dale et al., 2007; Freeman et al., 2008) and, in many cases, researchers would like to obtain a simpler by-trial index of the amount of attraction toward the unselected alternative.

Prior studies have used two measures, which are fully implemented in the MouseTracker: maximum deviation (MD) and area under the curve (AUC). For both of these measures, MouseTracker first computes an idealized response trajectory (a straight line between each trajectory's start and endpoints). The MD of a trajectory is then calculated as the largest perpendicular deviation between the actual trajectory and its idealized trajectory out of all time steps. Thus, the higher the MD, the more the trajectory deviated toward the unselected alternative. The AUC of a trajectory is calculated as the geometric area between the actual trajectory and the idealized trajectory (straight line). The area on the opposite side of the straight line is calculated as negative area. Figure 2 provides an illustration of how the MD and the AUC measures are calculated. An example trajectory is displayed in the figure, which heads rightward to the selected response (#2). The amount of spatial attraction toward an unselected response (#1) will be calculated. Also displayed is an idealized response trajectory (straight line) that is used to calculate the MD and the AUC. In prior studies, we have found that using the MD versus the AUC for the same data does not substantially change the results (Freeman et al., 2008). One observation, however, is that the AUC is a better index of the overall attraction toward the unselected alternative (incorporating all time steps), whereas MD is a better index of maximum attraction, but this attraction may be limited to fewer time steps.

Measuring complexity. Complexity arises in the behaviors of many dynamic biological systems, including the human brain (Spivey, 2007). In some cases, it may be helpful to measure the complexity of mouse trajectories. For example, if both response alternatives simultaneously attract participants' mouse trajectories (relative to only one), this additional stress might manifest as less smooth, more complex, and fluctuating trajectories. Some mouse-tracking studies have used sample entropy to index complexity (e.g., Dale et al., 2007; McKinstry, Dale, & Spivey, 2008), and some have used "*x*-flips" (Dale, Roche, Snyder, & McCall, 2008). MouseTracker calculates *x*-flips and *y*-flips, which are the number of reversals of direction along the respective axis. This captures the fluctuations in the hand's vacillation along the horizontal and vertical axes.

Distributional analyses. Some researchers may wish to examine the distribution of trajectories' trial-by-trial spatial attractions toward an unselected alternative (indexed by MD or AUC). This can be especially useful for formally determining the temporal nature of one condition's stronger attraction toward an unselected alternative relative to that in another condition. For instance, suppose that a researcher finds that trajectories in Condition 1 are continuously more attracted toward the unselected alternative than are trajectories in Condition 2. This is visually apparent by plotting the two mean trajectories and is statistically apparent by a significant difference in MD or AUC between Condition 1 and Condition 2. Underlying this reliable continuous attraction effect, however, could be a subpopulation of discrete-like errors biasing the results. For instance, if half of the trajectories in Condition 1 headed straight to the selected alternative, and the other half initially headed straight to the unselected alternative, followed by a sharp midflight correction redirecting the trajectory toward the selected alternative, the mean trajectories would exhibit a reliable attraction effect that appeared continuous, although it was actually caused by several discrete-like errors. If such a subpopulation of discrete-like errors were biasing the results, the distribution of Condition 1 would be bimodal (some trajectories show zero attraction, and the other trajectories show extreme attraction).

In prior work, we have empirically validated that the mouse-tracking paradigm afforded by MouseTracker and the computation of the bimodality coefficient (*b*; SAS In-

stitute, 1989) have sufficient methodological and statistical sensitivity to detect bimodality in such a pattern of trajectories (Freeman et al., 2008, Study 3). In that study, we asked participants to make simple categorizations using the mouse-tracking paradigm. On half of the trials, trials proceeded normally and trajectories headed straight to the response (control trials). On the other half of the trials, however, once mouse movement was initiated, the response labels suddenly turned red and switched sides (switch trials). This required participants to generate extreme trajectories appearing as discrete-like errors, where the direction had to be discontinuously reversed because of the responses' switching sides. Figure 3 depicts the mean trajectory for control trials, the mean trajectories for switch trials, and a "combination" trajectory representing an average across both switch and control trials (all trajectories are remapped rightward). This combination trajectory showed an attraction effect toward the unselected response (in the figure, the top left corner) that appeared continuous, although it was actually caused by some trajectories in the form of discrete-like errors and other trajectories showing zero attraction. This demonstrated how one population of trajectories showing zero attraction and another population of trajectories showing extreme attraction can spuriously produce continuous attraction effects. As is shown in the histogram (Figure 3), we reliably detected bimodality in the distribution of the AUC values in this combination condition (averaging across control and switch trials). This validated that the mouse-tracking paradigm and the computation of the *b* coefficient are sensitive enough to identify this spurious pattern.

Bimodality may be tested by determining whether $b > 0.555$. If $b > 0.555$, the distribution is considered to be bimodal, and if $b \leq 0.555$, it is considered to be unimodal. It is computed by the following equation:

$$b = \frac{g_1^2 + 1}{g_2 + \frac{3(n-1)^2}{(n-2)(n-3)}}.$$

In the equation, g_1 is skewness, g_2 is kurtosis, and n is the number of observations. For testing bimodality, MouseTracker *z*-normalizes the MD and the AUC values of trials within each participant, together across Condition 1 and Condition 2, for convenience. In prior work, mouse-tracking researchers have further alleviated concerns about latent bimodality by ensuring that the shapes of the two distributions are statistically indistinguishable. To accomplish this, the Kolmogorov–Smirnov test is used (available on the Web: www.physics.csbsju.edu/stats/KS-test

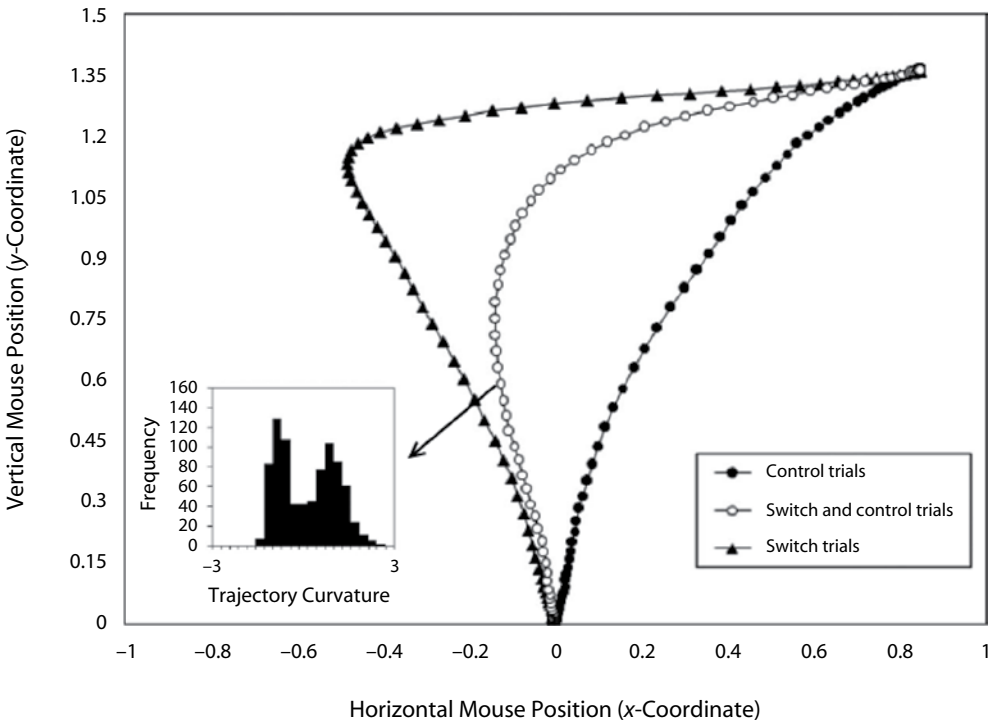

Figure 3. In a prior study, Freeman et al. (2008) found that averaging together across half of trajectories moving straight to the correct response (control trials) and half of trajectories making discrete-like errors (switch trials) spuriously produces continuous attraction. As is shown in the histogram, the distribution of AUC values in this combination condition including switch and control trials was bimodal ($b > .555$). This validated that the mouse-tracking paradigm and computation of the *b* coefficient are sensitive enough to identify this spurious pattern. From "Will a Category Cue Attract You? Motor Output Reveals Dynamic Competition Across Person Construal," by J. B. Freeman, N. Ambady, N. O. Rule, and K. L. Johnson, 2008, *Journal of Experimental Psychology: General*, *137*, p. 686. Copyright 2008 by the American Psychological Association. Adapted with permission.

.html). For use with this test, MouseTracker conveniently *z*-normalizes the MD and the AUC values of trials within each participant, separately within Condition 1 and Condition 2. This test, unlike the bimodality test, is inferential; if $p < .05$, the shapes of the two distributions reliably depart from one another.

Raw time analysis, velocity, and acceleration. A user can opt to retain trajectories in raw time (without time normalization). If a raw time analysis is conducted, the user decides how many raw time bins to create between 0 msec and some cutoff (e.g., 1,500 msec). As with a normalized time analysis, trajectories are space rescaled. However, rather than generating a certain number of normalized time steps, MouseTracker generates a user-defined number of raw time steps. Trajectories are visualized and averaged, as in a normalized time analysis, but measures of spatial attraction/curvature and complexity (MD, AUC, *x*-flips, *y*-flips) are not available. Instead, velocity and acceleration profiles are generated for each trajectory, and these profiles are averaged across all trials within a participant, separately for Condition 1 and Condition 2. They are also averaged across all participants, separately for Condition 1 and Condition 2. This information can be plotted in the Analyzer program or viewed in the output .CSV file.

Raw time analyses can be important in a variety of scenarios. For instance, if participants are presented with an auditory stimulus of spoken words, a researcher might be interested to know the mouse's behavior once a critical word is spoken (e.g., 400 msec after the beginning of a trial). Moreover, a researcher might simply want to examine for differences in mouse trajectories during a period of raw time (e.g., 700–1,000 msec). A raw time analysis also permits the calculation of velocity and acceleration.

Influences of Hardware

MouseTracker runs on any standard Windows XP/Vista/7 computer. There do not appear to be any major hardware constraints. Across a wide array of computers with various hardware specifications, we have found that MouseTracker's sampling rate of the mouse location does not fall outside a range of 60 to 75 Hz. MouseTracker attempts to sample the mouse location as frequently as possible, but because the sampling rate is constrained by the processor's clock speed (and other factors related to the processor's architecture), there is some variability in how frequently the mouse may be sampled on different computers (60–75 Hz).

We assume that most researchers will conduct MouseTracker experiments with a standard computer mouse, but any pointing device that is supported in Windows is compatible with MouseTracker (e.g., touchpad, trackball, joystick, pointing stick, graphics tablet, touchscreen, Nintendo Wii remote). However, different types of pointing devices provide different types of motoric experiences that could influence the nature and quality of MouseTracker data. For instance, touchpads (supplied with almost every modern laptop computer) make use of the dragging motions of the finger to control the location of the mouse cursor. Performance is usually impaired in computer pointing tasks when a touchpad is used rather than a mouse (Akamatsu & MacKenzie, 2002). Because the touchpad physically spans only a small area, tiny motions of the finger applied to the pad elicit much larger motions of the cursor on the screen, which leads to a more ballistic style of cursor movement. Because of the common occurrence of these larger, more ballistic movements in touchpad use, participants must unnaturally decrease the level of force applied to the touchpad during the final phase of trying to land the cursor on a target on screen (Akamatsu & MacKenzie, 2002). The result is less precise, jerkier cursor movement with more errors. Thus, different motoric experiences of pointing devices beyond a standard mouse, such as a touchpad, are likely to affect the trajectories that MouseTracker can capture in terms of accuracy, precision, ballistic quality, and perhaps quantization effects.

Display and pointer device settings that could influence MouseTracker data include the speed of cursor movement and screen resolution, which are set in the Windows control panel. MouseTracker is compatible with any cursor speed and any screen resolution (provided that the resolution is supported by the display adapter). Higher screen resolutions create the opportunity for an experiment to have larger distances that the mouse must traverse to reach a target. Faster cursor speeds allow such distances to be traveled more quickly. At extremely high speeds, however, one can imagine that cursor movement becomes ballistic and jerky, which could potentially dampen the sensitivity of MouseTracker trajectory data.

Designing Experiments

Experiments are designed using a .CSV file. An example and tutorial are provided with the software. There are two parts to an experiment .CSV file. In the first part, the user sets experimental and display parameters. In the second part, the user lists what stimuli are presented and what response options are given.

Display and experimental parameters. Various display and experimental parameters can be customized. These include: response-button width, height, location, and color; stimulus location; stimulus font style, size, and color (if using letter strings, e.g., words); whether to automatically relocate the mouse to the bottom center portion of the screen after the "Start" button is pushed; initiation time screening (whether to alert participants if they initiate movement too late); *x*-coordinate screening (whether to have participants rerun on trials in which they make extreme jolting movements toward the incorrect response alternative); RT deadlines; feedback on errors; intertrial interval durations; whether to mark a trial wrong when the mouse clicks on the incorrect response or hovers over it; and whether to freeze the mouse or let it move freely during compound trials when multiple stimuli are sequenced. The Designer program can be used to edit the display parameters of the .CSV file in a graphics-based manner. To load Designer: Start Menu → Programs → MouseTracker → Designer.

Stimuli and response list. In the experiment .CSV file, stimuli and responses are entered into as many rows

as are needed. There are four types of trials that MouseTracker allows: nonexperimental trials (e.g., instructions or break) in which a fullscreen image is displayed, trials involving an image stimulus, trials involving a letter string stimulus (e.g., a word), trials involving an auditory stimulus, and compound trials involving multiple stimuli presented in a sequence and/or simultaneously. (Compound trials can involve an unlimited number of images, letter strings, sounds, or a mix of these.) For each trial, inputted in a single row, the following is specified: the stimulus or sequence of stimuli, the various response alternatives (these can be either a letter string or an image), which response alternative is correct, which response alternative is the default comparison (e.g., hypothesized to attract participants' mouse movements), and a specific condition code. The condition code defines some meaningful code that can be used to select this trial in the Analyzer program. This can be general (e.g., "typical" or "atypical") or very specific unique codes (e.g., "200" or "red_circle_5") that can then be grouped into a meaningful condition in the Analyzer program.

Running Experiments

To run experiments, the user opens Runner (Start Menu → Programs → MouseTracker → Runner). It prompts the user to select an experiment .CSV file. Runner then executes this file and, once finished, outputs a data file carrying the extension .MT. This .MT file may then be loaded into the Analyzer software for visualization and analysis, which will be described next.

Analyzing Mouse Trajectory Data

To begin mouse trajectory visualization and data analysis, the user launches the Analyzer program (Start Menu → Programs → MouseTracker → Analyzer). This loads a user-interactive, graphics-based environment. An example screenshot appears in Figure 4. Analyzer allows the user to define two meaningful conditions for a particular analysis (Condition 1 and Condition 2), each of which can include many individual condition codes (as are specified in the experiment .CSV file). Trajectory data from all trials corresponding to these codes are then able to be imported into the program for visualization and analysis.

Loading data. To load data, the user clicks the "Load File" button to load one .MT file, or the "Load Folder" button to load multiple .MT files. Once loaded, the file(s) appear in the list on the left portion of the screen. The user is then prompted "Load .LOG files (if any)?" .LOG files are optional files tied to .MT files that contain information about which trajectories in a participant's data should be excluded from analysis. These .LOG files are generated using the "Exclude" and "Save" buttons (detailed later). Clicking "Yes" excludes trajectories that have been previously excluded and saved; clicking "No" disregards .LOG files (and thus, previous exclusions that were saved are not loaded).

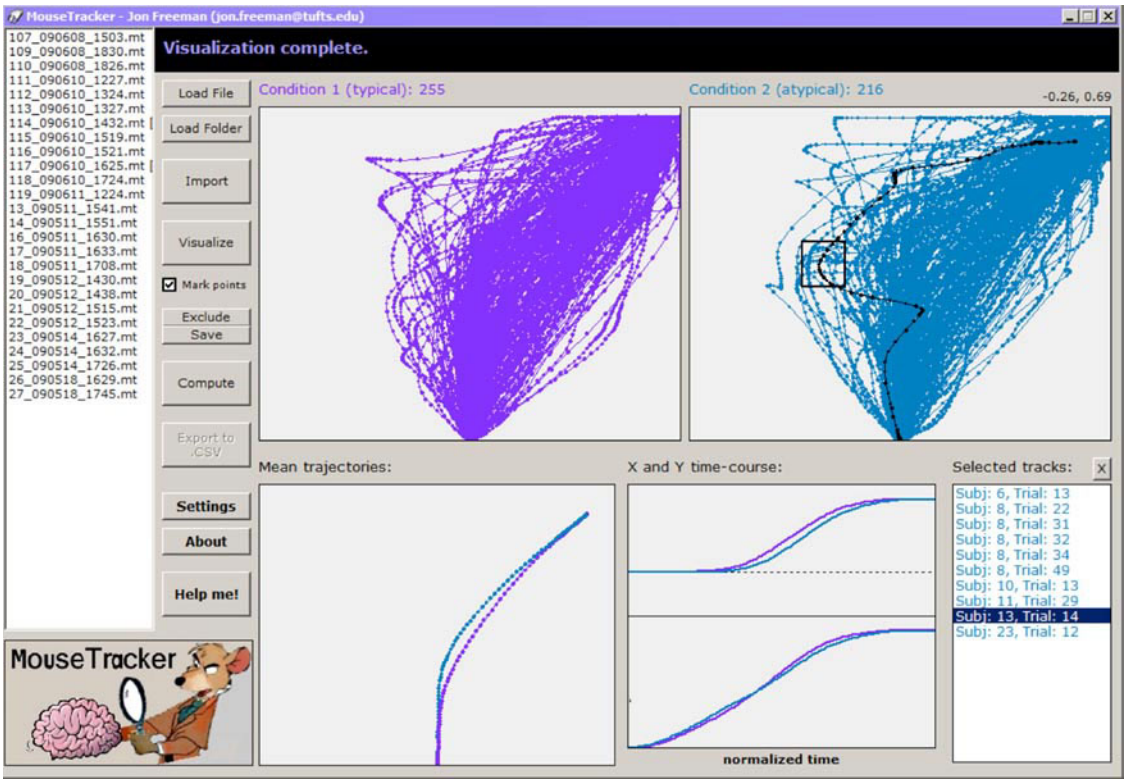

Figure 4. An example screenshot of the Analyzer program's main screen.

Importing trajectories. To import mouse trajectories from loaded data files, the user clicks the "Import" button. MouseTracker then searches through all the loaded .MT files for all condition codes that were used (as are specified in the experiment's .CSV file). A new screen appears, and all of the condition codes are displayed under the *All used codes* list (see Figure 5). In this screen, the user defines which condition codes are included in Condition 1 and Condition 2 for MouseTracker to analyze. Conditions 1 and 2 can also be renamed. Several additional options are available: exclude response errors; exclude trials whose RTs are not within a specified range; exclude trials whose initiation times are not within a specified range; whether the user would like to run a normalized time analysis or raw time analysis; whether to perform a between-subjects analysis; and whether to perform a correct/incorrect analysis (detailed below). In the Responses and remapping tab, the user can remap trajectories directed at one response alternative to another response alternative (e.g., for overlaying trajectories to permit direct comparisons), and can include/exclude trials involving specific responses. These options and specified condition codes can also be saved and loaded using .MTS files for convenience. When he or she is ready to import trajectories, the user clicks the "Go" button.

If the user chooses to perform a correct/incorrect analysis, condition codes are defined for only one condition. This is because trials involving a correct response are imported into Condition 1, and trials involving an incorrect response are imported into Condition 2. Thus, rather than examine two separate sets of condition codes, a correct/incorrect analysis allows a user to examine one single set of condition codes, but to separate out whether responses were correct versus incorrect. This is most helpful in experiments in which there are no true incorrect responses and participants are given no feedback on errors. Because correct and incorrect alternatives are always specified in the experiment .CSV file, a user could use this feature to separate out trials on the basis of whether the participant responded in a way that the researcher expected (i.e., correct) versus did not expect (i.e., incorrect). For instance, imagine an experiment in which participants are asked to decide whether they "like" or "do not like" different foods. There are no correct or incorrect responses. However, when a chocolate cake is presented, we can expect that participants should respond "like," and we may be interested to examine trials in which they respond unexpectedly ("do not like"). When participants are deciding whether they like or do not like chocolate cake, and they ultimately decide "do not like," a user might be interested to know whether trajectories nevertheless show an attraction to the "like" alternative before settling into "do not like." A correct/incorrect analysis would allow a direct comparison between the "like" trials (Condition 1) and "do not like" trials (Condition 2). Thus, rather than examine two separate conditions, a correct/incorrect analysis allows a user to examine the same condition but to separate out correct/expected responses versus incorrect/unexpected ones.

Visualizing, exploring, and manual exclusions. The user can choose to visualize trajectories with individual coordinates that are marked or not marked (appears as dots on the trajectories, as in Figure 4). After clicking the "Visualize" button, all trajectories corresponding with

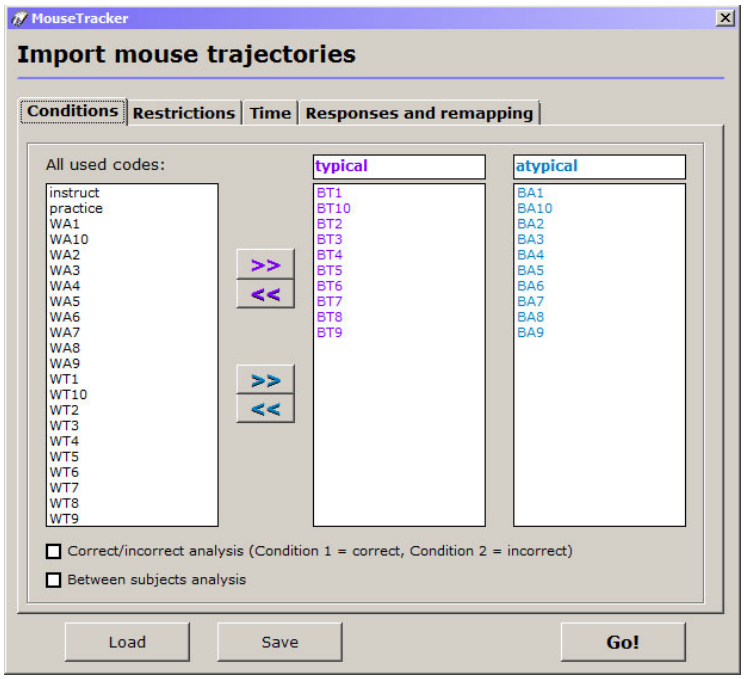

Figure 5. An example screenshot of the Analyzer program's import screen.

Condition 1 are displayed in the left pane, appearing here as purple. All trajectories corresponding with Condition 2 are displayed in the right pane, appearing here as blue trajectories. Trajectory colors can be customized by the user (see the Analyzer Program Settings section). The sample data shown here (Figure 4) come from an experiment similiar to that shown in Figure 1, where the "Start" button is at the bottom center and the response alternatives are at the top left and top right corners of the screen. Importantly, these trajectories have all been space rescaled, time normalized, and remapped to the response in the top right corner. Mean trajectories aggregated across all trials (and across all participants, if multiple participants are included in the analysis) are displayed in the bottom left pane. The time courses of the mean *x*-position and mean *y*-position (at each of the normalized or raw time steps) appear in the bottom middle of the screen (see Figure 4).

Every trajectory displayed in the Condition 1 and Condition 2 panes is able to be either individually selected for more detailed information or excluded from analysis. To select one or multiple trajectories, the user hovers over the Condition 1 or Condition 2 pane. The mouse cursor turns into a crosshair. The user then holds down the mouse and drags downward and rightward to draw a rectangular selection window, here appearing in black (see Figure 4). Once the mouse is released, any trajectories captured within this selection window are listed in the "Selected tracks" list in the bottom right of the screen. To select the trajectory, the user clicks on the trajectory item in the list, and that trajectory is redrawn using a different color (shown here in black). The color of the rectangular selection window and selected trajectories can be customized by the user (see the Analyzer Program Settings section). To obtain detailed information about a selected trajectory, the user double-clicks on the trajectory item in the list, and a new window will appear with details (see Figure 6 for a screenshot). Various details are provided here: subject ID, trial number, condition, RT, initiation time, stimulus file, left response alternative, right response alternative, actual response, two measures of spatial attraction toward all unselected alternatives (MD and AUC), complexity (*x*-flips and *y*-flips), and a time course of the trajectory's *x*- and *y*-coordinates as a function of normalized or raw time.

To manually exclude the selected trajectory item(s) from analysis, the user clicks the "Exclude" button. Doing this will recode the trajectory item(s) into Condition 888 (rather than into Condition 1 or 2). To save these exclusion changes, the user clicks the "Save" button. This will create a corresponding .LOG file in the same folder as the .MT file, which stores information about which trajectories have been converted into the Exclusion Condition 888. When loading the .MT file(s) later, MouseTracker asks whether the user wants to load .LOG files as well (and thus whether to load previous exclusions or not).

Computing and systematic screening. To compute by-trial measures of spatial attraction/curvature and complexity (MD, AUC, *x*-flips, and *y*-flips, detailed earlier), the user clicks the "Compute" button. Once these have finished computing, the mean MD and AUC values for Condition 1 and Condition 2 appear under the mean trajectory pane. If more than two response alternatives have been used, these mean MD and AUC values reflect trajectories' attraction toward the default-comparison response alternative (as specified in the trial's row in the experiment .CSV file). The user is then given the option of systematically screening for extreme trajectories on the basis of the MD or AUC measure. If choosing to conduct this systematic screening, the user is asked how many standard deviations away from the mean should be cut off and whether to use the MD or AUC measure. Trajectories excluded by this screening are recoded into Condition 666 (rather than into Condition 1 or 2). Whether this systematic screening is saved to .LOG files (like manual exclusions) is an option that is set by the user (see the Analyzer Program Settings section). The "Compute" button also generates two *z*-normalized values of the MD and AUC of all trajectories in Conditions 1 and 2, which are useful for distributional analyses (described earlier). One *z*-normalized value comes from pooling across both Conditions 1 and 2 ("together"), and another *z*-normalized value comes from pooling separately within Condition 1 and within Condition 2 ("separate"). These *z*-normalized MD and AUC values and the *x*-flips values appear in the final exported .CSV data sheet. If the user opted to conduct

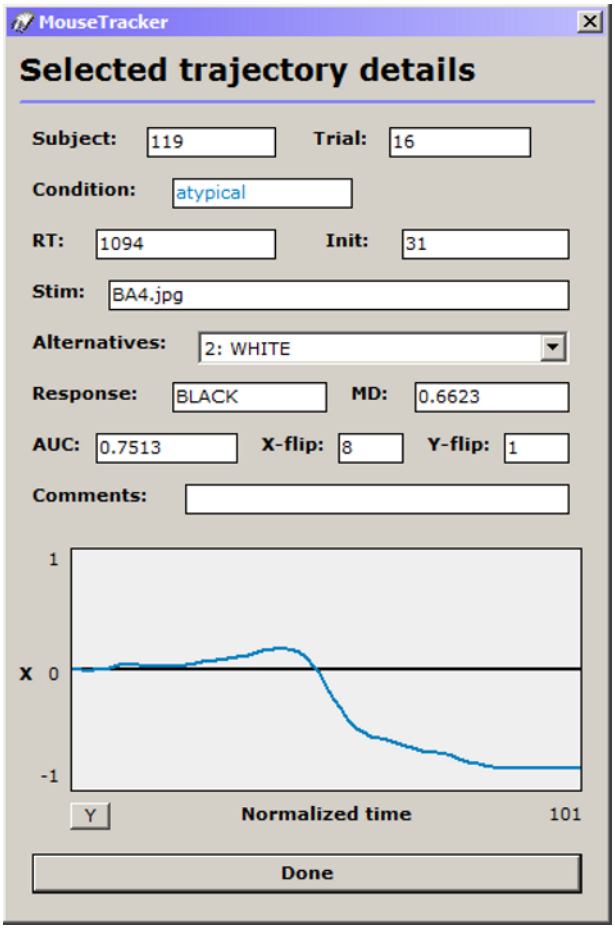

Figure 6. An example screenshot of the Analyzer program's selected trajectory screen.

a raw time analysis, then MD, AUC, *x*-flips, and *y*-flips are not computed. Rather, velocity and acceleration profiles are generated, and these are displayed in a new window after clicking the "Compute" button. This information appears in the output .CSV file.

Exporting Finalized Data

To export finalized data for hypothesis testing and further analysis in Excel, the user clicks the "Export to .CSV" button, which exports all of the data to a .CSV file readable in Excel. The user is prompted for a path and file name. MouseTracker then automatically loads this file in the default application associated with .CSV files (probably Excel) for the user to inspect it immediately.

There are two portions of an output .CSV file. The first is a list of every single trial across all participants and across all conditions. It lists all important information, including all of the measures that are viewable using the Analyzer program, in addition to the *x*- and *y*-coordinates corresponding with each trial, condition information, RTs, initiation times, MD, AUC, *x*-flips, and *y*-flips, among other information. Below the section of individual trials is a section detailing mean trajectories for Condition 1 and Condition 2 in each individual participant (all *x*-, *y*-coordinates), in addition to their mean values for MD, AUC, *x*-flips, *y*-flips, RT, and initiation time. If a raw time analysis was conducted, the *x*- and *y*-coordinates corresponding with the raw time steps (rather than with the normalized time steps) are provided. Moreover, velocity and acceleration information is detailed for each individual trajectory and for each participant's mean trajectories, and MD, AUC, *x*-flips, and *y*-flips are not included (because these are not available in a raw time analysis).

This outputted data sheet is MouseTracker's final product. Hypothesis testing can then be performed on mean trajectory data and various analyses can be conducted. See the analyses conducted in prior mouse-tracking studies for more information (Dale et al., 2007; Dale et al., 2008; Farmer et al., 2007; Freeman & Ambady, 2009; Freeman et al., 2008; Freeman et al., 2010; McKinstry et al., 2008; Spivey et al., 2005; Wojnowicz et al., 2009).

Analyzer Program Settings

Several features of the Analyzer program can be customized by clicking on the "Settings" button (see Figure 4). After this button is clicked, a new window appears that allows the user to set a number of options: the color of Condition 1 trajectories, the color of Condition 2 trajectories, the color of selected trajectories (and the rectangular selection window), the background color of trajectory panes, whether to save exclusions of systematically screened trajectories (made using the "Compute" button) to .LOG files, and whether to automatically check the Internet for software updates when the Analyzer program is started.

Accuracy and Reliability of MouseTracker Data

In two experiments, we determined the accuracy and reliability of MouseTracker's trajectory and RT data. In the first experiment, we computer-simulated cursor movements to show that the recorded trajectories from MouseTracker are highly similar to the computer-simulated trajectories that are known a priori. In the second experiment, we compared MouseTracker data from a simple categorization task that was repeated twice within each participant in order to determine test–retest reliability. Moreover, in order to demonstrate that MouseTracker is capable of recording accurate RTs, we compared RT measurements from MouseTracker with measurements from an established software that is known to record RTs accurately.

EXPERIMENT 1

To verify that MouseTracker captures mouse trajectories accurately, we conducted two simulations, each consisting of 24 trials, in which the response trajectory was generated by the computer and was thus known a priori. This allowed us to examine how well MouseTracker can record a known trajectory. To computer-generate trajectories, we developed a separate executable file (completely independent of the MouseTracker software) that could directly set the *x*-, *y*-coordinates of the mouse and move it along a known path at a known speed. This executable file was programmed to run in the background (without any graphical window displayed on screen) and to manipulate the location of the mouse cursor by making direct calls to the Windows application programming interface. This allowed the executable file to simulate movements of the mouse cursor just as a human user would do by physically moving a mouse.

Method

A simple MouseTracker experiment was designed. On the 24 trials in each simulation, Runner presented an image stimulus and waited for a mouse click at either the top left or top right corner while mouse movements were recorded. At the start of each trial, a "Start" button appeared in the bottom center portion of the screen. After the "Start" button was clicked, the mouse was automatically centered at the bottom center portion of the screen [0.00, 0.00], and the independent executable file, which was described previously, was immediately launched by pressing a designated keyboard shortcut (while the MouseTracker trial was still live).

In the first simulation (Simulation 1), the executable file (running in the background) directed the mouse position along a straight-line trajectory from the bottom center portion of the screen [0.00, 0.00] to a point located in the response button in the top right corner of the screen [0.98, 1.43] by recurrently changing the mouse's *x*-, *y*-coordinates. It moved the mouse along the straight-line trajectory in 60 equal increments occurring every 30 msec. After the 60th step (when the mouse was at its final destination in the top right corner), the executable file simulated a left-button mouse click, effectively completing the MouseTracker trial, and the executable file then terminated itself. This procedure was repeated for all 24 trials. In the second simulation (Simulation 2), the executable file directed the mouse position along a straight-line trajectory from the bottom center portion of the screen [0.00, 0.00] to the top center [0.00, 1.50], followed by an abrupt change in direction, redirecting the mouse along another straight-line trajectory from the top center [0.00, 1.50] to the top right [1.00, 1.50]. As in Simulation 1, it moved the mouse along this entire path in 60 equal increments occurring every 30 msec. After the 60th step, the executable file simulated a left-button mouse click and terminated itself.

Results and Discussion

For both Simulations 1 and 2 (in separate sessions), the outputted .MT data file from Runner was loaded into the Analyzer program (using the "Load File" button). Because Analyzer requires two conditions, we arbitrarily imported the first half of the 24 trials into Condition 1 and the second half of the 24 trials into Condition 2 (using the "Import" button). We then visualized the trajectories, computed MD and AUC values, and exported data to an output .CSV file (using the "Visualize," "Compute," and "Export to .CSV" buttons, respectively). Figure 7 depicts the Analyzer screenshots of the Condition 1 pane and the Condition 2 pane for both Simulations 1 and 2. Note that each Condition 1 pane and Condition 2 pane includes 12 individual trajectories, but because they completely overlap with one another, each set of 12 trajectories appears as only one single trajectory. Thus, from a casual inspection of the visualized data in Analyzer, MouseTracker appears to have effectively recorded the simulated trajectories of Simulations 1 and 2 in a nearly identical fashion across each set of 24 trials (and these were accurately recovered by the Analyzer program).

Simulation 1. To more rigorously assess the accuracy of the simulated trajectories, the .MT data file was manually opened in Excel for extracting raw *x*-, *y*-coordinates (without time normalization). Because on every trial the mouse began moving at slightly different time steps (due to random variability in the time elapsed between clicking the "Start" button and launching the independent executable file), we truncated by deleting any extraneous time steps with *x*-, *y*-coordinates of [0.00, 0.00] that took place before the mouse started moving. However, we always retained one time step with *x*-, *y*-coordinates of [0.00, 0.00] at the beginning of the trial to denote the start of the trajectory. Out of the 24 trajectories, 22 of them reached the final destination [0.98, 1.43] in 118 raw time steps, and 2 trajectories reached the final destination in 119 raw time steps. To make all 24 trajectories have equal length, we converted the 22 trajectories with 118 time steps into having 119 time steps by simply extending the last *x*-, *y*-coordinate pair into a 119th time step. By using the raw time stamp information of each of the 119 time steps and the knowledge that the mouse moved one sixtieth of the way between [0.00, 0.00]

Figure 7. Computer-simulated trajectories loaded into the Analyzer program (Experiment 1). In Simulation 1, a straight-line trajectory from the bottom center to the top right portion of the screen was simulated. In Simulation 2, a 90º trajectory was simulated involving a movement straight from the bottom center to the top center portion of the screen, and then a sharp redirection to the top right portion. Each of the four panes actually contains 12 individual trajectories, but because each set of 12 trajectories was recorded nearly identically, each set appears as only one trajectory because they overlap.

Table 1
Known and Observed Time-Binned *x*- and *y*-Coordinates in Simulation 1

	Bin 1 (1–30)		Bin 2 (31–60)		Bin 3 (61–90)		Bin 4 (91–119)	
	x	*y*	*x*	*y*	*x*	*y*	*x*	*y*
M	0.1246	0.1811	0.3748	0.5446	0.6254	0.9086	0.8708	1.2653
SD	0.0033	0.0048	0.0033	0.0048	0.0035	0.0051	0.0032	0.0047
Known	0.1249	0.1816	0.3748	0.5449	0.6247	0.9081	0.8702	1.2651

to [0.98, 1.43] every 30 msec, we calculated a separate series of 119 raw time steps that corresponded with the true known trajectory that was simulated.

To compare the observed trajectories with the true known trajectory, we created four time bins based on raw time steps (1–30, 31–60, 61–90, 91–119) and calculated *M*s and *SD*s of the observed trajectories' *x*-, *y*-coordinates. These coordinates appear in Table 1 along with the corresponding time-binned *x*- and *y*-coordinates of the true known trajectory. Note that the observed *x*-, *y*-values are highly similar to the known *x*-, *y*-values. As an additional analysis, we computed the Pearson correlation between the mean observed coordinates and the true known coordinates across all time steps, separately for *x*- and *y*-coordinates [$r_x(117) = .999950$, $p < .0001$, and $r_y(117) = .999950$, $p < .0001$]. Given the superficial accuracy of the observed trajectories (exemplified by visualized data depicted in Figure 7), the closeness of the time-binned observed *x*-, *y*-coordinates to those of the true known trajectory (Table 1), and near one-to-one correlation between observed and known coordinates across time, we conclude that MouseTracker quite accurately recorded these simulated trajectories.

Using the output .CSV file exported from the Analyzer program, we calculated the *M* and *SD* of the observed trajectories' MD and AUC values. Because the simulated trajectory was a straight line, we know that the true MD and AUC values are 0 (since there is zero curvature). The observed and known MD and AUC values for Simulation 1 appear in Table 2. The observed values were highly similar to the known values of 0. Thus, MouseTracker accurately captured the MD and AUC of the simulated trajectories.

Simulation 2. Recall that the simulated trajectories in Simulation 2 took the form of an initial straight line from the bottom center portion of the screen to the top center, and then a sharp 90º turn with a second straight line from the top center portion to the top right (see Figure 7). As in Simulation 1, we extracted raw *x*-, *y*-coordinates from the .MT data file, and we deleted extraneous time steps of [0.00, 0.00] that preceded mouse movement. Out of the 24 trajectories, 21 of them reached the final destination [1.00, 1.50] in 123 raw time steps, and 3 trajectories reached the final destination in 122 raw time steps. To make all 24 trajectories have equal length, we converted the 21 trajectories with 122 time steps into having 123 time steps by extending the last *x*-, *y*-coordinate pair into a 123rd time step. By using the raw time stamp information of each of the 123 time steps and the knowledge that the mouse initially moved one thirtieth of the way between [0.00, 0.00] to [0.00, 1.50] every 30 msec, and then moved one thirtieth of the way between [0.00, 1.50] and [1.00, 1.50] every 30 msec, we calculated a separate series of 123 raw time steps that corresponded with the true known trajectory that was simulated.

Four time bins were created based on raw time steps (1–30, 31–60, 61–90, 91–123), and the *M*s and *SD*s of the observed trajectories' *x*-, *y*-coordinates were calculated. Table 3 contains these coordinates in addition to the corresponding time-binned *x*- and *y*-coordinates of the true known trajectory. Note that the observed *x*-, *y*-values were highly similar to the known *x*-, *y*-values. We also computed the Pearson correlation between the mean observed coordinates and the true known coordinates across all time steps, separately for *x*- and *y*-coordinates [$r_x(121) = .999896$, $p < .0001$, and $r_y(121) = .999921$, $p < .0001$]. The superficial accuracy of the observed trajectories (Figure 7), the high similarity between the time-binned observed *x*-, *y*-coordinates and those of the true known trajectory (Table 3), and the near one-to-one correlation between observed and known coordinates across time demonstrate that MouseTracker accurately recorded these 90º simulated trajectories.

The *M*s and *SD*s of the observed trajectories' MD and AUC values (calculated from the output .CSV file) appear in Table 2. The true MD and AUC values of these trajectories were determined as follows. Recall that MD is the maximum perpendicular deviation from an idealized straight-line trajectory between an observed trajectory's start and endpoints. Also recall that AUC is the geometric area between the observed trajectory and this idealized straight-line trajectory (see Figure 2). For these simulated trajectories, MD would be 0.8321 because the maximum perpendicular deviation, in this case, is the altitiude drawn from the vertex of the right angle (formed by the observed trajectory) to the hypotenuse (the idealized straight-line trajectory), which is calculated as 0.8321 using basic trigonometry. As for AUC, the area between the simulated 90º trajectory and an idealized straight-line trajectory is simply a triangle with a base of 1.000 and a height of 1.500, resulting in a triangular area of 0.7500. This is the true known AUC. As is seen in Table 2, the observed MD

Table 2
Known and Observed MD and AUC Values in Simulations 1 and 2

	Simulation 1		Simulation 2	
	MD	AUC	MD	AUC
M	0.0011	−0.0003	0.8320	0.7565
SD	0.0001	0.0001	0.0009	0.0074
Known	0.0000	0.0000	0.8321	0.7500

Note—MD, maximum deviation; AUC, area under the curve.

Table 3
Known and Observed Time-Binned *x*- and *y*-Coordinates in Simulation 2

	Bin 1 (1–30)		Bin 2 (31–60)		Bin 3 (61–90)		Bin 4 (91–123)	
	x	*y*	*x*	*y*	*x*	*y*	*x*	*y*
M	0.0000	0.3794	0.0000	1.1279	0.2209	1.4981	0.7446	1.4981
SD	0.0001	0.0105	0.0001	0.0110	0.0078	0.0001	0.0077	0.0001
Known	0.0000	0.3745	0.0000	1.1236	0.2174	1.4981	0.7403	1.4981

and AUC values were highly similar to these known values. Thus, MouseTracker accurately captured the MD and AUC of these simulated 90º trajectories.

Taken together, the results of Simulations 1 and 2 show that MouseTracker accurately records mouse trajectories and accurately estimates MD and AUC. Next, we will examine the accuracy of MouseTracker's RT measurements and the reliability of its data overall.

EXPERIMENT 2

There were two goals of Experiment 2. The first was to verify the accuracy of MouseTracker's RT measurements. We accomplished this by conducting a study in which participants completed a simple categorization task using MouseTracker, and another time using software that is known to accurately record RTs: DirectRT (Empirisoft). The appearance and timing of the DirectRT task was made to be virtually identical to those of the MouseTracker task. If MouseTracker accurately records RTs, the *M*s and *SD*s of MouseTracker RTs should be similar to the *M*s and *SD*s of the RTs recorded by DirectRT. Moreover, the shape of the MouseTracker RT distribution should be similar to the shape of the DirectRT RT distribution. Distributional analysis is important because MouseTracker's sampling of mouse coordinates (60–75 Hz) could potentially interfere with the recording of RTs by forcing them into quantized intervals (rather than permitting an accurate continuum). If this was the case, this would be revealed in a comparison between the cumulative frequency distributions of MouseTracker RTs and DirectRT RTs.

The second goal of Experiment 2 was to determine the reliability of MouseTracker measurements. To accomplish this, we had participants complete an identical version of the MouseTracker task a second time. This allowed us to assess the test–retest reliability of MouseTracker's measurements.

Method

Participants. Twelve volunteers participated in exchange for $5.

Procedure. Participants completed three identical categorization tasks, once run in DirectRT and twice run in MouseTracker. The task was run twice in MouseTracker in order to examine the test–retest reliability of MouseTracker data. We counterbalanced across participants whether they completed the task first in MouseTracker and second in DirectRT (rather than first in DirectRT and second in MouseTracker). The second version of the MouseTracker task (retest) was always completed third. In between each of the three tasks was a filler task. All of the tasks were run on the same computer, and the MouseTracker and DirectRT tasks were virtually identical in appearance and timing.

On each trial, participants were presented with an image of a circle and were asked to identify whether it was red or blue by clicking a response button that was located in the top left or top right corner (marked RED and BLUE). We counterbalanced across participants whether RED appeared on the left and BLUE on the right, or vice versa (but this remained constant throughout all three task versions). Each task contained 20 trials (10 red and 10 blue) that were presented in a pseudorandomized order and were preceded by several practice trials. At the start of each trial, participants clicked a "Start" button at the bottom center portion of the screen. After clicking this button, an image of a circle appeared in its place, and participants clicked on a response button in either the top left or top right corner to indicate red versus blue.

Results and Discussion

There were no trials in which participants identified the color incorrectly. Trials with extreme MD values (4 *SD*s above or below the mean) were discarded (1.7%).

Accuracy of MouseTracker RT data. RTs in the MouseTracker task did not significantly differ from those in the DirectRT task [$t(11) = 1.19, p = 26$], although there was a trend of RTs in MouseTracker (M = 1,098 msec, SD = 203 msec) being longer than RTs in DirectRT (M = 1,052 msec, SD = 186 msec). This lack of significance is likely due to low statistical power, given a small number of participants. With RT *SD*s as the dependent measure, there was no evidence of MouseTracker adding any variability to RT measurements (M = 167 msec, SD = 61 msec) relative to those in DirectRT (M = 160 msec, SD = 59 msec) [$t(11) = 0.36, p = .72$]. To inspect for possible quantization effects or differences in distributional shape, we plotted the cumulative frequency distribution (CFD) of the MouseTracker RTs and compared it with the CFD of the DirectRT RTs (Figure 8). Any quantizing would be evidenced as step-like patterns in the CFD for MouseTracker RTs, which were not seen (see Figure 8). Moreover, the Kolmogorov–Smirnov test indicated that the shapes of the DirectRT and MouseTracker RT distributions, once made equal mean and variance (via *z*-normalization), did not significantly differ from one another (D = .05, p = .94). Thus, although MouseTracker RTs were slightly longer than those in DirectRT, MouseTracker did not introduce any more variability to RT measurements, and did not constrain RTs to a limited set of values (quantizing), or affect its distributional shape.

We conclude that MouseTracker measures RTs relatively accurately. It did not add random error (*SD*s) to RTs, nor did it affect their distributional shape or elicit quantizing. However, MouseTracker appeared to introduce a small amount of systematic error (*M*s) to RTs, shifting the distribution by approximately 40–50 msec. This may have been due to small differences in the way MouseTracker

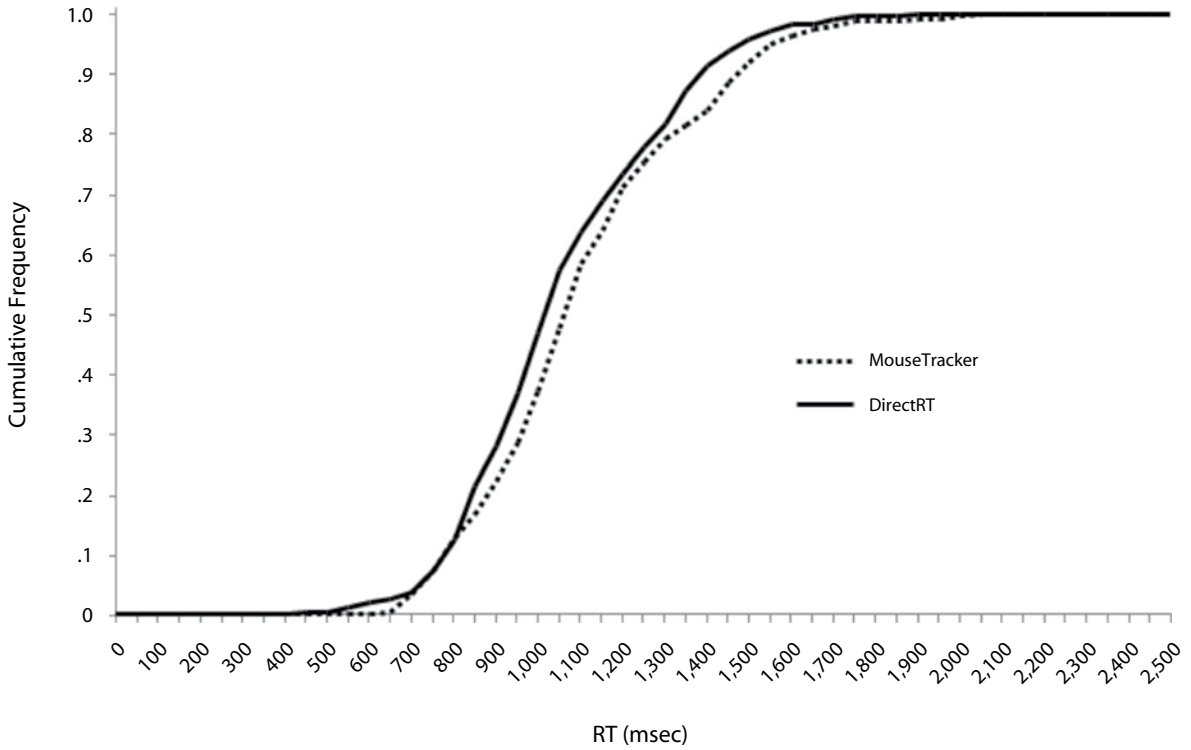

Figure 8. Cumulative frequency of MouseTracker and DirectRT trials is plotted as a function of reaction time (RT).

displays stimuli on the screen, causing a slight delay in participant responses. We recommend that users consider MouseTracker's RTs to be an accurate measure, but that they bear in mind that the RTs may be slightly (but systematically) longer than RTs more accurately recorded by software such as DirectRT. Because this error is systematic, it does not pose any problems if all RT measurements within an experiment are performed by MouseTracker. Directly comparing RTs from MouseTracker with those from software such as DirectRT, however, may be more difficult.

Reliability of MouseTracker data. Serving as estimates of test–retest reliability, Pearson correlations between the first and second MouseTracker tasks were computed for the following measures: MD, AUC, RT, initiation time, and *x*-flips (Table 4). Across the measures, test–retest reliability was quite good [$rs(10) = .78–.95$] (all $ps < .01$). We conclude that MouseTracker's measurements are reliable, showing consistency over time.

CONCLUSION

Online methods for assessing the temporal dynamics of mental processing have been scarce. The RT has been the gold standard in many research domains. In the present article, we presented the software package, MouseTracker, and described how its recording and analysis of computer mouse movements can reveal the real-time evolution of psychological responses. Using MouseTracker allows a single RT to be opened up into a continuous ongoing stream of rich cognitive output. We hope that researchers will adapt current tasks that use RT measures for use with MouseTracker, so that continuous streams of output can be tracked and the fine-grained temporal dynamics leading up to participant responses can be unveiled. Moreover, new tasks can be designed for endeavoring new interrogations into the temporal details of mental phenomena that researchers have not been able to fully inspect.

MouseTracker was designed to allow any psychologist or cognitive scientist from any niche to easily harness the power of a mouse-tracking methodology. MouseTracker is free but affords the rich temporal resolution of expensive ERP and eyetracking measures. Our hope is that this software will equip researchers with a practical, inexpensive, and time-sensitive methodology—needing nothing more than a laboratory computer and a mouse—to begin digging more deeply into the real-time dynamics of mental activity.

AUTHOR NOTE

We thank Evan Silverstein for creative insights. While developing the software, J.B.F. was supported by a Tufts University graduate fellowship.

Table 4
Test–Retest Reliability of MouseTracker Data

Mouse Trajectory Parameter	*r*
Maximum deviation	.78
Area under the curve	.81
Reaction time	.95
Initiation time	.90
Complexity (*x*-flips)	.79

The research reported in the present article was supported by Research Grant NSF BCS-0435547 given to N.A. Correspondence concerning this article should be addressed to J. B. Freeman, Department of Psychology, Tufts University, 490 Boston Avenue, Medford, MA 02155 (e-mail: jon.freeman@tufts.edu).

REFERENCES

Abrams, R., & Balota, D. (1991). Mental chronometry: Beyond reaction time. *Psychological Science*, **2**, 153-157.

Akamatsu, M., & MacKenzie, I. S. (2002). Changes in applied force to a touchpad during pointing tasks. *International Journal of Industrial Ergonomics*, **29**, 171-182.

Dale, R., Kehoe, C., & Spivey, M. J. (2007). Graded motor responses in the time course of categorizing atypical exemplars. *Memory & Cognition*, **35**, 15-28.

Dale, R., Roche, J., Snyder, K., & McCall, R. (2008). Exploring action dynamics as an index of paired-associate learning. *PLoS ONE*, **3**, e1728.

Dietrich, E., & Markman, A. B. (2003). Discrete thoughts: Why cognition must use discrete representations. *Mind & Language*, **18**, 95-119.

Farmer, T. A., Anderson, S. E., & Spivey, M. J. (2007). Gradiency and visual context in syntactic garden-paths. *Journal of Memory & Language*, **57**, 570-595.

Finkbeiner, M., Song, J. H., Nakayama, K., & Caramazza, A. (2008). Engaging the motor system with masked orthographic primes: A kinematic analysis. *Visual Cognition*, **16**, 11-22.

Freeman, J. B., & Ambady, N. (2009). Motions of the hand expose the partial and parallel activation of stereotypes. *Psychological Science*, **20**, 1183-1188.

Freeman, J. B., Ambady, N., Rule, N. O., & Johnson, K. L. (2008). Will a category cue attract you? Motor output reveals dynamic competition across person construal. *Journal of Experimental Psychology: General*, **137**, 673-690.

Freeman, J. B., Pauker, K., Apfelbaum, E. P., & Ambady, N. (2010). Continuous dynamics in the real-time perception of race. *Journal of Experimental Social Psychology*, **46**, 179-185.

Gold, J. I., & Shadlen, M. N. (2001). Neural computations that underlie decisions about sensory stimuli. *Trends in Cognitive Sciences*, **5**, 10-16.

Goodale, M. A., Pelisson, D., & Prablanc, C. (1986). Large adjustments in visually guided reaching do not depend on vision of the hand or perception of target displacement. *Nature*, **320**, 748-750.

Magnuson, J. S. (2005). Moving hand reveals dynamics of thought. *Proceedings of the National Academy of Sciences*, **102**, 9995-9996.

McKinstry, C., Dale, R., & Spivey, M. J. (2008). Action dynamics reveal parallel competition in decision making. *Psychological Science*, **19**, 22-24.

Newell, A. (1980). Physical symbol systems. *Cognitive Science*, **4**, 135-183.

Pylyshyn, Z. W. (1984). *Computation and cognition*. Cambridge, MA: MIT Press.

SAS Institute (1989). *SAS/STAT user's guide*. Cary, NC: Author.

Schmidt, T. (2002). The finger in flight: Real-time motor control by visually masked color stimuli. *Psychological Science*, **13**, 112-117.

Song, J. H., & Nakayama, K. (2006). Role of focal attention on latencies and trajectories of visually guided manual pointing. *Journal of Vision*, **6**, 982-995.

Song, J. H., & Nakayama, K. (2008). Target selection in visual search as revealed by movement trajectories. *Vision Research*, **48**, 853-861.

Spivey, M. J. (2007). *The continuity of mind*. New York: Oxford University Press.

Spivey, M. J., & Dale, R. (2004). The continuity of mind: Toward a dynamical account of cognition. In B. H. Ross (Ed.), *The psychology of learning and motivation: Vol. 45. Advances in research and theory* (pp. 87-142). San Diego: Elsevier.

Spivey, M. J., & Dale, R. (2006). Continuous dynamics in real-time cognition. *Current Directions in Psychological Science*, **15**, 207-211.

Spivey, M. J., Grosjean, M., & Knoblich, G. (2005). Continuous attraction toward phonological competitors. *Proceedings of the National Academy of Sciences*, **102**, 10393-10398.

Spivey, M. J., Richardson, D. C., & Dale, R. (2008). The movement of eye and hand as a window into language and cognition. In E. Morsella, J. A. Bargh, & P. M. Gollwitzer (Eds.), *Oxford handbook of human action* (pp. 225-249). New York: Oxford University Press.

Tipper, S. P., Howard, L. A., & Houghton, G. (1998). Action-based mechanisms of attention. *Philosophical Transactions of the Royal Society B*, **353**, 1385-1393.

Wojnowicz, M. T., Ferguson, M. J., Dale, R., & Spivey, M. J. (2009). The self-organization of explicit attitudes. *Psychological Science*, **20**, 1428-1435.

(Manuscript received June 1, 2009;
revision accepted for publication September 13, 2009.)