Unpublished draft for the Engineering for Humans course at Aalto University 2021
Antti Oulasvirta

Contents

Chapter 1

Motor control

1.1 Introduction

This may come as a surprise for students of computer science: The way people move their bodies is a central research question in the quest to develop better computers. Although one may easily think computer use as a predominantly cognitive or linguistic activity, in fact much of what we *do* is movement. Understanding motor control will help you make computer use more efficient, enjoyable, and safe.

Motor control refers to the production of body motion with desired characteristics. Consider swiping a touchscreen to turn the page, or an eSports professional's quick response to a competitor's move. Basic research on motor control studies the underpinning neural, psychological, and biomechanical mechanisms. Researchers investigate, for example, how a bird species controls the frequency and force used in flapping the wings, or what explains an athlete runner's superior reaction time. Applied research on motor control, on the other hand, is interested in factors that help improve motor control in practical settings. In the context of HCI, the emphasis is put on understanding the design of better technology. In this Chapter, the focus is on **engineering models** (Section X) that help, on the one hand, understand problems in motor control deeper than intuition and, on the other, solve practical problems related to design. We cover elementary motor control tasks ranging from reactions to complex, carefully coordinated tasks like typing.

1.1.1 Research on motor control in HCI

Motor control has been a central topic in human factors and HCI research for decades. Every method, device, and technique for **input** involves some way of *mapping* users' physical motion to a set of input message. There is no way to interact with a computer – notwithstanding brain–computer interfaces – that does not include *physical motion* of the human body. As we discuss in Section X, to design an input method, designer have to make a larger number of design decisions that affect how easy and comfortable the implied motions are. If we change something as innocent the CD gain function, the function that maps the velocity of the sensor to the velocity of the cursor, you may need to move your hand longer for the cursor to move the same distance. Also visual design of an interface affects motor control: the positions and sizes of targets on the display affect the type of motions we need to select them.

Research on motor control in HCI comprises three types of activities. First, **empirical research** looks at motor control in computer input, with the aim of understanding factors that affect performance. In a typical study, users are asked to perform a motor task to the best of their ability, and their performance is measured and compared across conditions.

Second, **modelling** seeks to form mathematical and computer models that link essential factors with metrics describing motor performance. These models are typically expressed as regression models, like Hick–Hyman law and Fitts' law that we describe later in this Chapter. They contain terms that describe the motor task or some conditions such as those related to the design, and link them to some variable or variables of interest, like task completion time. The focus in HCI is not that much on theoretical plausibility but on practical validity, how they can be used to inform design and decision-making in realistic contexts.

Third, **innovation** focuses on novel technological means to facilitate motor control. This branch of research typically pursues demonstrated improvements over existing means of input. For example, research on accessible interfaces looks at input devices that allow people with cerebral palsy to control computer cursor.

Quite different, yet mostly complementary scientific perspectives to the topic exist. The **performance view** seeks to understand speed and accuracy of produced motions. In the context of HCI, 'performance' refers to the capability of the user to successfully and efficiently produce desired input events. A central phenomenon is *speed–accuracy trade-off*: the finding that

movements cannot be produced faster without compromising their accuracy and, vice versa, trying to be more accurate will compromise speed. Every motor task has its own trade-off function; the function that maps changes in speed to accuracy. Another perspective is *skill acquisition*: how performance develops over time, and what kinds of practice conditions are favorable. The performance view can also take a *robustness* perspective: how reliably is the user able to keep a certain level of performance when conditions change? This can be studied by looking at the probability of errors during performance or variability under changing circumstances. Reliability is particularly important for mobile devices: a good design allows the user to maintain high performance when on the move.

The **information capacity view** seeks to understand how efficiently messages can be conveyed via computer as an information channel (see Section X). The benefit of information theory is that it provides a single frame of converge to look at both speed and accuracy, as measured via concepts like throughput.

The **individual differences view** emphasizes the different capabilities and motivations of people. Whereas performance-oriented research normally looks at young, healthy adults, the goal of individual differences research is to understand the diversity of everyday motor control in HCI. It rejects the "one design fits all" approach and studies methods that adapt, allow customization and personalization.

It is ethically important to enable all people to use computers to the best of their ability. With increasing digitalization of services, low-level enablers like input devices, have become a differentiating factor between those who can and those who cannot, with at-times severe implications to people's ability to participate in the society. In accessible design, the ethos is that design that starts with the view that people are different will not only serve special groups but all users.

1.1.2 Understanding motor tasks in HCI

A *motor task* is the elementary unit of research in motor control studies in HCI. A motor task is a task where a desired state-change in the computer needs to be produced by a movement, or movement sequence, with *particular properties*. As we learn next, this is more complex that it sounds. To understand a motor task, we need a bit of terminology that helps us see the contributing factors.

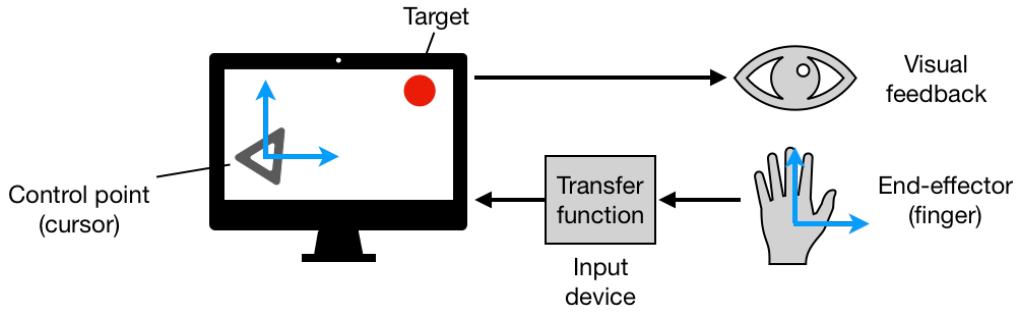


Figure 1.1: Terminology to define a **motor task** in human–computer interaction: a control point with its movement degrees, the target, end-effector with its movement degrees, the transfer function, and feedback

Factors defined by the computing system

We start by looking at factors affected by the computer:

- Control point: What is being controlled in the computer
- Degrees of freedom: How many degrees of freedom are there to move it
- Movement demands: How the control point must be moved to complete the task
- Transfer function: How movement is mapped to changes in control point
- Feedback: What information is available for correcting movement

Control point refers to a step-change in the computer's processing to which control is being applied. Step can be either discrete or continuous. A text prompt is a discrete control point: It allows the entry of a single letter of a time in order to commit a string. The mouse cursor is in principle a continuous control point. However, the resolution of the sensing device and the display jointly determine how many discrete values it can effectively take.

Degrees of freedom refers to the number of dimensions in which the control point can be controlled. A slider is controlled in one dimension, its DOF = 1, whereas the mouse cursor is controlled in two dimensions, its DOF = 2. There are input devices that have more than our regular DOFs, for example a joystick with DOF = 6. A higher number of DOFs does not automatically mean that performance will be higher, but most likely it will be harder to

learn, because the involved motor actions are more complex. DOFs can be further analyzed in terms of movement types:

1. Translational movement refers to motion on a plane.

2. Rotational movement refers to angular motion around a pivot point.

Movement demands are the constraints for acceptable movement. These can be set by the design of the input device or the environment. Motor demands come in three types: spatial, temporal, and kinetic. Spatial demands refer to requirements to keep movement within given spatially defined bounds. For example, to press a button, the push-down movement must hit the key cap of the button from the right angle. Temporal demands refer to time within which movement must be finished, or the speed with which it must be carried out. Kinetic demand refers to force that must be produced. For example, mechanical and touch buttons have widely different temporal and kinetic demands, however their spatial demands are similar.

Transfer function refers to the transduction of physical motion into movement of the control point (Section X). The function is jointly constituted by the full input sensing pipeline.

Feedback is critical for correcting performance and developing motor skill. Feedback can be instantaneous or delayed. Instantaneous feedback will provide feedback associated to motor action, while delayed will provide it at an opportune moment later on, for example after a session has ended.

A *complex motor task* can consist of several concurrent or consecutive atomic motor tasks. For example, entering a given phrase on a touchscreen using index finger involves a sequence of motor tasks with changing spatial target. Some tasks require carrying out tasks concurrently, for instance many games require simultaneous control on keyboard and a pointing device.

Factors defined by the user

From the user perspective, the defining characteristics of a motor task are:

- End effector: What part of the human body needs to be moved to produce a desired change in the control point

- Performance objective: What does the user want to achieve in carrying out the task

- Kinematic chain: What other parts of the body contribute to the control of the end effector

End effector is the segment in body – such as a tip of a finger – that must be moved in order to produce the desired effect on the sensing device. In computer vision -based hand tracking, we typically use finger tips as end effectors to pinch, point, rotate etc. When using a touchscreen device, the end effector *coupled* with the control point. When using an indirect pointing device, such as a touchpad, the two are *decoupled*. [1]

Performance objective is the subjective goal the user has when carrying out the motor task. It can concern things like how fast, with how few errors, or with what sort of energy use the task should be completed.

Kinematic chain is the chain of joints that contribute to the movement of the end effector. For example, to move the index finger as an end-effector in touchscreen interaction, the kinematic chain typically consists of the shoulder, the elbow, and the wrist – assuming that the finger is locked. However, also the position of the head is important, because it affects visual feedback. Each joint has its own degrees of freedom. The kinematic chain of reaching to grasp an apple is shown in Figure 1.2. If you 'lock' a joint, for example elbow, the motor control challenge will change wildly. You can try for example using a touchpad (e.g, on laptop) with a different kinematic chain; try using, say, the forehead as the end-effector. The impact on performance will be drastic. Understanding the kinematic chain in input is important, it will help you understand which muscles and muscle groups contribute to the task, and how the visual system is positioned in relationship to the display to receive feedback. Note that the DOFs of the kinematic chain may be different from that of the input device. For example, the mouse has DOF = 2, but the shoulder–elbow–wrist chain has 3 + 1 + 2 = 6 DOFs.

Chapter overview

In the rest of the Chapter, we discuss five fundamental motor tasks: simple reaction, choice reaction, pointing, steering, and gesturing. Their main differences are given in Table 1.3. Each is associated with a motor challenge common in HCI. We provide an engineering model for each.

[1] Note: although the term end effector originates from studies of human motor control, sometimes the term is mistakenly used in HCI literature to describe the control point (e.g., cursor).

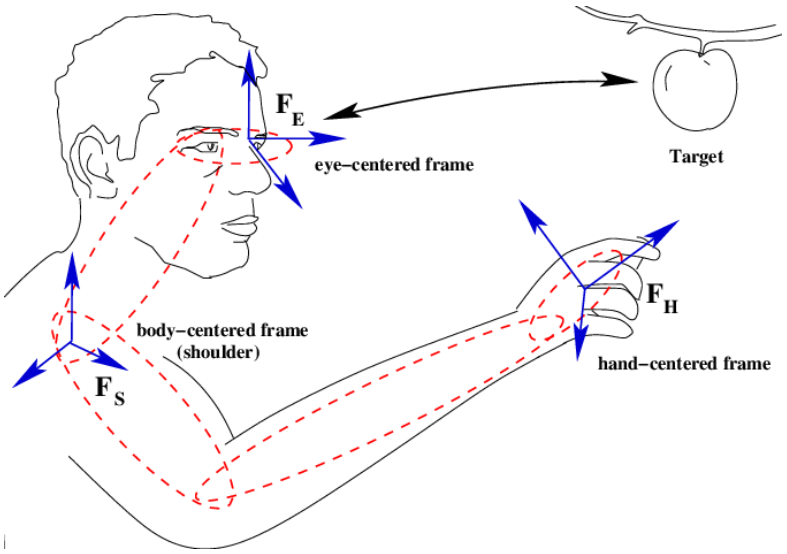


Figure 1.2: The kinematic chain and control dimensions in reaching to grasp an apple. [Soures et al. 2007]

Motor task	Target	Degrees of freedom	User's goal	Example model
Simple reaction	An abrupt event in the	1	React by pressing a button	Ratcliff
Choice reaction	Choice among N options	N > 1 options	Choose correct option	Hick-Hyman
Pointing	A spatially extended	One or more	Bring control point within target bounds	Fitts
Steering	A tunnel	One or more	Keep within bounds	Accot-Zhai
Gesturing	A shape	One or more	Reproduce movement shape	Cao-Zhai

Figure 1.3: Five fundamental motor tasks in HCI

Figure 1.4: **Simple reaction** is a task where the user must respond to a prompt as quickly as possible by pressing a button. **Choice reaction** generalizes this to the case where more than one response option is available.Image source: Wikimedia Commons

1.2 Simple reaction

Simple reaction is perhaps the most elementary response type in HCI.: Something appears on the display or in the environment, and the user must respond to it as quickly as possible. Consider the following examples:

- Getting rid of an annoying dialogue asking if you want to install a new version of software
- Blocking an enemy's move with a counter-move in a computer game
- Answering an incoming call on a mobile device

As opposed to choice reaction that is discussed next, only one response alternative is available in simple reaction—this is why it is called simple. The response can be stated by saying something aloud, pressing a button with a finger, or even by thinking of a response in brain–computer interaction. From a communication perspective, simple reaction carries a single bit of information: that a response has taken place. Typing a phrase, for instance, is far more complex, as one must consecutively select (choose) the right key from a set of at least 26 alternative characters.

Performance in this task is measured in milliseconds as the time duration between the onset of the event and the user's response. Simple reaction

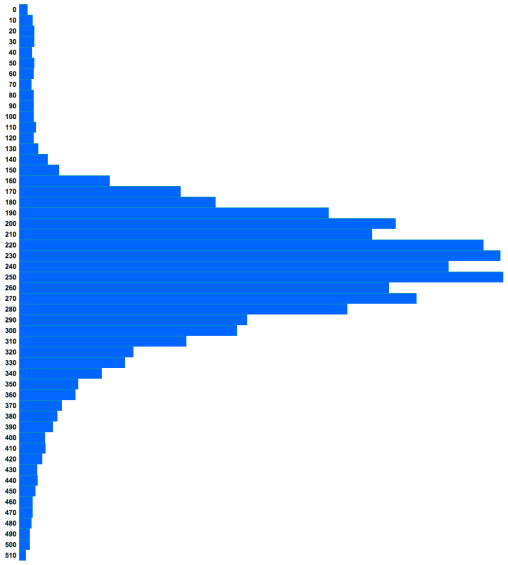


Figure 1.5: Distribution of reaction times from an online reaction time service [*www.humanbenchmark.com*; MUST BE ROTATED]

tasks is among the fastest human responses in human–computer interaction. Typical responses, depending on the input and output modalities, are in the range of few hundred milliseconds. However, within this rather narrow range, relatively large differences can be observed depending on the condition. Figure [**?**] shows reaction time data from almost ten million responses in an online service. Reaction time distributions are often Gaussian or skewed Gaussians.

Simple reaction has been studied in psychology for over a century and the involved cognitive processes are somewhat known. Some of the earliest studies considered the vigilance of soldiers in World War II: After sleep deprivation, or in a stressful event, how would reaction time be altered? Beyond military, simple reaction has broad applications in studies of human factors, sleep deprivation, psychological assessment, and in animal research. As a model, we discuss the Ratcliff model, which provides both theoretical understanding and practical applications to making interfaces that improve reaction time.

1.2.1 Ratcliff Model

The Ratcliff model[2] is an **evidence accumulation model** that predicts the distribution of reaction times as a function of what happens after the stimulus has appeared. The idea in evidence accumulation models is that that perceptual evidence for and against responding accumulates until some *threshold* is met and the motor response is launched.

- Stimulus onset: The event that one should respond to appears
- Perceptual encoding: The event is encoded as a candidate for one that should be responded to
- Evidence accumulation: Every fixation samples more evidence pro/against the decision to respond
- Decision: When enough evidence has accumulated to meet a decision threshold, the corresponding motor action is launched
- Motor action: Launching the overt, movement response

Hence, between the stimulus and the observable response many things happen. Evidence accumulation models can account for some effects of user interface design as well as various task-related, individual, and contextual factors. They predict naturally occurring variation we can observe in performance when the same reaction task is repeated.

The Ratcliff model assumes that simple reaction RT has two sources of variation: decision time T_d and nondecision time T_{er}. Nondecision time is further broken down into two subcomponents x and y. The first nondecision event is the perceptual encoding of the stimulus that lasts for some duration, marked with x. After perceiving the stimulus, a stochastic decision process starts. During this period evidence is accumulated (diffused) in the brain that the stimulus should be responded to. This evidence accumulation phase is affected by perceptual conditions like noise (e.g,. poor resolution, poor eye sight) and the complexity of the visual scene. Finally, after sufficient evidence has diffused to surpass threshold a, the decision process stops and continues to motor response process with duration y. Thus, reaction time is given as:

$$RT = T_d + T_{er} = x + T_d + y \tag{1.1}$$

[2]Ratcliff and Van Dongen 2011, Wagenmakers 2007

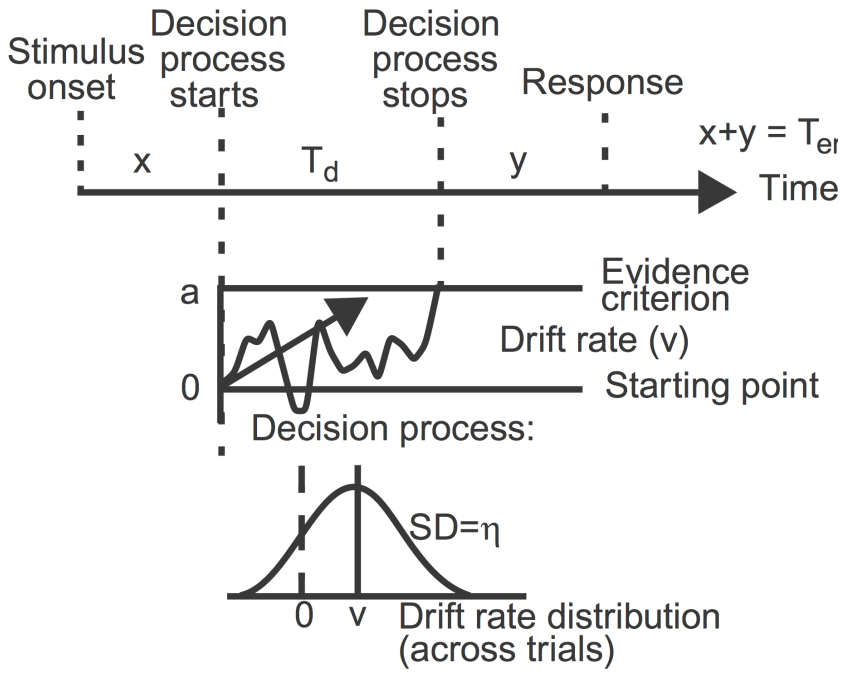


Figure 1.6: The Ratcliff model explains simple reaction as a linear sum of a decision task and two non-decision tasks. After perceiving the prompt, a decision task starts. It is assumed to be a stochastic diffusion process where evidence accumulates toward threshold a, at which point the response is emitted. [Figure adopted from Ratcliff and Van Dongen 2011, *PNAS*]

Figure 1.6 illustrates the model.

Understanding this decision process is the key to understanding how user interfaces improve users' reaction time. The duration of the decision process T_d is user- and task dependent and varies across trials. The drift rate (or the accumulation of evidence) is assumed to be normally distributed with mean v and SD η. [3] The two nondecision components x and y are summed to T_{er} and treated together in the model. They can also change according to the user interface. For example, an auditory prompt may take longer to register than a simple visual symbol. This nondecision component is also assumed to vary across trials with SD s_t.

The standard way of plotting the predictions is *the hazard rate function*. It gives the probability that the decision process terminates in the next instant of time, given that it has survived to that time. Formally $h(t) = f(t)/(1-F(t))$, where $f(t)$ is the probability density function and $F(t)$ is the cumulative density function. Figure 1.7 shows three examples assuming the same decision threshold a:

- *Slow but perfect responder*: In the top-most figure, drift rate v is mediocre (0.4) with no variation ($\eta = 0$). The shape of the function is "perfect" in the sense that it is only achievable by a user who can decide the stimulus with no hesitation.

- *Fast responder*: Here drift rate v is higher but there is more variation ($\eta = 0.3$). This yields a much faster response, peaking around 300 ms.

- *Slow responder*: Here drift rate v is mediocre (0.4) but there is some variation ($\eta = 0.3$). The hazard distribution has a long tail. This kind of variation could be produced for example by sleep deprivation or by noisy, hard-to-interpret display.

Consider an application [4] to driving. A user is seated in a driving simulator and has to brake when the lead vehicle breaks. In one of the conditions, the user was simultaneously speaking to a phone. Braking time increased. About 30 milliseconds were attributable to s_t; that is, their detection of braking lights slowed down. The rest about 100 msec were attributed to reduced

[3] The diffusion process is given by $dX(t) = vdt + sdW(t)$, where $dX(t)$ is the change in the accumulated evidence X for a time interval dt, v is drift rate, and $sdW(t)$ are zero-mean random increments with infinitesimal variance s^2dt.

[4] Ratcliff and Strayer 2013 Psychon Bull Rev

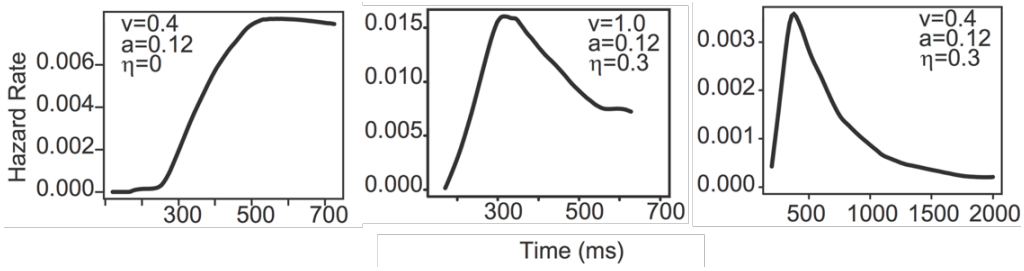


Figure 1.7: Three hazard function examples. For comparison, notice the different scales of axes [Figure adopted from Ratcliff and Van Dongen 2011, *PNAS*]

drift rate. This result indicated that phone conversation retards evidence accumulation, hampering the driver's ability to respond to abrupt events. This observation challenges the intuition that it is safe to talk while driving because eyes can be kept on the road.

1.3 Choice reaction

In *a choice reaction task*, instead of one response option like in simple reaction, n options are available. When a cue (stimulus) appears, the user must execute the *corresponding* response as quickly as possible by pressing the associated key. Each cue is associated to a single response; however, cues can appear with different probabilities. Fingers are supposed to be resting on the associated keys, in order to minimize effect of pointing in the response.

Performance in choice reaction tasks is measured as *choice reaction time* (CRT): It is the time that has elapsed from presentation of the cue to the response. Errors – choosing wrong response or not responding – can be dealt with either by insisting on correct response, or by allowing errors to happen and reporting error rate alongside with CRT.

Choice reaction is a generalization of simple reaction. When $n = 1$, we have the simple reaction. When $n = 2$, we have the 2-alternative forced choice (2AFC) task, there are two response options and one must be chosen. 2AFCs are common in HCI. Consider an incoming call, for example; It forces the user to pick between Answer and Reject call options. When n increases beyond two, we find an interesting and practically important relationship between n and CRT.

1.3.1 Hick–Hyman Law

The Hick–Hyman law is a statistical relationship between n and CRT discovered independently by William Edmund Hick and Ray Hyman. The law states that when the number of response options increases, choice reaction time increases. Trying to be faster than what the law suggests will lead to errors. By allowing more time for responding, fewer errors will occur.

Formally, given n equally probable response options, the average CRT follows approximately:

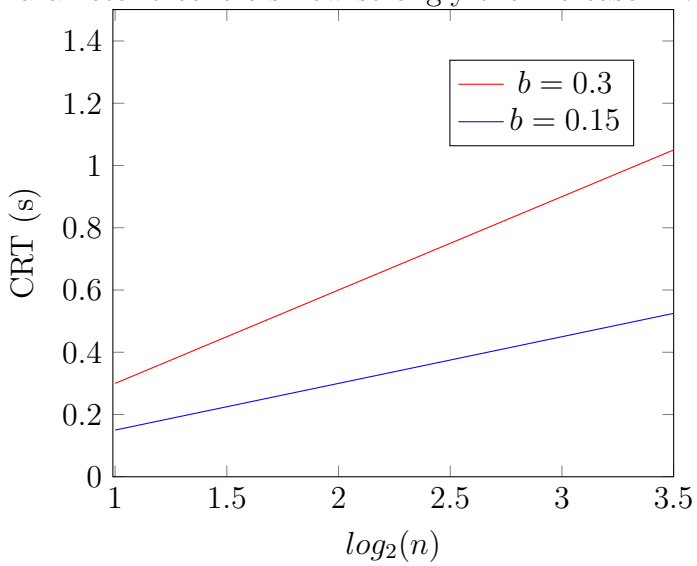
$$CRT = a + b \cdot \log_2(n) \tag{1.2}$$

where a and b are empirical constants, determined by fitting the line to data. One can be added to n in cases there is uncertainty about whether to respond or not:

$$CRT = a + b \cdot \log_2(n + 1) \tag{1.3}$$

Not-to-respond is just one more option.

Parameter b controls how strongly the increase in n affects CRT:



Why is there a *binary* logarithm in the standard formulation? According to one theory, it reflects binary search from the part of the user. When a cue appears, the user first picks one half of the options and reject the other half, then picks half of the remaining options, and so on, until finally identifying the correct response. This form also links Hick–Hyman law to information theory, see Section X.

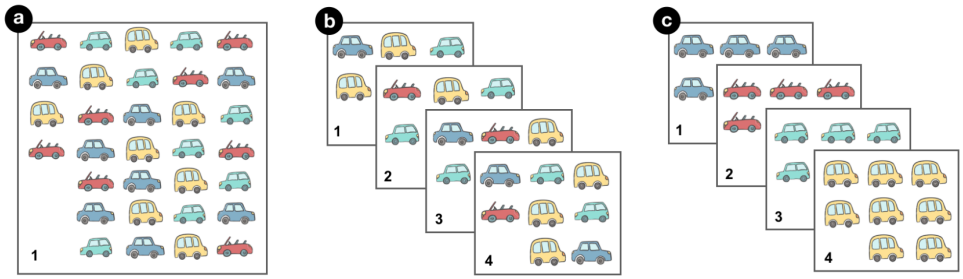


Figure 1.8: You need to show 32 items to the user. According to Hick–Hyman law, which of the designs is fastest to use? [Adopted from Liu et al. CHI2020]

Another interpretation of the law is that it denotes **uncertainty about the stimulus**. In the case of choices with unequal probabilities, the law can be expressed as:

$$CRT = a + bH \tag{1.4}$$

where

$$H = \sum_{i}^{n} p_i \log_2(1/p_i + 1) \tag{1.5}$$

and p_i marks the probability of response option i. Even though there is a large number of options available, if only a small subset is effectively in use (i.e., the sum of their probabilities is close to 1), CRT can be low.

Applications: Hick–Hyman law is one of the two laws – the other is Fitts' law – that Card, Moran, and Newell (1983) introduced to HCI as design principles that can be used to improve the usability of user interfaces. However, it saw relatively much less applications than Fitts' law. Why? Because the implication of Hick–Hyman law appears trivial: it states that less is better. If you can design an interface that has fewer responses, users will be quicker responding to it.

However, contrary to a common misunderstanding, the law does *not* imply that interface elements should be organized hierarchically or into pages [Liu et al. CHI2020]. The law predicts that there is a benefit for showing *all* elements at once. Consider the design example in Figure 1.8. Because of the logarithmic term, the best design, contrary to our intuition is Design a). Hence the design principle should be 'more is better'. However, when other factors are considered, the situation is not so simple. There are benefits to pagination and hierarchy, which are not governed by Hick–Hyman law.

Information foraging theory explains that well organized hierarchies help users save time by *skipping* whole sections of elements (see Section X). Visual search time and pointing also start taking time when the number of elements increases.

The second form of the law, which states CRT as a function of entropy H, implies that decreasing uncertainty will improve performance. But how to achieve this in practice? **Stimulus–response compatibility** has a strong effect on CRT. When stimuli and responses are 'compatible', they are ordered or otherwise structured in a consistent way. Some simple mapping exists, for example cues have the same spatial order as responses. This means that the response should be similar to the stimulus itself, such as turning a steering wheel to turn the wheels of the car. The action the user performs is similar to the response the driver receives from the car. Abby Liu and colleagues showed that in HCI tasks where compatibility is high, the Hick–Hyman slope almost flattens out. It is generally desirable to find consistent mappings between stimuli and responses.

Design and training also have an effect on the slope b and intercept a of the model. The Hick–Hyman law governs novice-to-intermediate range of performance. When the user receives extensive practice on responding to the task, the slope diminishes, eventually flattening out. With thousands of practice trials on the same task, response time can be effectively constant when n is smaller than ten [Mowbray].

The scope of the law in HCI tasks is quite limited. When $n > 10$, task completion time becomes dominated by other factors. Experimental research in psychology assumes that CRT is measured in conditions that minimize the effect of visual search and pointing. The n end-effectors are supposed to rest on the n keys associated with the n responses. In applications like gaming, fingers may already rest on top of the relevant keys, thus virtually eliminating the effect of pointing. However, in real-world conditions, when n increases beyond two, CRT may include also a component of visual search (Section X) and pointing (Section X). This happens when the target needs to be found and the end effector moved to select the option. Minimizing the need for searching or pointing will allow expert users to respond faster. This can be achieved by placing keys needed for response selection close to each other, like for example in gaming keyboards and mice. We conclude that the applicability of Hick–Hyman law in HCI is limited. It should be considered only in cases where the user's fingers are already on the n buttons, when stimulus–response mapping is not perfect, and when the user is not an expert

Figure 1.9: In **pointing**, the goal is to move the end-effector (here: tip of thumb) to contact a spatially extended target with appropriate force. Image source: pxhere.com

in the task.

1.4 Pointing

Aimed movements are movements where success is defined by external constraints. In order to elicit a desired message, the movement must be coordinated in space or time in a defined way. The difference to simple reaction and choice reaction is that they make no assumption about movement, only about available options.

Aimed movements are perhaps the most common and elementary type of action in computer use. To elicit the right command, we need to move the body in the right way and at the right time. For example, to send character 'a', a sufficiently small body part must land exactly on the cap of the button with sufficient (but not excess) force. With the exception of brain computer interfaces, most UIs are operated by aimed movements. Two fundamental types of **movement constraints** can be distinguished.

- *Spatially constrained aimed movements* are restricted at the end or during the movement to a specified region or point. *Discrete aimed movements* are movements *to* spatially bounded targets. An example is moving a mouse cursor on top of a button to select it. This movement type is prevalent in HCI, it is in fact one of the prime paradigms used

to communicate intentions and commands to a computer. Consider for example buttons, widgets, links, icons and so on. *Continuous aimed movements*, in contrast, require keeping the control point within a bounding box during the whole duration of the movement. For example, driving a car requires continuously steering it so as to keep it on the lane. We discuss steering and gesturing in the two next sections.

- *Temporally constrained aimed movements* must hit a target defined in time. The target can be hit during a specific interval, or the goal is to be as close to the target as possible. An example of the latter is playing notes on a piano, and and example of the former jumping over obstacles in a video game. Often these two types of constraints occur together. In an *interception* task, we need to catch a moving object by (1) placing a selector on its future path and (2) pressing the button when the object is within the selector's effective region. Consider for example hitting a tennis ball served by the opponent or sniping an enemy player in a first person shooter game.

Pointing is a discrete, spatially constrained aimed movement. The goal is to move the control point on top of an area denoting the target. If the control point ends up missing the area, the movement is considered having missed the target. Typically there are two performance objectives: be as fast as possible and do not miss the target; i.e. keep the error rate below a threshold. While an area target is the most common target type in HCI, three other tasks are studied in literature:

- A point target: The target is defined as a point in space. User's accuracy is measured as Euclidian distance of the end point from the point.
- A line/surface target: The target is a line in space, and the user's goal is to move the control point to cross the line. Performance is measured in terms of movement time. When the target is a line segment (with finite length), accuracy can be measured as hit/miss.
- A postural/angular target: Joints must be rotated to a particular angle.

DOFs in regular HCI tasks are 1 (e.g., slider), 2 (e.g., touchscreen pointing) or 3 (e.g., control in a VR space).

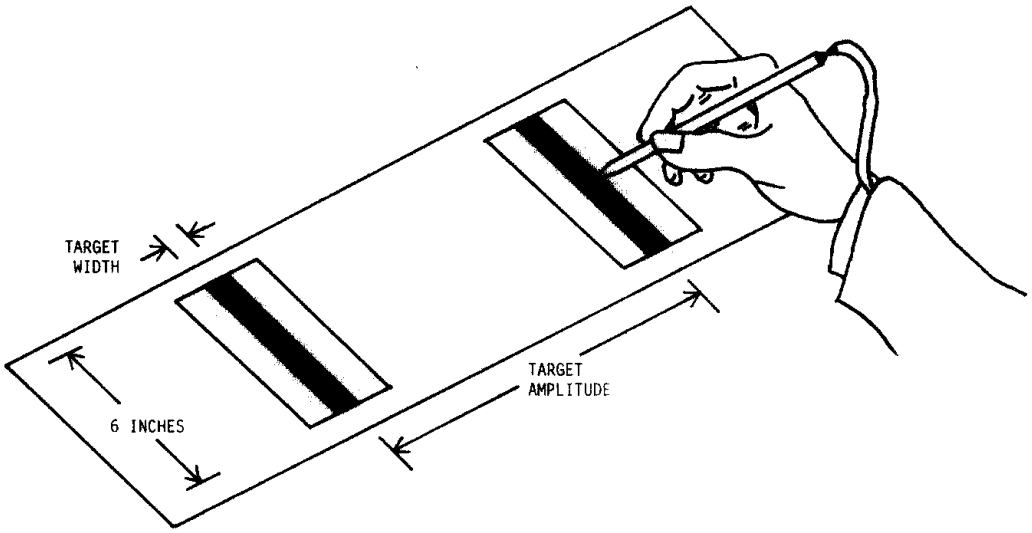


Figure 1.10: The original experimental setup of Paul Fitts manipulated movement D and target width W in a reciprocal tapping task. The two metal plates were to be hit with a stylus in an alternating sequence as fast as possible.[Figure adopted from I.S. Mackenzie 1992, *Human–Computer Interaction*]

1.4.1 Fitts' law

Fitts' law is a well-known predictive model for pointing. Given parameters describing *the pointing task*, in particular distance to target and its size, it predicts average movement time (MT), the time from the beginning of the movement to bring the control point on top of the target. Fitts' law is considered one of the most successful engineering models in HCI [Seow2005;Gori;Liu].

The discovery of the model was a combination of theoretical insight with rigorous empirical work. Paul Fitts was an American psychologists working for the military in the era following WW2. In a series of controlled experiments he asked his subjects to move a stylus in hand back and forth between two metal plates as fast as possible, while measuring this takes. Inspired by Shannon's information theory, Fitts hypothesized that human motor system is capacity-limited. In other words, it could not express more information than the capacity of the brain and nervous system would allow. Any attempt to express more would lead increase noise and therefore compromise the

accuracy of movement.

The key concept was motor noise: "Information capacity is limited only by the amount of statistical variability, or noise, that is characteristic of repeated efforts to produce the same response" [ref]. This observation was insightful. Instead of measuring a single movement, several attempts at the same movement would be observed, in order to understand characteristic noise. Doing this over several conditions would allow characterizing how *noise changes*.

The experimental task, shown in Figure 1.10, is called *reciprocal tapping task*. In the study, Fitts systematically manipulated D and W – distance and width of target, respectively. The found a logarithmic relationship of movement demands and MT:

$$MT = a + bID = a + b \times log_2(\frac{2D}{W}), \tag{1.6}$$

where

- MT is movement time, or the duration of the time it takes from the onset of the movement to when it ends,
- a and b are positive empirical parameters that are case-specific,
- D is movement distance and
- W is width of target.

Fitts' original data from 1954 is given in Table 1.1. It tabulates MT as a function of D and W. The insight that Paul Fitts made is that while there is no obvious relationship between MT and neither D nor W, they can be combined into a single term that does. **Index of difficulty** is a term describing the difficulty of the motor task and given as

$$ID = log_2(2D/W) \tag{1.7}$$

Can you figure out why ID describes 'motor difficulty', why are tasks with higher ID motorically harder? Two illustrating examples are given in Figure 1.11. You can approach this question by thinking what happens when distance increases and when width is decreased. Both increase ID and therefore also MT. In other words, other things being equal, targets that are further away or smaller are harder and therefore slower to point. In other

MT	D	W	ID	Predicted MT
180	2	2	1	107
212	2	1	2	202
203	4	2	2	202
281	2	0.5	3	297
260	4	1	3	297
279	8	2	3	297
392	2	0.25	4	392
372	4	0.5	4	392
357	8	1	4	392
388	16	2	4	392
484	4	0.25	5	486
469	8	0.5	5	486
481	16	1	5	486
580	8	0.25	6	581
595	16	0.5	6	581
731	16	0.25	7	676

Table 1.1: Data for Fitts' original stylus pointing task. Model prediction is $MT = 12.8 + 94.7 \times ID$, where $ID = log_2(2D/W)$.

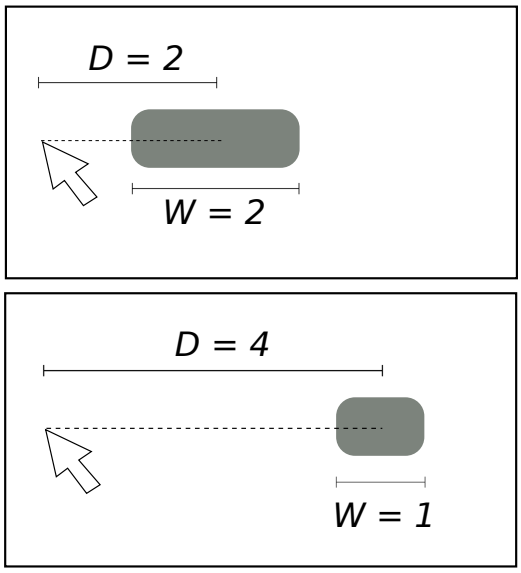


Figure 1.11: Rapid (top) and slow (bottom) selection tasks according to Fitts' law.

words, movement time is related to the inverse of spatial error. ID also makes computations simpler: instead of the non-linear relationship, we can now deal with a simpler, linear relationship.

Fitts' model for the obtained data is shown in Figure 1.12. Averaging MT data obtained in each ID condition, and fitting empirical parameters a and b, Fitts found the relationship in Equation 1.6. After computing ID, parameters a and b are estimated, yielding in this case:

$$MT = 12.8 + 94.7 \times ID, R^2 = .967 \tag{1.8}$$

The fit of the model was unreasonably high for psychological models, and is often around $R^2 > 0.8$ and even $R^2 > 0.9$, for reasons we return to.

ID has a deeper interpretation in information theory. Fitts' was interested in the information capacity of the human motor system. ID represents the number of bits required to select a target with width W within a set of n contiguous regions that evenly partition the interval [0, $2A$]. For example, if $A = 16$ and $W = 4$, then $n = 8$ and $ID = 3$ [Myers88]. Within this interpretation, a is the noninformational parameter, a constant added to every movement. Such constant cost can be due to initialization or termination of the movement. b is the informational factor for ID.

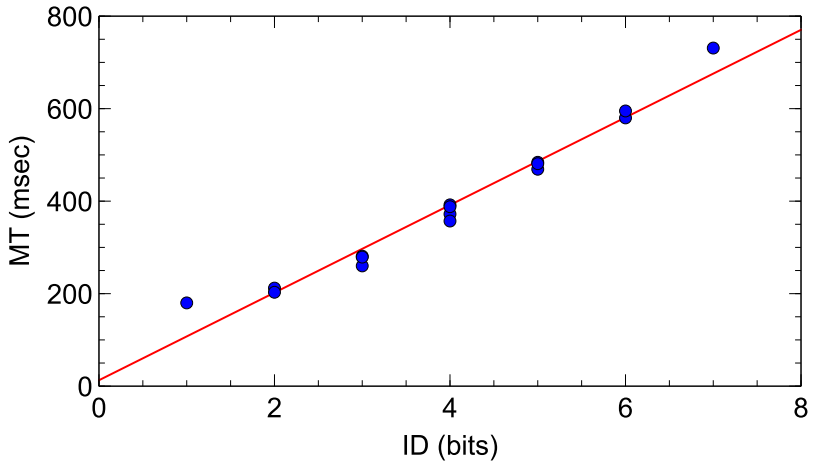


Figure 1.12: A Fitts' law model of stylus pointing. The plot shows observed movement time MT versus index of difficulty ID. The linear trend depicts the model $MT = 12.8 + 94.7 \times ID$. Model fit is $R^2 = .967$. Data presented in Table **??**

Effective width

Analyzing the information-theoretical relation further, Scott MacKenzie and colleagues [] provided a widely used variant:

$$MT = a + b \log_2 \left(\frac{D}{W_e} + 1 \right), \tag{1.9}$$

where W_e is **effective width**, referring to the empirical the spread of end-points around target center. While W refers to the actual width of the target, W_e is the target that users *can* hit most of the time, its *effective* width. You can think of effective width as motor noise that we can measure when a user is repeatedly carrying out the motor task.

Effective width can be computed from observed end points, assuming they are normally distributed. When end-points are normally distributed, standard deviation can be used to determine W_e. The typical cut-off is $\sigma = 4.6$, which amounts to 4 % of end-points. In other words, effective width W_e would here determine the width of the target that would be hit 94 % of time. The idea is shown in Figure 1.13. Because the cut-off defines acceptable proportion of errors, it should be decided case-by-case.

This formulation makes the relationship of the model with speed–accuracy

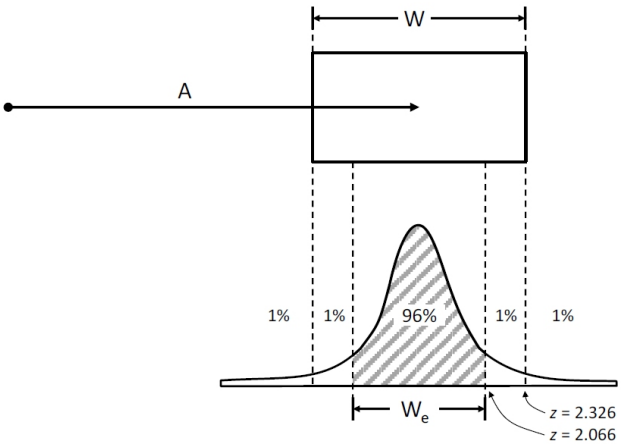


Figure 1.13: Effective width denotes the target size that the user would be able to hit X% of time, where X is usually set to 96 %. Effective width is computed by assuming that end points are normally distributed and setting a cut-off based on standard deviation. Image source: I.S. Mackenzie 1992, Fitts' law

trade-off clearer. If the user tries to be faster, actual target width may not change, but effective width will increase. When ID increases – in other words, the task becomes harder – either noise (effective width) will increase or the user will have to be slower.

Model variants

Fitts' law is a candidate for the most tested scientific model in HCI research. It has been evaluated in a number of user groups, input devices, and contexts, including under water! It has weathered numerous challenges to its theoretical and mathematical assumptions; however, judging from the amount of papers presenting Fittsian models, it is still very much relevant and timely.

An important variant adopted from psychology is the *iterative corrections model* [ref]. The idea is that any pointing movement consists of several *ballistic* movements, in-between of which there are corrections. A ballistic movement itself cannot be modified after triggering it, however the next one can be planned considering sensory feedback. These redirections are 'corrections', thus the term 'iterative corrections'. Detailed kinematic recordings, however, showed only one or at most two corrections. Moreover, considerable variation

has been found in the duration of the initial sub-movement, thus the idea of equal durations is violated.

An extension of the idea was proposed by the psychologist David E. Meyer [ref]. *The stochastic optimised sub-movement model* defines MT as a function of not only D and W, but also the number of sub-movements n:

$$MT = a + b\left(\frac{D}{W}\right)^{1/n}, \tag{1.10}$$

where n is an upper limit on sub-movements. The authors found empirically that $n = 2.6$ minimizes RMSE. Several extensions have been proposed to compute also end-point variability similar to the concept of effective width.

These two models assume *intermittent feedback control*. Intermittent control means that control actions cannot be carried out any time but only after 'locked' periods. In this case, the ballistic part of a motor action cannot be altered, but there is a window of time afterwards to make corrections. The two models assume that such corrections are based on the error at the start of that action. Meyer's momdel also assumes that the neuromotor system is noisy, and that this noise increases with the velocity of the sub-movements. This causes the primary sub-movement to either undershoot or overshoot the target. One known shortcoming of the model is that the number of sub-movements is fixed. For a given D and W, the sequence of sub-movements would always be the same, and it is not possible to explain why the target is missed at times.

Applications

Pointing models are arguably the most successful engineering models in HCI research. They are used for designing better layouts, interaction techniques, and input devices. By exposing how user performance is affected by design-relevant factors, and by offering a unified account of both speed and accuracy, Fitts' law has become the core of our understanding of aimed movements. Fitts' law has also provided a basis for empirical comparisons of input devices [Mackenzie04], and it has driven innovation in interaction techniques [expandingtargets]. As a computationally inexpensive model, it has been incorporated into algorithmic approaches to design. Good examples are the key-target-resizing technique used in virtual keyboards [] and layout optimization algorithms that have challenged Qwerty as the dominant keyboard layouts [BiZhai,Karrenbauer2014].

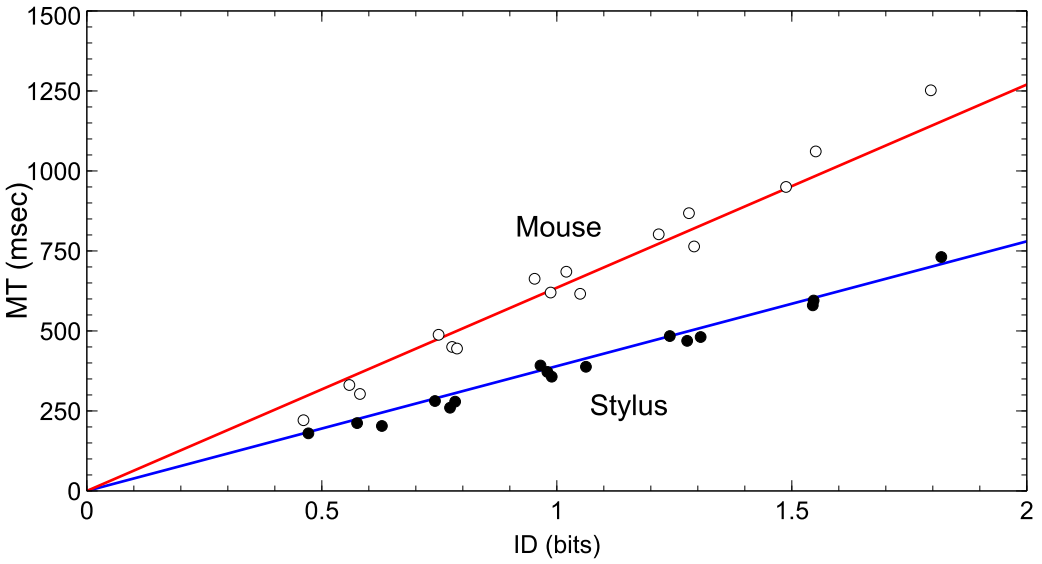


Figure 1.14: Fitts' law models allow the comparison of user interfaces. Here, stylus shows superior performance over the mouse. Data from [Mackenzie92]

Fitts' law permits rigorous empirical comparison of input methods. Consider the problem of comparing two input devices. Figure 1.14 shows a plot comparing data from [MackenzieThesis] for stylus vs. mouse in the reciprocal selection task. The plot shows that stylus is better throughout the ID range, it is thus preferable. However, if there was a *crossover* point, it would be exposed by the model, even if there was no observation at that ID point.

The concept of ID is powerful here. It collapses two parameters describing a motor task into a single variable which is linearly correlated with MT. The alternative, called the naive-but-tempting approach by Shumin Zhai [Zhai2004], would be to measure speed and accuracy in a pointing task. However, comparison would be limited to the selected observation point. Conducting a Fitts' law study invites us to systematically vary D and W, and the model will provide a point of reference – the a and b parameters, for comparing the input methods across the full scope of the motor task.

But Fitts' law is not limited to empirical comparison. Fitts' law can be used *analytically* – that is, prior to collecting empirical data – in two ways:

- Predict mean MT for a pointing task; however, empirical parameters a and b must be known. They can be obtained for example from literature.

- Compare pointing tasks: When a and b are equal, they can be ignored, because MT will be determined by ID only.

ID – the index of difficulty – thus offers a handy entry point to analyzing a motor task. As we learned, increasing ID is associated with increasing MT. All other things being equal,

- An increase in D will increase MT
- A decreases in W will decrease MT
- If D or W changes, MT can be kept constant by changing the other.

However, when no other considerations are in play, implications are trivial: targets should be made as large and close as possible. The situation gets more interesting when there are multiple targets and their respective positions and sizes should be determined. Here, you cannot simply make all targets very large. In Section X, we discuss application in the optimization of layouts and in Section X we discuss interaction techniques exploiting Fitts' law.

Discussion: What Fitts' law can/cannot do

Fitts' law is best viewed as a regression model that *intrapolates* between observation points. However, the obtained model is fragile: Even small changes in the task, user, or conditions can insist on re-obtaining the parameters. When D or W are set to extreme values, many things that affect motor control change: the muscles we use, posture, and visual feedback. The a and b parameters stay stable when such defining conditions are kept relatively stable. The need to collect an extensive dataset for every change in an interface makes it hard to use it in design, although analytical uses are possible, as pointed out above.

Fitts' law is a simplification of what happens in pointing. Because it models mean MT in ID conditions, it effectively hides *variability* other than in end points. Variability is inherent in all motor control, affected by factors like movement strategy [], feedback [], and the involved muscle groups []. Sometimes a reparametrization is not enough, but one needs a different variant with different terms [FFits]. This is frustrating, because there is no a priori principle to anticipate when Fitts' law fits and when it does not. Another criticism is that a mere statistical relationship between design and performance outcomes offers very little insight in user interface design [Mueller2017].

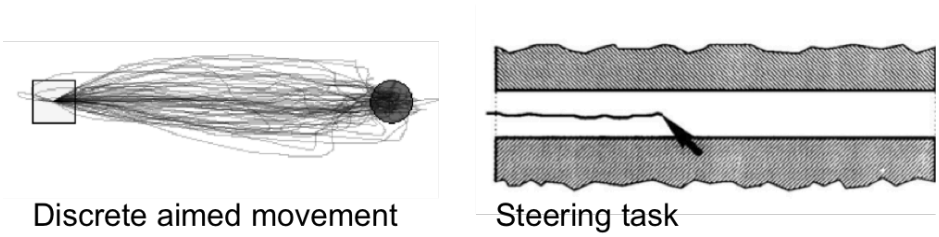


Figure 1.15: In a discrete aimed movement, only the end point of the movement matters: whether it hits the target or not. In steering, the movement trajectory must be kept within spatial bounds, 'a tunnel' of given width.

1.5 Steering

In a discrete aimed movement, only the end point of the movement matters: whether it hits the target or not. In steering, by contrast, the movement trajectory must be kept within spatial bounds, 'a tunnel' of given width. If the control point slips across the bounds of the tunnel, the movement is considered an error. Steering is common in HCI, in menu selection, games, etcetera, there are visible or invisible tunnels within which the cursor must be kept of the event is lost. Figure 1.16 shows an example.

1.5.1 Accot–Zhai steering law

Johnny Accot and Shumin Zhai presented a model of steering at CHI 1999, which has been later dubbed *the steering law*.

They derive the model from Fitts' law by assuming an infinite number of targets on the way between the origin and target. First, assume that there is a single movement from A to B to be made. Fitts' law predicts that movement time will be a function of movement distance and the width of B. Assume now that this distance is split into *two* equally long movements A-B and B-C. Total time now is the sum of the two movements. Then assume that the distance is split into N movements. When N approaches infinity, the authors derive the model

$$MT = a + b\frac{D}{W} \tag{1.11}$$

where W is the width of the tunnel, D its distance, and a and b empirical parameters.

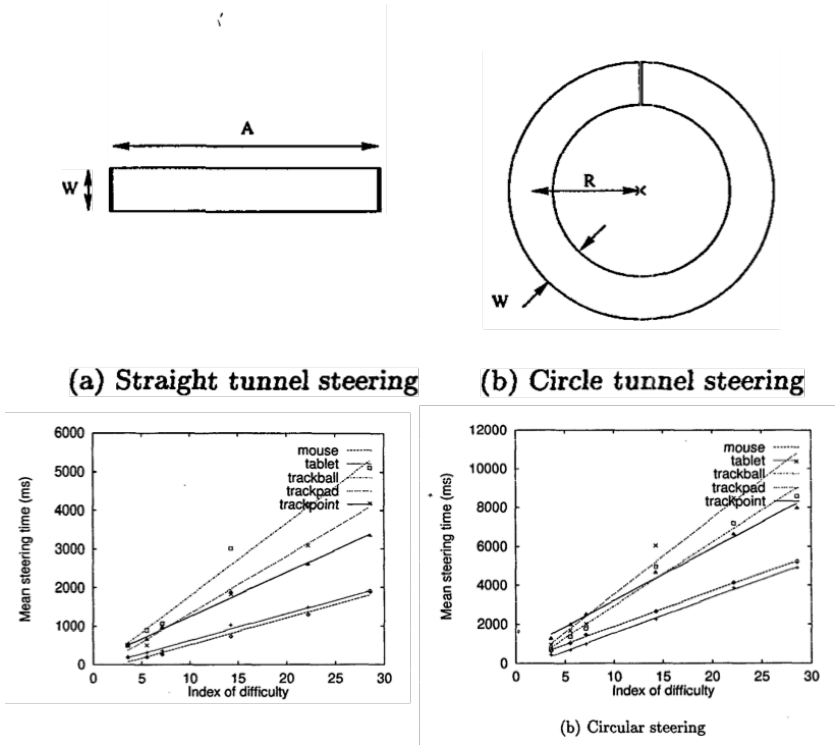


Figure 1.16: Two steering tasks captured by the Accot–Zhai steering law.

The model applies to tunnels where curvature stays constant. When the input device, users, or other conditions change, a and b must be re-calibrated.

1.6 Gesturing

Gestural interfaces, discussed in Section X, are based on continuous shapes as input. Consider for example handwriting as text input: in order for a letter to be recognizable by the decoder, the shape must obey certain segment lengths and curves.

1.6.1 Viviani power law of curvature

The models discussed thus far in this chapter predict task completion time. A limitation of the Accot–Zhai steering law is that it does not account for complex shapes or changing shapes. Consider for example tracing a shape or

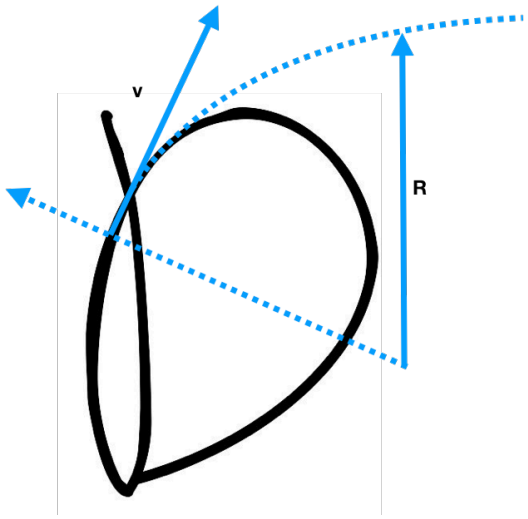


Figure 1.17: Try drawing letter 'd' with stylus or pen. The Viviani power law of curvature predicts momentary velocity v when drawing smooth curved trajectories like this. R is radius of the curvature.

drawing. But what happens *during* movement? We need to understand that in order to understand gestures, where curvature is changing.

The *Viviani power law of curvature* (PLoC) [ref] is a kinematics model for smooth curved trajectories. *Kinematics models* cover aspects of motion during pointing: position, velocity, acceleration, or jerk—however, without consideration of the time-varying phenomena that produce them. They predict the moment-by-moment motion or its properties, such as radius of curvature or tangential velocity at any point. This makes kinematics models useful for modeling gestures.

PLoC pertains to handwriting and drawing behavior, in particular when the trajectories are smooth; that is, they do not have sharp corners. PLoC relates the radius of curvature $r(s)$ at any point s along the trajectory with its corresponding tangential velocity $v(s)$:

$$v(t) = kr^{\beta} \tag{1.12}$$

where k is an empirical gain factor and β an empirical parameter. The model states that the larger curvature, the slower the motion of the end effector will be at that point.

Now, the total time for a full segment S, assuming only smooth curvature

(no corners), can be computed as:

$$T = \frac{1}{k} \int_0^S r(s)^{-\beta} ds \tag{1.13}$$

In a study using stylus as the input device, $K = 0.0153$ and $? = 0.586$. The model has reached high fit with empirical data in drawing, with and without visual guidance.

What happens if the movement is physically larger or smaller? *Isochrony* is the empirical observation that average velocity of movements increases with distance [**?**]. Thus, movement distance is a weak predictor of movement time in a trajectory. Users simply move larger distances faster. The Viviani PLoC has been shown to cover isochrony. The power law -like pattern has been argued to be due to pattern generators in the neuromotor system that operate in an oscillatory fashion [SchaalSternad2001].

1.6.2 Cao–Zhai CLC model

The Cao–Zhai CLC model predicts completion time for complex gesture shapes. Presented by Xiang Cao and Shumin Zhai in 2007, it is based on the assumption that a continuous gesture can be decomposed into components, each of which can be captured by a lower-level model [ref].

- *Curves:* To compute $T(curve)$, Viviani PLoC can be used when momentary R is known. In a simplified version offered by the authros, curves are assumed to have a constant radius: $T(curve) = \frac{\alpha}{k} r^{1-\beta}$, where α is the angle of the curve. *Lines:* Time needed to complete a line depends on the length of the line L. Cao and Zhai offer a power model: $T(line) = mL^n$. People tend to be faster with longer lines, and this is captured in a power-like relationship between production time and line length. *Corners:* Corners contribute only little to total time, around 40 ms on average by the authors. To simplify computations, it can be ignored.

To apply the model, the gesture is first segmented into three types of elements: (smooth) curves, (approximately) straight lines, and corners. Figure 1.18 shows the three types. Total time to perform the gesture is simply the sum of durations spent in those elements:

$$T = \sum T(curve) + \sum T(line) + \sum T(corner) \tag{1.14}$$

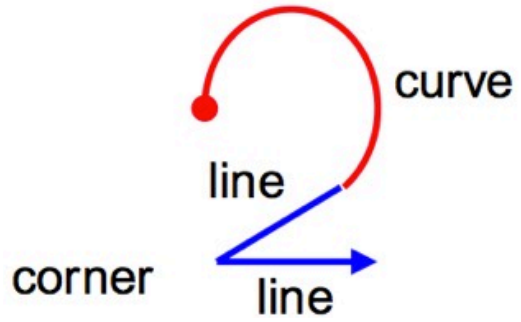


Figure 1.18: The Cao–Zhai CLC model is applied by segmenting a shape into three constitutive types: corners, lines, and curves. We apply a predictive model for each to produce an estimate of total completion time as their linear sum.

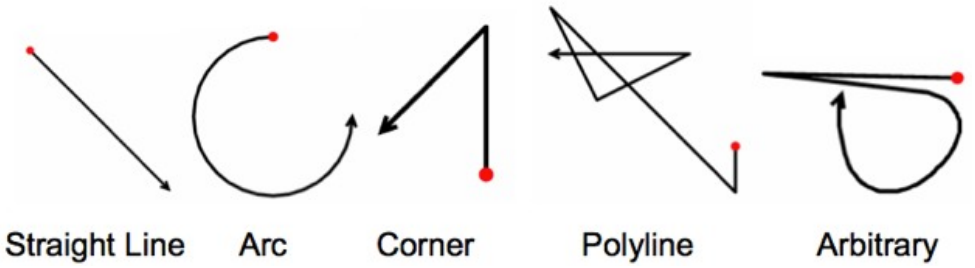


Figure 1.19: Examples of gestural shapes and polylines that can be analyzed using the CLC model

The model achieves a high model fit for tasks where users are asked to draw gestures repeatedly as fast as they can. In some cases it overestimates total time, which is partially attributed to corner-cutting behavior. After initial trials, users learn not to draw the full shape, as asked, but they cut corners, leading to shorter completion time.

1.7 Motor skill

1.7.1 Motor chunking and schemata

1.7.2 Complex motor skills

1.7.3 Typing

1.8 Discussion

Exercises

Exercise 1. Learning to understand motor control problems in HCI takes a bit of practice. Fill in this table. Analyze the motor task of entering text with index finger using a virtual keyboard a touchscreen-based mobile device. How would it change if the input device was changed to a physical keyboard?

	Text entry on a virtual keyboard	Text entry on a physical keyboard
Control point		
DOFs		
Movement demands		
End effector		
Performance objecive		
Kinematic chain		