Design creativity with maths:
Designing Gadgets

Professor Harold Thimbleby

Introduction to the 2002/3 Geometry Lecture Series

Gadgets like mobile phones and car navigation systems are often difficult to use. Most of the time we cope, trying to ignore their more confusing and specialised features. Yet occasionally, in some situations, trying to use a complex system can be dangerous and expensive, not just tedious.

A traveller trying to use an automatic ticket machine has to find their destination, ticket class and type of ticket, enter cash, confirm, collect the tickets and change -- all under the pressure of having to catch the train on time as well. People under these circumstances make predictable mistakes, such as leaving their change behind. Such errors can be fixed by changing the design.

Trying to use a mobile phone, or even the radio or navigation system, while driving a car can be so distracting to be dangerous. Ideas like voice control are not going to change the underlying complexity, as anyone who has been frustrated by telephone voice menus will know! Deeper ideas are needed.

A nurse using a syringe pump to provide automatic drug injections is under extreme pressure to "do the right thing" in a distracting environment, yet errors can have unfortunate consequences for patients. Errors caused through ignorance about how to use the equipment may be reduced by better training, but skill-based errors can only be reduced by better design. In fact, most user ignorance can be better dispelled by simplifying designs than by more training!

The 2002/3 Geometry lectures will explore the underlying theories of system design, so that we can see why things are difficult to use, and how they can be made better and easier to use. The slogan for the series of lectures is "design creativity with maths."

The maths we cover is simple stuff and surprisingly effective in leading to improved designs, thus helping make systems much easier and more reliable to use. I am not aware that these important mathematical techniques are used by industry, but I hope these lectures will show how easy and useful they are.

The first lecture introduces graphs, mathematical objects that are basically just dots and arrows (and therefore easy to draw and understand). We can imagine web sites to be graphs: the web pages are the dots, and the links between the pages are the arrows. We can also imagine devices like mobile phones to be graphs. Immediately, any phone is like a web site. Which means, more constructively, that we can simulate and test one on the web very easily -- or we can write its user manual as a web site. Such a manual would be complete and correct -- unlike most real mobile phone manuals! We can do lots of other things with graphs, like measuring how long it takes to get across them, and this gives insights into design -- how long would a user take, and how can we redesign things to be easier?

Later lectures will look at how matrices can be used, how codes (e.g., the maths of codes like Morse code) can be used, and how symmetry can be used. The lectures will give plenty of examples that out-perform proprietary products.

Because of their creative and practical design element, all the lectures will appeal strongly to designers and manufacturers, particularly those designing or making highly interactive products such as mobile phones, car radios, and even aircraft cockpits. But the lectures will also appeal to the rest of us, everyday device users -- we who have to put up with using ticket machines, photocopiers and mobile phones... In all lectures, the maths is not difficult and you will go away able to do some device design or analysis of your own. You'll see why these things are hard to use, and you'll wonder why industry does not use maths to make the world easier for us.

Summary of the lecture demonstrations

The first lecture showed how devices (such as mobile phones, video recorderrs, and even games) can be simulated on a computer, and showed how various sorts of manual and description can be generated automatically, for instance to provide a web site documenting how to work the devices. In turn, these sorts of description can be converted into complete and correct user manuals. We also showed how code, as device manufacturers might need to actually build a real, fully-working device can be generated automatically.

For technical people who missed the lecture: the demonstrations showed several finite state automata, how they could be modified and simulated, how their structure allows usability questions to be answered (as well as intelligent help provided), and how they can be used to generate HTML for web-based manuals or Javascript for executable implementations. The FSAs were defined in Mathematica code, but the simulations were done in a program I wrote (which parsed the Mathematica).

The lecture will be recorded on the Gresham College web site, and you will be able to get the video demonstrations from http://www.gresham.ac.uk under the Geometry lectures.

Overview of these notes

These notes show how Mathematica can analyse descriptions of interactive devices, by writing text, drawing pictures, and by doing numerical analyses. Mathematica is a sort of mathematician's word processor: all the text and pictures here were created in it, and all the information shown about devices, whether pictorial, numerical or textual, was worked out by Mathematica from definitions of the devices. The results have not been touched up: everything is automatic, and the same sorts of results could be worked out for other devices as desired -- for instance, if a new device was being designed. The descriptions of devices used here are ones that we demonstrated working in the lecture. Exactly the same definitions can also be used for generating user manuals, intelligent help or for building complete systems.

These brief notes don't exhaust all the possibilities, of course.

Since some people reading these notes will not be interested in any details of how Mathematica works; the Mathematica instructions themselves have not been printed. (Anyone who wants the Mathematica code can email Harold Thimbleby for it.)

A very simple device

The first, and simplest, device we'll consider is a simple light bulb, with three states: off, dim and fully on. This can be drawn as a 'transition diagram' with three circles, one for each of the states, and with arrows between them showing how one could change the state. You can think of the diagram as a game board: when you press a button, you move along the right arrow to a new circle (for clarity I haven't written down arrow names). As it happens, this bulb allows any state to go to any other state directly, so every line is a double-headed arrow -- but this is rarely the case with more complex devices.

Most device descriptions are quite big, but the light bulb is simple enough so that we can show it in its entirety. You can see below how Mathematica has got names for the states, descriptions of how it works, and how to draw it on screen for working simulations.

Professor Harold Thimbleby [3]

Location

Gresham Professor of Geometry [4]

Media content

Video
Audio
Transcript
Computational mathematics [5] Mathematics [6]

Computational mathematics Mathematics

Links:
[1] http://blip.tv/file/get/GreshamCollege-Designing_Gadgets857.MP4
[2] http://blip.tv/file/get/GreshamCollege-DesigningGadgetsProfessorHaroldThimblebyGreshamCollegeL608.MP3
[3] http://www.gresham.ac.uk/professors-and-speakers/professor-harold-thimbleby
[4] http://www.gresham.ac.uk/gresham-professor-of-geometry
[5] http://www.gresham.ac.uk/category/lecture-categories/mathematics/computational-mathematics
[6] http://www.gresham.ac.uk/category/lecture-categories/mathematics