Dasher Manual

David J.C. MacKay

January 25, 2006 — Draft 1.1

The first part of this manual assumes you are steering Dasher with a mouse, in order to write in English. Dasher also works in over a hundred other languages. Please see the Dasher Special Needs Guide for information about alternatives to mouse-steering.

1 Dasher

Dasher is an information-efficient text-entry interface, driven by natural continuous pointing gestures. Dasher is a competitive text-entry system wherever a full-size keyboard cannot be used - for example,

- on a palmtop computer;
- on a wearable computer;
- when operating a computer one-handed, by joystick, touchscreen, trackball, or mouse;
- when operating a computer with zero hands (i.e., by head-mouse or by eyetracker).

The eyetracking version of Dasher allows an experienced user to write text as fast as normal handwriting - 29 words per minute; using a mouse, experienced users can write at 39 words per minute.

Dasher can be used to write efficiently in any language.

Dasher is fast and fun to learn. (See what users round the world say, in section 8.)

Dasher is **free software**. It's distributed under the same license as GNU/Linux, the GPL.

1.1 How does Dasher work?

Dasher is like an arcade game: 'Attack of the killer alphabets', perhaps.

Financial Times, 5th February 2002

Dasher is a zooming interface. You point where you want to go, and the display zooms in wherever you point. The world into which you are zooming is painted with letters, so that any point you zoom in on corresponds to a piece of text. The more you zoom in, the longer the piece of text you have written. You choose what you write by choosing where to zoom.

To make the interface efficient, we use the predictions of a language model to determine how much of the world is devoted to each piece of text. Probable pieces of text are given more space, so they are quick and easy to select. Improbable pieces of text (for example, text with spelling mistakes) are given less space, so they are harder to write. The language model learns all the time: if you use a novel word once, it is easier to write next time.

A big advantage of Dasher over other predictive text-entry interfaces that offer word-completions to the user is that it is **mode-free**: the user does not need to switch from a writing mode to an “accept-model-predictions” mode.

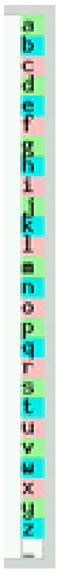
Another advantage is that it is easy to train the model on any writing style: simply load up an example file, then write away!

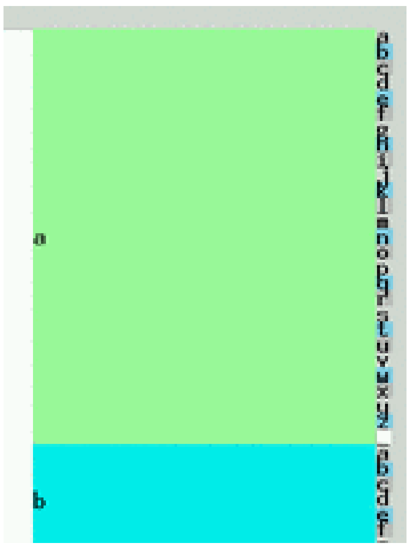
It’s quite hard to convey in words what Dasher looks like, so please visit `www.dasher.org.uk` to see movies.

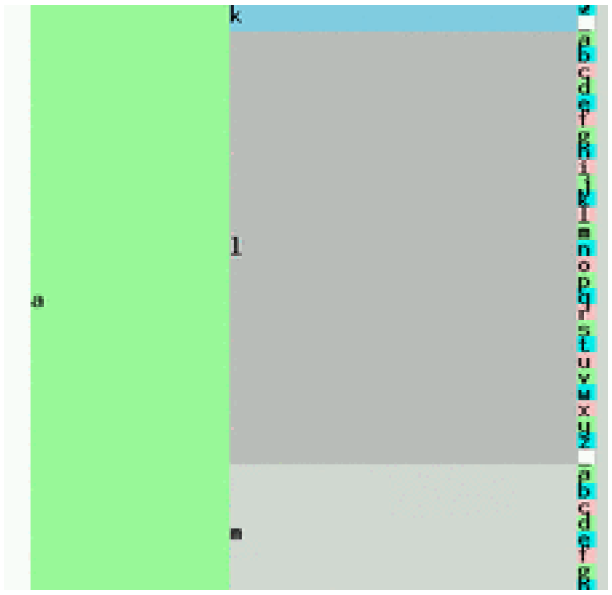
1.2 Dasher explained – the library analogy

Imagine **a library containing all possible books**, ordered alphabetically on a single shelf. Books in which the first letter is “a” are at the left hand side. Books in which the first letter is “z” are at the right. In picture (i) below, the shelf is shown vertically with “left” (a) at the top and “right” (z) at the bottom. The first book in the “a” section reads “`aaaaaaaaaaaa`...”; somewhere to its right are books that start “`all good things must come to an end`...”; a tiny bit further to the right are books that start “`all good things must come to an enema`...”.

When someone writes a piece of text, their choice of the text string can be viewed as a choice of a book from this library of all books - the book that contains exactly the chosen text. How do they choose that book? Let’s imagine they want to write “`all good things` ...”


(i)


(ii)

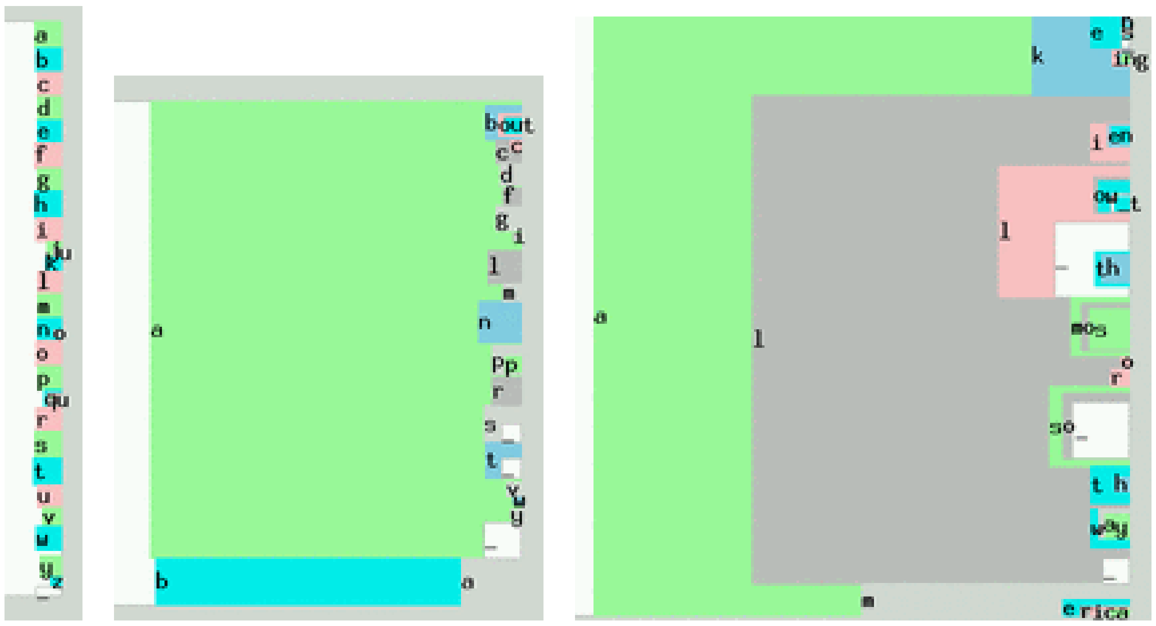

(iii)

First, they walk into the “`a`” section of the library. There, they are confronted by books starting “`aa`”, “`ab`”, “`ac`”, ... “`az`” [Picture (ii)]. Looking more closely at the “`al`” section, they can find books starting “`ala`”, “`alb`”, ... “`alz`” [Picture (iii)].

By looking ever more closely at the shelf, the writer can find the book containing the text he wishes to write. Thus writing can be described as **zooming in on an alphabetical library, steering as you go**.

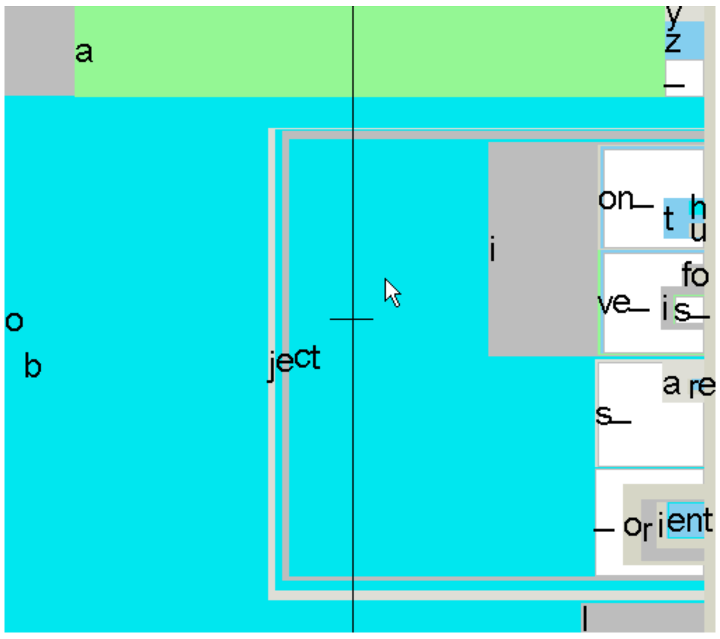
This is exactly how Dasher works, except for one crucial point: *we alter the SIZE of the shelf space devoted to each book in proportion to* ***the probability*** *of the corresponding text.* For example, not very many books start with an “`x`”, so we devote less space to “`x`...” books, and more to the more plausible books, thus making it easier to find books that contain probable text.

Here is the corresponding sequence of pictures of the library in Dasher. (The character ”_” denotes the space character.)


(i) (ii) (iii)

Dasher can be trained on examples of any writing style, and it learns all the time, picking up your personal turns of phrase.

The image below shows the state of the Dasher interface while the user is writing the word 'objection'; alternative words that could easily be written at this point include 'objective', 'objects_', and 'object_oriented'.



If you find Dasher hard to imagine based on these static pictures, please take a look at the movies on `www.dasher.org.uk`.

1.3 Tips for novices

Don't give up if it takes you a minute or two to get started - within ten minutes, you'll be blazing along. It's a lot like driving a car. You should **start by driving cautiously**. If you can't tell where you are going, *stop going*. Indeed, you will probably learn Dasher fastest if you come to it with car-driving analogies in mind, rather than standard computer analogies. For example, the way navigation works is not by DRAGGING but by STEERING: if cars worked like windows computers, you would have to "grab" the piece of road you want, then "drag" it towards you; but in a car, when you wish to drive right, you POINT RIGHT with your steering wheel. Dasher does not work by dragging either. **Do not try to grab things and drag them. Just decide where you want to go, and point there.**

The single most important concept that a novice user needs to understand is that one should always continue *inside* the text written so far: to select the book that contains "all" as its first word, one does **not** enter the "a" section of the library, then exit the "a" section, and enter the "l" section. One enters the "a" section, then finds the "al" section that is *within* the "a" section, then enters the "all" section *within* the "al" section.

It's just like finding a name in a phonebook. To find "`Alison`", you don't go to the "`A`" section of the phonebook, then the "`L`" section: you go into the "`A`" section, then find *within it* the "`Al`" section, and so forth. Once you are in the "`Al`" section, you never leave it.

The second most important idea is that what you have written depends *only* on where you finally end up in the library, not on how you got there; so there is no need to steer accurately on your way to your destination. **You are allowed to cut corners.** (For example, in the previous image, if you wanted to write 'objects_are', it would be fine to move the mouse straight towards the letters 'are', even if this takes the mouse across the unwanted grey 'i' square.)

Common errors. Often, a beginner who is trying to find a particular letter will drive the display forwards fast while hunting for the letter. The rule of the road for Dasher users is just like that for car-drivers: don't drive forwards until you have identified where you want to go! So, after you have found the first letter of your sentence, and zoomed towards it, please **SLOW DOWN and don't proceed any further into this first letter's square until you have figured out where you should be steering towards**. Your next letter *is* there, immediately inside the first square you have entered. The letters are ordered alphabetically. If you can't see your letter, figure out where it must be on the basis of the letters you *can* see. Then point to the right place and enter the second letter's square.

1.4 Example

Imagine you want to write '`I once had a whim`'. You write '`I once ha...`' and the Dasher display looks like figure 1. You want to write '`had`'. What should you do? There are lots of letter `d`s on the screen, and all of them are rather small. The five arrows in figure 1 show some of these `d`s. The purple arrow points to a `d` that we can't see yet, but we know it must be there because we can see '`a`', '`b`', and '`c`' above it.

A common beginner's mistake is to keep rushing forward and spot *any* of these letter `d`s, and zoom into it. For example, figure 3 shows what happens if the user zooms towards the `d` highlighted in figure 2.

If you go in this `d`, you are writing '`I once` **head**`...`'. The other two `d`s labelled by red arrows (in figure 1) correspond to writing '`I once` **heard**`...`' and '`I once` **hedge**`...`'.

It is crucial to understand that there is only one correct `d`, namely the `d` that is immediately inside the blue box corresponding to letter '`a`' in the sequence '`I once ha`'. That blue box is highlighted in figure 4.

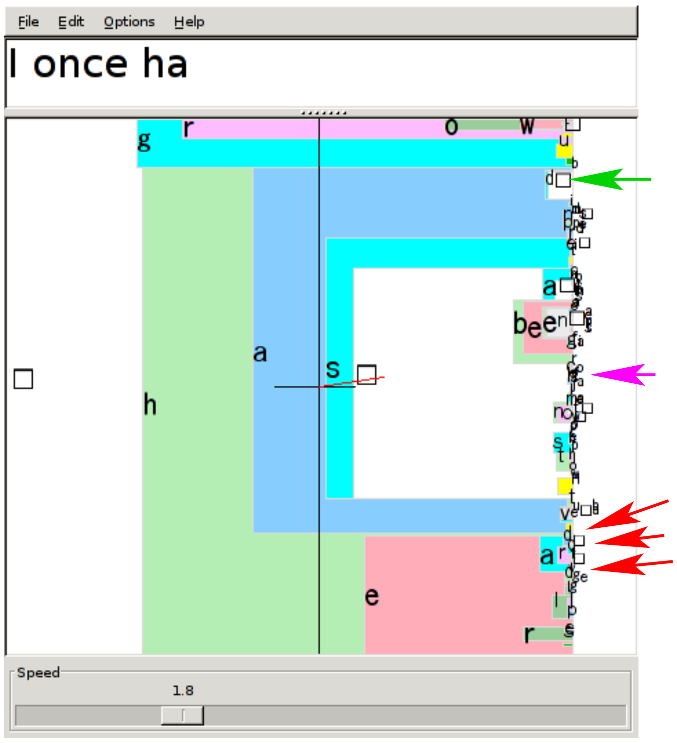
If you ever leave that blue box (as we did in figure 3) then we lose the letter '`a`'.

1.5 Summary

Don't click. Don't drag. And don't speed.

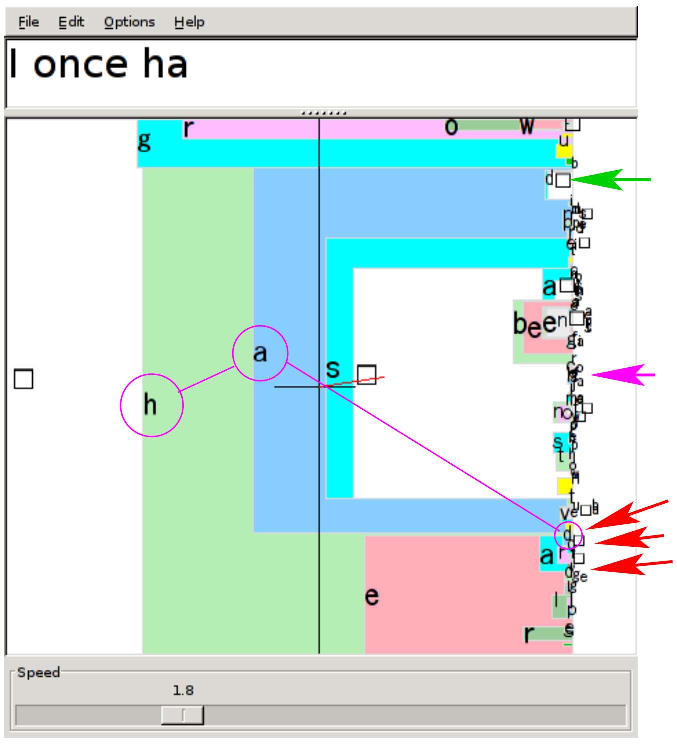
1.6 What do the colours mean?

In the English-language desktop version 3 of Dasher,



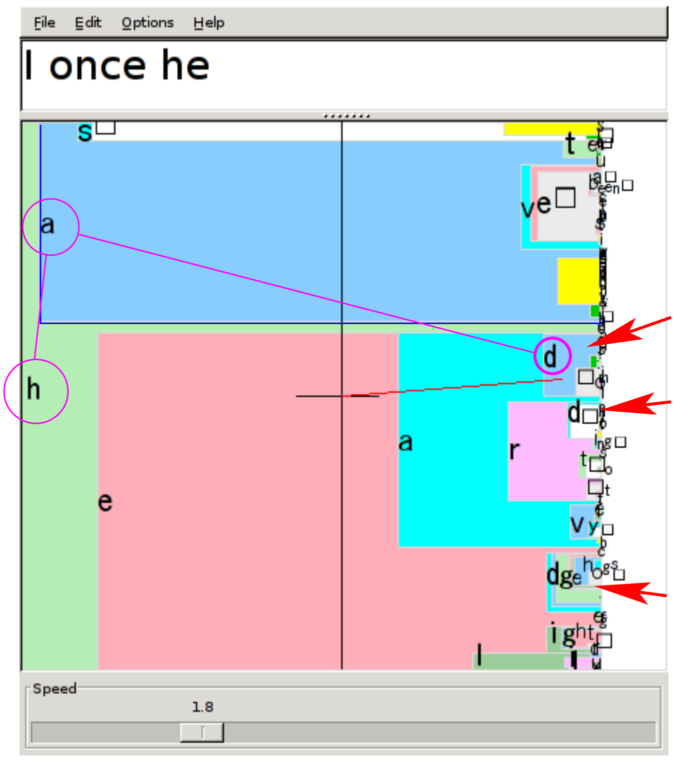
I once ha

Figure 1: Writing ‘`I once had a whim`’. Where should the user steer now?



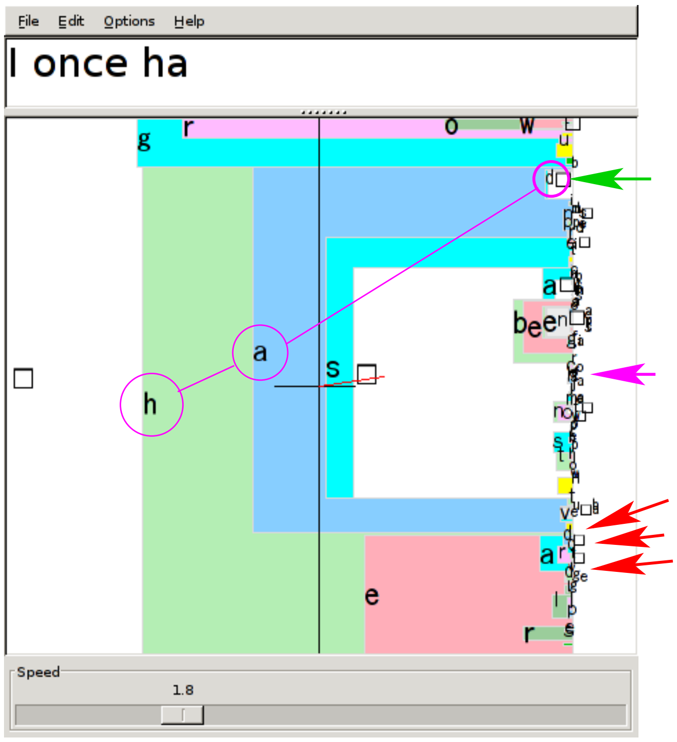
I once ha

Figure 2: Some alternative letter `d`s, with a beginner’s error highlighted.



I once he

Figure 3: What happens when you select the wrong `d`.



I once ha

Figure 4: The correct letter `d` is the one marked by the green arrow. This example illustrates the rule "always keep going *inside* the box that you want" – in this case, the blue box associated with the letter `a` of the word '`had`'.

- **white** squares contain the space characters (always placed at the bottom of the alphabet);
- a special **yellow** box contains the upper case characters [A-Z];
- a **red** box contains numerals [0-9] (if the full alphabet is enabled);
- a **green** box contains punctuation characters (with the characters most similar to the space character [.,;:-] placed at the bottom, next to the **white** space character. Within the punctuation section, letters with similar roles are coloured similarly: punctuation characters similar to the period [.,;:-] are coloured slate-blue;
- the other colours are included simply to discriminate the squares from each other.
- In languages with accents (grave, acute, circumflex, etc), the accents are displayed in an **orange** box.
- In the Japanese version of Dasher, colours are used to distinguish the different hiragana groups (eg ka,ki,ku,ke,ko are orange).
- In the Korean version of Dasher, three different colours are used to distinguish initial consonants, vowels, and terminal consonants.

You can alter the colour scheme by editing the `colour` and `alphabet` xml files, as described in section 3.

1.7 How to start dasher (version 3)

Set the dasher application running; when the dasher window comes up, either click the left mouse button or press the space bar to set it going. [One of these two will work.] Repeat this action (click or space), when you are finished, to stop dasher from dashing. Your computer's mouse controls Dasher.

Adjust the speed slider to fix the maximum speed Dasher will zoom at. A speed of 1 is good for a beginner, increasing to 2 after 5 minutes' practice, and to 4 when you are expert.

1.8 How to start dasher (version 2) on pocket PC

Install, run (wait a few seconds for it to load up the training file), then touch the screen with the stylus to make dasher move.

There are three special strongly-coloured squares in this version of Dasher, coloured yellow, red (in some releases only), and green; all three squares do not produce any character; these squares indicate sub-groupings of the alphabet. (Uppercase Alphabet, Numerals (in some releases only), and Punctuation.)

2 How to use Dasher in real life

Dasher can be used to communicate through your computer, and (to some degree) to control your computer. Obviously, the simplest way to communicate with Dasher is to have the person you're talking to watch the screen as you write. But there are lots of other ways to communicate through Dasher.

2.1 Copy to clipboard

If you have the 'copy on stop' feature switched on, then, every time you stop Dasher, the contents of Dasher's text box get copied directly to your computer's clipboard. You can then use the 'Paste' function of another application to transfer what you've written.

2.2 Speaking

Dasher connects to your computer's built-in text-to-speech system. There are several ways to use Dasher for speaking.

You can choose to have Dasher 'speak each word': every time you pass through the end of a word (by entering a space character for example), that word gets spoken immediately.

You can also choose to have Dasher 'speak on stop': every time you stop Dasher moving, everything that is in the text box gets spoken.

Finally, you can speak exactly when you want to by switching on **Control Mode** (which is under the Options Menu in Dasher version 3). This mode brings up an extra box in the Dasher alphabet, coloured grey, which works a bit like an escape key on a keyboard. If you go inside the Control box, you will find several control-related options: Stop (red); pause (yellow); Move; Delete; and Speak. Inside the Speak box are the options to 'speak everything', 'speak new' (just the new words that you wrote since the last utterance), and 'speak again' (which re-speaks whatever was last spoken).

2.3 Save to file

You can save whatever is in the text box of Dasher as a plain text file by clicking the 'Save file' icon in the icon bar, or using the menus at the top of the Dasher window (selecting File→Save).

2.4 Send text to other window

You can also have whatever is written in Dasher be directly beamed to another window. Enable the 'send text to other window' option under the options menu. Then the window that you select will get Dasher's output.

2.5 How to get Dasher to 'always stay on top'?

In Dasher version 3, this feature is not currently provided. It may be the case in Linux, however, that you can configure your window manager to put particular windows on top, but this will depend on your operating system configuration.

In Dasher version 4, 'always on top' will be possible.

2.6 Font sizes

Dasher has a text box (where the text appears) and a Dasher canvas (where all the fun zooming action happens). You can change the font sizes of both these regions using the Dasher menus at the top of the Dasher window. The two fonts are called the 'Edit font' (for the text box's font), and the 'Dasher font', I think. (I forget the exact menu names in Dasher version 3, sorry.) To change the edit font size bring up the usual dialog box and change size. To change the Dasher canvas size, find the Dasher-size option, which offers three to choose from, named something like: small, medium, and large.

3 Personalizing Dasher

There are three classes of files you can tweak in order to make Dasher work better for you: `alphabet` files, `colour` files, and `training` files.

3.1 Personalizing the language model

Dasher's predictions (in version 3 of Dasher) are based not on a dictionary but on a training text of ordinary text. For example, when you download Dasher version 3, it comes with a file called `training_english_GB.txt`. This is 300 kbytes of ordinary english harvested from various documents on the internet.

When you use Dasher, it stores everything you write in another personal file with the same name as the training file. Next time you use Dasher, it reads in the original training file and everything you wrote last time, to help it predict better. Dasher learns all the time. To get the best results from Dasher:

- If possible, provide Dasher with a training text in your own style – a plain text file made from documents you have written before, and containing your own pet phrases, friends' names, and so forth. Either append this file to the training file, or replace the original training file.
- If you think your personal training file may have become corrupted with rubbish text, edit it using any plain text editor. (Or ask a friend to do this for you.)
- If you use Dasher for many months, the personal training file may become so large that Dasher becomes slow to start up; if so, edit the training file using a plain text editor.

3.2 Personalizing the alphabet

Which characters are available to you, and their order, is determined by the alphabet file. For example, you might use `alphabet.english.xml`. Dasher comes with many alternative alphabets. You can edit alphabet files to change which characters are in the alphabet, or their order. When you edit this xml file, it might be a good idea to save the new file with a new name and change the name of the alphabets in the new file, to avoid confusion. Each field in the xml file specifies a symbol by three items: the character that should be *displayed* (`d=`...); the character that goes into the *text* when this symbol is selected (`t=`...); and the background colour number of the box for this symbol (`b=`...), of which more below.

3.3 Personalizing the colour scheme

You can change the colours of the Dasher world in two ways. The colour file (for example `colour.xml` or `colour.euroasian.xml`) specifies the 200 colours in the palette that Dasher uses. Each line specifies red, green, blue values.

These colours are used to colour multiple objects in dasher. If for example you want to change the colour of the "red line", change the second colour line of the colour file, which reads `<colour r="255" g="0" b="0"/>`.

You can change which of these colours is used for each symbol's box by changing the "b" field for that symbol in the alphabet file.

4 Languages

Dasher works in hundreds of languages.

For each language there is an alphabet file (or possibly more than one alphabet file). On the Dasher website we aim to supply at least one training text for each language. If we don't have a good training text for your language, please help us by making one and sending it to us.

See the Dasher website for the list of languages supported by Dasher. As of October 2005, all the major languages of the world are well supported, with the exception of Japanese and Chinese, for which Dasher version 3 offers only phonetic support (hiragana and pin-yin). In Dasher version 4, we will provide full support for Japanese and Chinese.

[More issues to discuss here: how combining accents work; fonts. Scripts to convert to composed or decomposed form.]

5 What's new in Version 4.0

A preview version, 3.99, was released in October 2004.

There is an automatic speed control, which will speed up Dasher when it senses you are able to go faster.

There is a new socket interface allowing Dasher to listen to head-trackers or gaze-trackers or EEG-interfaces without going through the mouse.

There are several 'button modes' for people who want to drive Dasher using one, two, three, or four switches.

There is a 'click mode', for people who like to point where they want to go, and click to take a single step in that direction. (Good for beginners?)

5.1 What's coming in version 4.2

We will implement a game-mode for Dasher: this will help novices learn to write. A teaching hand will guide the novice when he goes astray, and novices who write fast without needing much guidance will win lots of points.

New language models will be added, which will make predictions at the word level as well as the letter level.

Japanese and Chinese will be fully supported.

The Dasher team also hopes to create a free web-cam-based head tracker and gaze tracker, 'VIM', by April 2006.

Dasher Special Needs Guide

David J.C. MacKay

January 25, 2006

Dasher can be driven in many ways. This guide helps you find which versions of Dasher are most efficient for you.

As of April 2005, the newest versions of Dasher mentioned here are implemented on the Linux platform only. We aim to have them all implemented on all platforms by September 2005.

Dasher is designed on the principle of getting *as much information as possible* from the gestures you can make.

We can get information from whichever of the following is easiest for you:

1. *Continuous* gestures (conveyed via a joystick, trackpad, head mouse, or gaze tracker, for example) often achieve the highest rates of writing.
2. *Discrete* gestures (switches, button presses) may be able to convey information in three different ways:
   (a) The *time* at which you press a button can convey information. (This idea is used in grid systems controlled by a single button.)
   (b) *How long* you press a button for can convey information. (This idea is used in Morse code, where two durations are distinguished.)
   (c) The *choice* of *which* button you press can convey information. (This idea is used in ordinary keyboards.)

6 Continuous gestures

Dasher's normal mode (**mouse mode**) is driven by a two-dimensional continuous steering gesture. Dasher also has a **one-dimensional mode**, for users who can control only one dimension.

Can you make one or two continuous gestures? If you can operate a joystick, mouse, trackpad, or rollerball, then you have a two-dimensional control. If you can point on a touch-screen then that's perfect too. Can you move your nose around? If you can shake your head, that's a one-dimensional control; if you can nod, that's two. A head-mouse can be quite cheap, and it is a convenient way to drive Dasher. (We recommend the **SmartNav3** from NaturalPoint, which costs about $200, and works under microsoft windows only; this device used to be called the NavPoint TrackIR until 2002, when that brand name was transferred to a different device. We also recommend the Origin instruments **Headmouse Extreme**, which costs about $1000; it works as a USB mouse on any computer.) Can you waggle one finger or one foot? These head-mice can be used to track fingers and feet as well as heads. For a detailed comparison of SmartNav3 with Headmouse Extreme, please see `http://www.inference.phy.cam.ac.uk/dasher/Headmouse.html`.

If you are severely paralysed, the best option may be a gaze tracker. Do you have control of where your eyes are looking? With a gaze tracker we can write at 25 words per minute. Gaze trackers are quite expensive: we paid $2000 for the QuickGlance from EyeTech Digital Systems, and the Tobii eyetracker costs about $20,000. [We recommend both of these systems. You attach QuickGlance to an existing computer; Quickglance II costs about $4000. Tobii is a complete computer with built-in eyetracking cameras.] Dasher also works with the Eye response

Erica, with LC's **Eyegaze**, and with Metrovision's gaze-tracker. All three of these systems are complete computers with eye-tracking cameras attached.

If joysticks, mice, rollerballs, and gaze trackers don't work, there may be a few other ways to convey a continuous one-dimensional signal. Lips and eyebrows should both work, though we don't know of any manufacturer selling appropriate devices. Breath is a one-dimensional signal too. If you can control your breath, it should be possible to make a breath mouse for you. We made our $22 breath mouse using a USB optical mouse, a belt, and some elastic, and our most experienced user can write at 15 words per minute by breath alone.

6.1 Starting and stopping

There are several ways of starting and stopping Dasher. Pressing a button (for example, the left mouse button or the space bar) is one option. But if you can not press any buttons, it's possible to start and stop using only continuous gestures: in the options menu, select "start on position"; and switch on "control mode". When control mode is switched on, the Dasher alphabet includes a special Control node (a bit like an `Esc` key on a keyboard), within which various control functions are available. When you are inside the control node, Dasher moves more slowly than normal, for safety. The control node options include 'pause' and 'stop'. Use 'pause' if you are half-way through writing something, and want to pause for a moment. Use 'stop' when you have finished. Pause and stop produce the same behaviour, except stop may cause other automatic actions, such as 'speak on stop', or 'copy the text on stop'.

When Dasher is paused or stopped, it can be restarted using any of the starting methods that are enabled. If 'start on position' is enabled, then whenever Dasher is stopped a sequence of large targets will be displayed; you restart Dasher by pointing at (or looking at) the first (red) target, then the second (yellow) target. (We use two targets in sequence to make it difficult to start Dasher by accident.)

6.2 Recommendations for head-tracking

Many trackers have 'smoothing' options, which determine the frequency with which the mouse position is updated; these options are normally used to smooth and damp down the mouse motion. For Dasher, we don't want such smoothing. We like instant, live, raw and jerky mouse coordinates. If there is a 'smoothing' control, turn it right down.

The 'gain' (sometimes called the 'speed') of the head-tracker is also an important setting to adjust. Some trackers' gains can be adjusted in software. You can also adjust the gain by changing the geometry of your tracker: if you move the tracked dot from your forehead to the brim of a baseball cap, for example, then you roughly double the gain. Sitting closer to the tracker may also increase the gain. Find a gain setting that is comfortable. I like high gain because it allows me to steer with very small head motions.

6.3 Recommendations for gaze-tracking

For good results with gaze trackers, we strongly recommend that the gaze-tracker be made to be as responsive as possible. Many trackers have 'smoothing' options, which determine the frequency with which the mouse position is updated and the number of successive gaze images used to estimate the mouse position. These options are normally used to smooth and damp down the mouse motion. For Dasher, we don't want such smoothing. We like instant, live, raw and jerky mouse coordinates. When you are navigating, your eye moves very quickly to the target you are interested in, and we want Dasher to respond instantly. The ideal settings for Dasher may be very different from the ideal settings for other software. Ask your eyetracker manufacturer to make it easy to change the settings when switching application.

Dasher has several options designed for use with gaze-trackers. We recommend using **eye-tracker mode** (under Options/Preferences/Control). In this mode, the dynamics of Dasher are slightly different from standard dynamics, making error-correction easier by gaze.

If your gaze-tracker's calibration drifts with time, for example when your head moves, then you should select the **Autocalibrate eyetracker** feature. When this feature is switched on, Dasher keeps track of your steering and infers the vertical calibration error, and corrects for it. You can see this correction taking effect by noticing the vertical offset between the mouse position as displayed by Dasher (by the tip of the red line) and the gaze-tracker's mouse position (shown by the system's mouse cursor).

To avoid difficulties with the mouse being bounded by the top and bottom of the screen, we recommend choosing a window size for Dasher that is *not* full-screen in size. Place the Dasher window so that there is a margin above and below the Dasher canvas.

Technical note: In Dasher version 4, we will introduce alternative ways for Dasher to receive tracking information from gaze trackers, head trackers, or similar systems. Rather than sending everything through the mouse, we will enable communication of coordinates, and coordinate-uncertainties, through an alternative socket.

7 I can't use mouse mode or one-dimensional mode

OK, we have several versions of **button Dasher**, which will be available in Dasher Version 4 from January 2006.

7.1 Are time-critical gestures not an option?

Some ways of conveying information make use of the *timing* of gestures. However, some people can't make gestures at a required instant. For example, spastics find it very difficult to do an action 'exactly now!'

If time-critical gestures are not an option, go to section 7.2.

If you *can* convey information by *precisely timed* gestures, go to section **??**.

7.2 'Timeless' choices of Dasher

So, you want to steer Dasher at your own pace. Can you make fairly-accurate continuous gestures, given time? For example, can you position a pointer accurately on a screen, then press a button to indicate that you are ready? Or can you touch a touch-screen fairly accurately?

- If so, try **click mode**. Go to section **??**.
- Otherwise try **direct button mode** or **menu button mode**. Go to section 7.4.

7.3 'Timeless' continuous Dasher: click mode

In **click mode**, you position the mouse pointer where you want to go, then press a button when you are ready. Dasher then zooms in on the position you chose.

Alternatively, if you have a touch screen, a single touch on the screen initiates a zoom to that position. (This functionality has yet to be implemented.)

7.4 'Timeless' choices of Button Dasher

How many different switches, keys, or buttons can you easily operate?

1 With just one button, the only timeless way to convey information is by the *duration* of your button-presses. Can you make a distinction between short presses and long presses? If so, you can use **menu button-Dasher**. Connect up your short press to the 'menu' action, and your long press to the 'select' action.

2 You can use **menu button-Dasher**. Connect one button to the 'menu' action, and the other to the 'select' action. If one button is easier to press, make that button the 'menu' button.

$2\frac{1}{2}$ If you can easily press two buttons, and, for special occasions, you are able to press a third button, you can use **menu button-Dasher** or **direct button-Dasher**.

1. Set up **menu button-Dasher** as described above, and use the third button as your escape key – to make Dasher go away, for example. [This feature is not currently provided within Dasher.]
2. In **direct button-Dasher**, each button produces a particular navigation action such as 'up', 'down', or 'back'. If you have $2\frac{1}{2}$ buttons, map the convenient two to 'up' and 'down', and the inconvenient button to 'back'.

3 You can use **direct button-Dasher** or **menu button-Dasher** as described above.

4, 5, 6, or 7 With more than three buttons, you have the option to use **direct button-Dasher** with three, four, five, or six 'forward' directions. Please try **menu button-Dasher** too, even though it uses only two buttons.

8 or more Try **direct button-Dasher** and **menu button-Dasher**. With this many buttons, you also have the option of using a system like T9 – the predictive-text system found on many mobile phones.

To read more about **menu button-Dasher** and **direct button-Dasher** see sections **??** and **??**.

7.5 Button Dashers that exploit timing

Two questions: How many different switches, keys, or buttons can you easily operate? And can you make switch-closings and switch-openings with equal timing accuracy? That is, can you not only *press* a button at an accurate time, but also *release* it with similar precision?

Whatever your answers are to these questions, you have three choices: **static one button mode**, **one button menu mode**, and **dynamic one button mode**. (We've got ideas about how to exploit two or more precisely-timed buttons, but we haven't implemented them yet.)

If you want both presses and releases to convey information, then select the **two events** option. (Not yet implemented.) The default is that only button presses convey information, and the releases are ignored.

7.5.1 Static one button mode

A pointer drifts down the screen. Press the button when the pointer is alongside your intended destination. If you miss the first chance to click, wait for the pointer to come back. If you need to back up (unzoom), press the button when the pointer is at the top or bottom of the display.

This is called 'static' mode because, as long as you don't press the button, Dasher doesn't move.

This mode has several parameters that you should adjust to get best results. Parameter (name?) controls the rate of drop of the pointer. Parameter (name?) controls the factor by which the display zooms in. Parameter **offset** corrects for your personal tendency to press the button a little early or late. It is measured in milliseconds, and a typical value is 100, which means that you usually press the button 100 ms late.

Another relevant parameter is the (name) parameter, common to several of the button modes, which controls the time taken for each zoom event.

7.5.2 One button menu mode

The display cycles through a sequence of navigation directions. Press the button when the direction you want is highlighted. If you need to back up (unzoom), press the button when the whole display is highlighted.

This mode's menus should be configured as described in section **??**. This mode has one additional parameter, (name?), which determines the speed of cycling through the options.

When choosing the number of menu options and their relative sizes, bear in mind your clicking ability. If you can click very accurately but only infrequently, then it may be most efficient for you to increase the speed parameter, and have more menu options. Experiment!

7.5.3 Dynamic one button mode

(This mode has not yet been implemented; in 2003, a different dynamic one-button mode, also known as metronome mode, was implemented. We believe the dynamic mode described below will be superior.)

In **dynamic one-button mode**, Dasher moves slowly in one direction while the button is not pressed, then changes direction and zooms more rapidly for a fraction of a second when the button is pressed.

In the simplest version of this mode, a second button is used to halt Dasher and to initiate backing up (unzooms).

In a truly single-button version of this mode, the steady zooming-in of Dasher reverses after a short time, then Dasher pauses and enters a temporary menu mode, offering the options to back up, restart, or stop.

7.6 Appendix: Old notes on button-Dasher

The first table summarises the planned versions of button-Dasher and suggests a radio-button menu layout.

Button Dasher Menu

	No Timing					Timing		
# Buttons	1sl	2	2sl	3		1d	1sl	1du
menu mode	◎	🟡		🟩	static	🟡	🟩	🟩
direct mode	□	□	🟩	◎	dynamic	◎	🟩	🟩

Key	
sl	short and long presses used
d	only timings of down events used
du	timings of down and up events both u
🟩	an available method
🟡	favourite method
◎	runner-up method
□	this choice is available, but has defect

Options for Menu mode	
B	Number of boxes
$\mathbf{p}$	The probabilities $(p_1, p_2, \ldots)$ (allow to select uniform $\mathbf{p}$, or dial up a geometric series)
$1/p_{\max}$	unzoom factor
s	cushion parameter

Options for Direct mode		
$\mathbf{p}$	The probabilities (p_1, p_2) (allow to select uniform $\mathbf{p}$, or dial up a geometric series)	[1sl, 2 only]
$\mathbf{p}$	The probabilities – with $p_1 = p_4,\ p_2 = p_3$	[2sl only]
$1/p_{\max}$	unzoom factor	[3 only]
s	cushion parameter	
1/0	Whether to show the two options by dividing line (default) or by boxes	[1sl, 2, 3]

Options for static mode	
ϕ	zoom-in factor
E	time taken for pointer to do one sweep

Options for dynamic mode	
Speed	(uses the normal speed slider)
Rotation rate	**as a multiple of speed**
…	additional parameters associated with long presses (1sl only)

Options for all modes	
Button 1	Definition
Button 2	Definition
Button 3	Definition
Frames	How many intermediate frames to render when a click initiates a 16x zoom.
s:l boundary	time defining short:long
1/0	Whether to show top and bottom of the 'official' canvas

The 13-option radio-button menu could be included in the current Control menu. All the buttons options are mutually exclusive alternatives to Normal Mouse Mode, One-dimensional Mode, and Eyetracker Mode. Alternatively, we could have a single option, 'Button mode' sitting in a four-way radio button: Normal Mouse Mode, One-dimensional Mode, Eyetracker Mode, or Button mode; then a separate 13-choice radio-menu (at the bottom of the Control menu) would be used to specify which of the button modes the user wants. The other options could all go in Advanced, in principle, but it won't be big enough to hold them all. So I think we need to replace Advanced by, or divide Advanced into, **Miscellaneous**, **Buttons**, and **Model**. Model is where we put the radio-button to choose between various language models, and the Smoothing slider. Miscellaneous gets Timestamp, the OneDimensionalMode slider, and the Start-on-mouse-position slider. Buttons gets everything in this document. While we do this, I think we should put the "Control Mode" switch into the "Control" menu, under Starting and Stopping.

Here is an alternative orientation for the button menu.

Button Mode

	No Timing			
Buttons	1sl	2	2sl	3
menu mode	◉	🟡		🟩
direct mode	□	□	🟩	◉
	Timing			
Buttons	1d	1sl	1du	
static	🟡	🟩	🟩	
dynamic	◉	🟩	🟩	

8 User feedback

Marc says:

When I saw the video of Dasher in action, I was blown away. This is really exciting stuff. The idea of it goes beyond what many have tried to solve. I quickly downloaded it to a Pocket PC and began showing it to all of my friends at work. They were all impressed and everyone has had fun playing with it. I work in the console games industry, and I highly recommend you pursue developing this for console game machines that only use a joystick for an input device. This would solve quite a few problems that industry is trying to solve with, text messaging in multiplayer games that don't have keyboards, and would be even greater on wireless phones with instant messaging. Very nice indeed. I've just seen part of the future.

Lori says:

Wow, what a fantastic program! I have Spinal Muscular Atrophy which causes progressive weakness in my fingers and wrists. I have tried a variety of alternative input methods and the Dasher is one of my favorites. Great job!

I'm using it on my Windows XP desktop system and also on a HP iPaq 6315. It works great on both systems.

'Kiwi' says:

Okay, so here I am, a disabled programmer. I just discovered dasher a few hours ago, and I'm in love. I desperately need dasher, and I'm willing to do all I can to help.

and then one day later,

Im writing this email in dasher. In only an hour or two of use Ive gotten pretty good at it. Im pleasantly surprised because Ive had so much trouble with voice recognition I had despaired of ever being able to program again. Now Im confident that I can do anything I want in a matter of months.

FYI Im using Dasher with an alternative mouse called the Foot Rat. It works quite well and I would recommend it to anyone with a hand disability.

Bryan says:

I've been using Dasher now for a few months. I depend on it for all of my communication, written and "verbal," as I have lost my ability to speak to ALS a few months ago.

Congratulations on the development of a great product.

'yogi' says: `:-)`
`very impressed :-D -`
looked a bit clumsy to start with but after a minute of using I had already attained a respectable speed (Once i'd got the principle)

Steve says: I am fighting ALS (MND) and words can not express how grateful I am to have found your Dasher project!

Thank-you, Thank-you!

Andrew says: I haven't yet decided if Dasher's key feature is that it's fast and practical, or that its lots of fun!

Daaf says: I am speachless.. seeing the animations / movies / screenshots I thought it was utter chaos... working with it has proven to be so simple though...

WOW!

"pedrodawa" says Breakthrough !

I am not kidding... You guys are starting something that can make keyboards things of the past. Just take your time and do it right. If this is the prototype, I can hardly wait for the real thing. This is amazing stuff !!!

Mark says: Excellent program. This could really change the way we use handhelds – beats Graffiti to the ground, that's for sure.

NJovich says: I tried the binary and was quite surprised just how easy it was to get into. You can slow down by moving your mouse to the right and speed up by moving it to the left. And that works rather intuitively, I like it. I'd really love to see this in game consoles where text-input can be a real pain even today.

Another slashdot poster says: I downloaded the software, tried it out, and after two minutes, I'm impressed. MUCH easier to use that I thought that it would be, It almost seems to be reading My mind, as to what I want to say...

While it could still use a bit of work, overall it is an incredible new paridgm in the way that text-entry can happen. palm-top users are going to fall in love with it's ease of use, I predict.

A more 'finished' version would be nice for the desktop users, perhaps allowing it to reside in a side window tray that scrolled out when selected, and did the text entry in whatever text box has the edit focus. Add that, and it will become a permanent addition to My desktop's.

Kudos to David MacKay and his crew for creating something unique and new, and actually enhancing the user interface at the same time.

And another: Download this, and give it a serious try. More than 20 seconds. If you try it for five minutes, you'll see the power of it.

It's amazing how quickly you can pick up the basics (unlike Graffiti and other handwriting techniques).

It's a start of something great, I think.

I think this is one of the few areas where software patents actually make sence (I assume the people who made this have got some kind of patent for it). Unlike a lot of examples of software patents this is 'non obvious' and (as far as I know) is not a simple extention of someone else's work. Perhaps this could be used as a standard to judge other software patents: If something does not achive this standard of idea it should not be patentable.

Chris says: I just wanted to say congratulations on Dasher - it is a truly incredible program! ... I have passed on website links for Dasher to a school for disabled students because I believe you have a system here wich could be incredibly useful to them!

Congratulations on your idea again!! It makes a fantastic change to typing!!! I am now running it on my PC and an IPAQ :)

Margaret says I work with people who have ALS. I already have a few clients who are using Dasher, who feel it's faster than their current input method.

June 2003, a slashdot user, trying out the new Mac version:

I am using Dasher to write this, after having never heard of it before today. I think that it is the most interesting piece of software I have seen in the last year.

It works better than any software has the right to. The interface works really well, and today marks the first day in a while that I have not worried if my job is giving me carpal-tunnel, because I know that even if I lose my ability to type, I will still be able to be productive as a software developer!!!

Plus, it's fun!

Shaolin wrote I saw this at the Linux expo today ...

It was one of the few things that pretty much blew me away.

It also allows you for example using Gnome's accesibility layer to input into other applications and control the formatting as well using Dasher alone.

Alex Churchill writes

A satisfied customer joins the group

Hello, Dasher team and users!

My name is Alex Churchill, and I've been using Dasher as my primary (near-exclusive) text entry system for about six months. This is due to computing-related injury (RSI, specifically tenosynivitis): I suffer significant pain using either keyboard or mouse for any length of time, but my job (and leisure activities) include large amounts of computer use.

Thankfully, I'm able to compose any amount of emails, scripts, and even do a certain amount of programming, using a graphics tablet instead of a mouse, and Dasher as a main keyboard replacement (using the Windows XP On-Screen Keyboard for occasional keystrokes, and an excellent program called RemoteKeys for commonly used words and DOS and Unix commands).

I currently use Dasher at speed 7.7, and am expecting to max out the speed slider shortly ;) MANY THANKS for producing a program without which I literally wouldn't be able to do my job!

Best wishes,

Alex Churchill

VB says: Thankyou. From all my students.

I have recently started teaching students of various ages with varying forms of disabilities. Although a few can read and write, spelling may be an issue, but use of the keyboard was very slow and usually caused a bit of frustration with certain students.

A colleague mentioned Dasher and now my session has been transformed from a time consuming chore for the students, to one of fun with lots achieved by the students with regard to writing.

I admire the fact this has been distributed free as I know a lot of the centres I teach at are always low on funding. This little program is a work of art, I really don't know why this isn't more widely known throughout the special needs circles, let alone anywhere else.

Again Thankyou all involved with its development and I wish you long and prosperous lives. Wed 27/4/05

Tiago Guerreiro says:

Myographic Dasher Control

Hi,

my name is Tiago Guerreiro and I am a MSc student at Technical Superior Institute at Lisbon, Portugal. My research is in the multimodal interfaces area and its relation with acessibility. I am working with electromyographic devices to capture muscle activation. In my current prototype I can control a mouse pointer with neck side movements or forearm muscle contractions. I tried my prototype with Dasher and I can write very fast and accurately. It worked great to show my results. And I didn't have to touch any Dasher source code....I launched operating system mouse events so I can interact with all the applications in my computer. I thought you would like to know about another way to interact with Dasher. With neck movements this can be used for tetraplegic individuals, for example.

Your work is great... Thank you. Tiago Guerreiro