Layout Rules for Graphical Web Documents

Jan Borchers Oliver Deussen Arnold Klingert Clemens Knörzer

Abstract

The number of companies, institutions, and individuals competing for attention in the World-Wide Web is growing exponentially. This makes designing informative, easy-to-grasp, and visually appealing documents not only important for user-friendly information presentation, but also the key to success for any information provider. In this paper, we present layout guidelines for textual and graphical, static and dynamic, 2-D and 3-D Web documents which are drawn from fields as diverse as typography, Gestalt psychology, architecture, hypertext authoring, and human-computer interaction. Web documents are classified into five basic types, and our layout rules are applied to each of these. Finally, we show how currently evolving standards (HTML 3.0 for text and still graphics, Java for 2-D animation, and VRML for 3-D worlds) support applying those rules.

1 Introduction

Whenever a new information-conveying technology is invented, it usually takes many years until authors develop new media that make use of the technological advantages suddenly available. The printing press, for example, was a quantum leap only from a technological point of view; new media, like the novel or the newspaper, still had to be designed [Kay95].

A similar process can be identified in connection to the Internet: After its basic technology became available in the early seventies, it took a while until services like email, electronic bulletin boards, and ftp developed into new media. It took another twenty years until the World-Wide Web [Berners94] as a new medium supporting hypertext functionality revolutionized the Internet. And today, with graphics and network technology being at the verge of making navigable, three-dimensional, multi-user environments widely available, new media, or usage forms, for those new worlds are still widely missing.

An important defining attribute of a new medium, apart from its type of contents, is its *layout*, or appearance, and, in the case of an interactive medium, its *behaviour*. Speaking in computing terms, we could say that a medium is not only defined by the type of information it tries to convey, but also by its *user interface*, or *look+feel*. Type of presentation and level of attractiveness often decide whether a document succeeds or fails to convey its contents: In 1967, the well-known media philosopher Marshall McLuhan already declared that "the medium is the message" [McLuhan67].

In the case of Web documents, this means that graphically pleasing, clearly laid-out, and easily navigable documents will attract and keep many more readers than others. In the early days of the Internet, most users were academic, often from the field of computer science, and generally willing to accept an ugly interface as long as it supplied them with the information they needed. Today, however, the majority of Web users are not computer professionals already. Commercial services are beginning to create Web sites of outstanding quality that, in return, attract enormous amounts of visitors each day. These developments have made advertisements in Web documents no uncommon sight, which will soon make the creation of successful Web sites a rewarding business.

This competitive situation between Web information providers is intensified further by the "surfing syndrome": Most people come across a page because they selected some hyperlink that pointed to it while surfing the Web – if the page does not appear interesting enough, it is left again quite quickly, and the user goes back to where he came from, to find somewhere else whatever he was looking for (information, entertainment, etc.). Or, as it has been put in a recent article about information design for new media: "Users Are Cruisers" [McAdams95].

In this paper, we will give some guidelines for the design of textual and graphical Web documents that have this capability of attracting and keeping readers. Instead of reinventing the wheel, however, we will show that other fields of science and arts, such as typography, architecture, psychology, human-computer interaction, etc., offer rich resources addressing our layout problems. Nevertheless, we will see that many adaptations to those rules are necessary before they can be applied to interactive, on-line documents.

We will sort our treatise on new media in increasing order of their difference from traditional ones, starting with simple on-line texts, followed by animated interactive pages, and leading to 3-D virtual reality presentation forms.

Of course, the contents of, or intention behind, a document should have a strong influence on its graphical design – this is why we will classify online documents into a number of types, depending on their intention, before trying to apply our layout rules.

1.1 Overview

The rest of this paper is organized as follows:

- Section 2 will give an introduction to the field of cognitive psychology known as *Gestalt Theory*. It explains how human perception processes visual patterns, and how this knowledge can be used to create layouts or environments that are easier to operate and remember.

- Section 3 gives an overview of design rules for conventional printed documents from classical printers' typography.

- Section 4 presents layout rules for Web documents, and our classification of Web pages into five basic types. We also show how different intentions behind pages influence their optimal layout.

In the second part of this paper, we will take a closer look at those standards which have evolved recently for the creation of various types of on-line documents, from static graphics and text, via 2-D dynamic pages, to 3-D Virtual Reality worlds:

- Section 5 gives an overview of the layout features *HTML* in its latest version 3.0 [Raggett95] supports for the design of non-animated, 2-D documents, and how they simplify creating visually appealing pages with text and simple graphics in them.

- Section 6 leads from static to dynamic documents. It presents the *Java* language environment [Gosling95] which, in connection with enhanced Web browsers like *HotJava*, supports integrating executable Java programs (*applets*) into Web pages. That way, almost arbitrary programs may be executed on the browser side once a Java page has been downloaded, bringing animation and local interaction into Web pages.

- Section 7 finally takes the step into the third dimension.,It discusses some of the problems inherent to the layout of 3-D environments, and how rules from 2-D can be extended to cover the three-dimensional case. As the most prominent example, it examines *VRML*, the Virtual Reality Modeling Language [Bell95], a description language for three-dimensional object worlds that has been designed with low-bandwidth transmission in mind. It is becoming the standard language in which virtual realities are created within the World-Wide Web framework.

Each standard is briefly presented, and its usefulness for the creation of Web documents with high-quality layout is examined.

The paper concludes with a summary about the current state of layout rules for Web documents, and an outlook on the additional support that will be required to create aesthetic documents in our on-line world of quickly changing standards.

2 Fundamentals of User Interface Design: Gestalt Psychology

Whether to design a good user interface, Web page, or 3-D environment, we always have to take a look at some basic questions first: How do we perceive objects on screen, on paper, or in the real world; and according to which principles does our perception combine objects into groups?

Those questions have been treated by a scientific field called "Gestalt psychology", founded by Köhler, Wertheimer, and Koffka in the 1920's [Koehler29]. The applicable essence of their research can be summarized in more than hundred "Gestalt laws" that explain, for example, why certain patterns are considered as belonging together.

We will present some of those laws here (see also [Schmitt94]) that may be most important for the design of on-line documents and user interfaces. The first four laws deal with static objects, while the last law treats the dynamic case. This last law is especially interesting when designing VRML or Java applications.

- **Law of Succinctness**
  This rule, also called *Law of Good Shape*, states that perception tends to see objects as having some perfect or simple shape because this makes them easier to remember. For example, a polygon with many edges which is "almost round" is often perceived as being a circle when looked at for a short period. In a way, this transformation is essentially a built-in lossy compression algorithm of our memory: associating the already existing notion of a circle takes up less storage capacity than remembering the complex shape we actually saw (cf. Figure 1(a)).

  A practical application of this law is that interface objects like buttons should have a simple shape. Figure 2 presents some alternative shapes for buttons. If the shape is too simple, the user does not associate information with it, whereas an appropriate shape is remembered well. If the shape is too complex, however, the user might forget its meaning and the corresponding functionality.

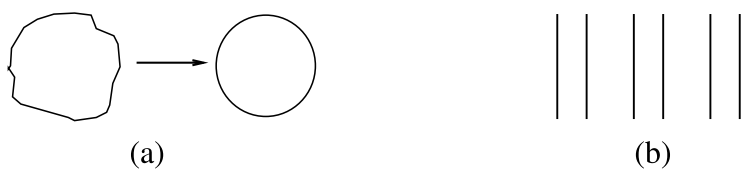
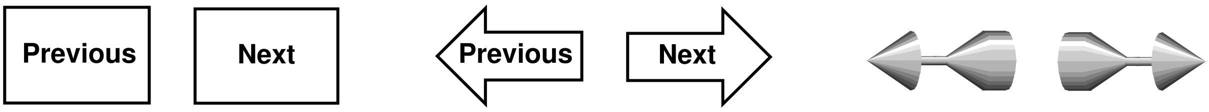


Figure 1: a) An almost round object is perceived as a circle; b) Objects close to each other appear to belong together.



(a) (b) (c)

Figure 2: a) If the shape of a button is too simple the user associates no special information with it; b) a good shape expresses the functionality of the button well; c) if the shape is too complex, the user might forget its meaning.

(a) (b)

Figure 3: a) Objects forming closed shapes belong together; b) Similar objects belong together.

- **Law of Proximity**
  Another effect can be seen in Figure 1(b): objects that are closer to each other seem to form a group. This rule is important to screen design; it shows us an easy way to indicate that certain pieces of information belong together.

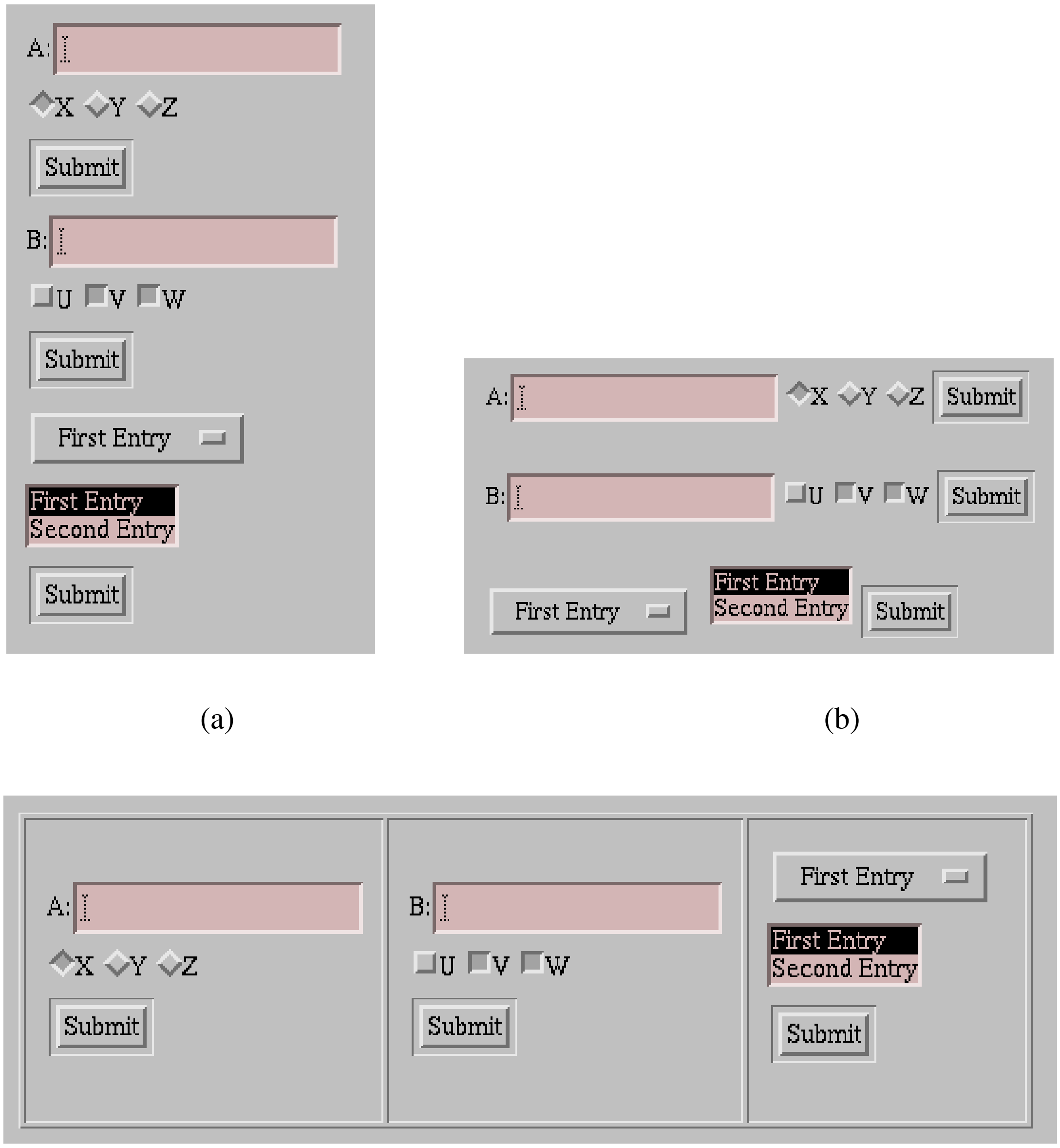
- **Law of Unity**
  Objects that form closed shapes are also perceived as a group. This effect can overrule the Law of Proximity from Figure 1(b), as seen in Figure 3(a).

  The boxes and frames used frequently in graphical user interfaces, e.g., to combine a row of buttons into a more complex user interface component with certain semantics, make use of this law (see Figure 4).

- **Law of Equality**
  Similar objects are another candidate for grouping by our perceptional system. This is the reason why, for example, push buttons arranged in a row should all be the same size – a rule which is still frequently violated in existing documents or interfaces.

- **Law of Continuity**
  Furthermore, there is a tendency of perception to assume continuity in objects: We do not consider Figure 5(a) to consist of a double wedge, but of two intersecting straight lines.

  In Figure 6(b), text indentation forms several groups that are considered to belong together because we perceive a continuity in the levels of indenting.



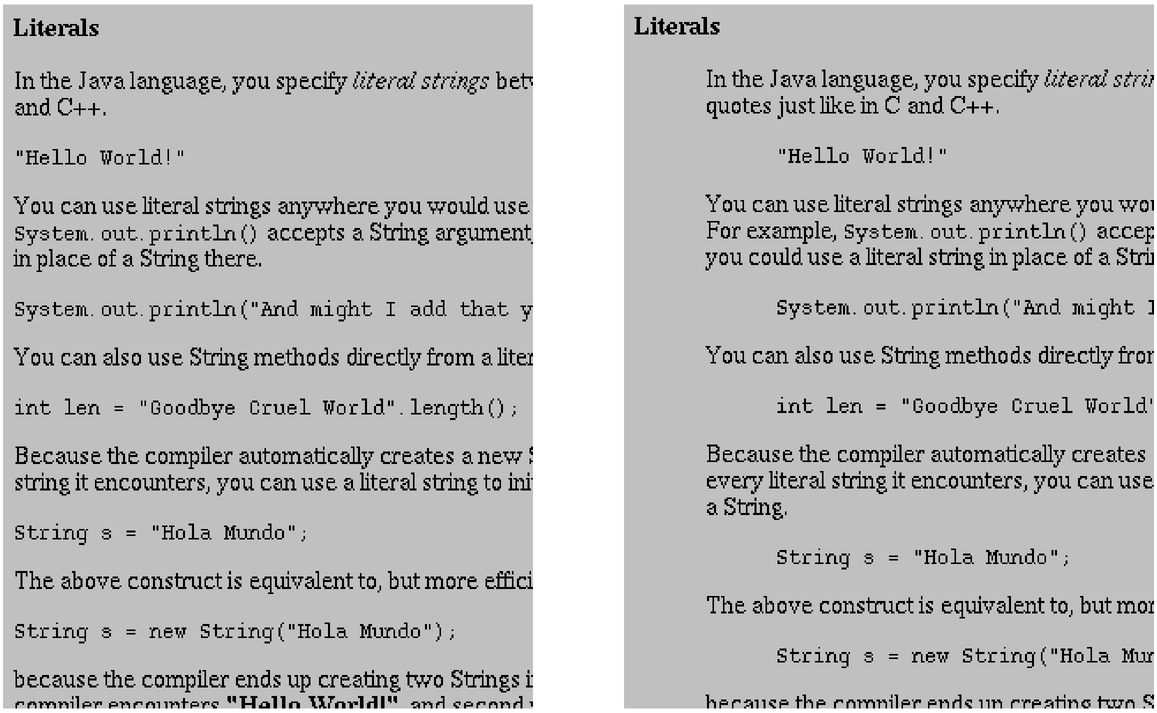
(a) (b)

(c)

Figure 4: Examples for the law of unity: a) unstructured boxes lead to uncertainty about the functionality, e.g., about the semantics of the third “submit” button; b) rearranging in lines gives a clearer layout already; c) finally, using frames to group objects leads to obvious semantics.

(a) (b)

Figure 5: a) Intersecting lines, or double wedge? In case of ambiguity objects are perceived to form the simplest shapes possible. b) We try to map objects to things we know.



Literals

In the Java language, you specify *literal strings* betw
and C++.

"Hello World!"

You can use literal strings anywhere you would use
System.out.println() accepts a String argument,
in place of a String there.

System.out.println("And might I add that y

You can also use String methods directly from a liter

int len = "Goodbye Cruel World".length();

Because the compiler automatically creates a new S
string it encounters, you can use a literal string to ini

String s = "Hola Mundo";

The above construct is equivalent to, but more effici

String s = new String("Hola Mundo");

because the compiler ends up creating two Strings i

Literals

In the Java language, you specify *literal strin*
quotes just like in C and C++.

"Hello World!"

You can use literal strings anywhere you wo
For example, System.out.println() accep
you could use a literal string in place of a Stri

System.out.println("And might I

You can also use String methods directly fron

int len = "Goodbye Cruel World'

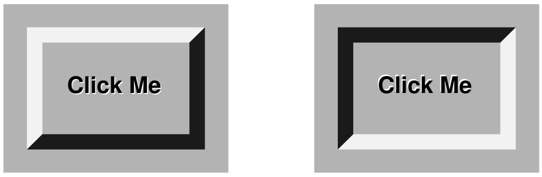
Because the compiler automatically creates
every literal string it encounters, you can use
a String.

String s = "Hola Mundo";

The above construct is equivalent to, but mor

String s = new String("Hola Mur

Figure 6: a) No indentation leads to a text sequence badly structured and hard to read; b) indenting text forms several groups that are perceived as belonging together.



(a) (b)

Figure 7: a) A button with appropriate shadow. b) A button with inverse shadow is percieved as being “pushed in” (lying behind the drawing plane) because our experience suggests that light comes from above.

- **Law of Experience**
  We also try to match objects perceived to things we already know (Figure 5(b)). This is why user interfaces that rely on well-known real-world metaphors are more successful: they minimize memory load. Most people will not map this figure to the letter “E” because their experience does not expect it in this position.

  Another example are “emoticons” frequently used in on-line conversation, such as :-) . The first time, most of us probably had to be told that they have to be viewed sideways, but once we made this experience we interpret subsequently encountered objects correctly.

  Figure 7 shows another application of this law. The left button is perceived as a real button because our experience interprets its shadow as belonging to an object standing out from the drawing plane. The right button is considered to be an object behind the drawing plane, or being “pushed”. This is due to our natural experience that light usually comes from above, which results in many subtle hints we take from shadows to understand (pseudo) 3-D scenes. This effect becomes even more obvious when looking at virtual reality environments (see section 7).

- **Law of common motion**
  If objetcs are moved in front of a static background, we tend to group them together. This is especially the case if they have sufficiently correlated velocity vectors.

Going into Gestalt psychology in its whole depth is beyond the scope of this article, but it should be clear that these laws are fundamental for the way people see and understand their surroundings, including user interfaces and documents.

From an information processing view, an “ideal” document or interface could then be defined as one whose image in the user’s long-term memory does not change anymore because the interface already obeys the Gestalt laws: the perception system does not misinterpret it when applying its rules.

In the next sections we will show how gestalt laws are applied to conventional typography, and how they can be used to design good Web documents.

3 Layout Rules for Conventional Documents

To find design guidelines for Web documents, we start with those rules that traditional typography offers us. We will briefly present the basic schedule that every typographer follows when creating a new printed document, giving some examples of rules for the different steps, and we will identify how those steps have to be adapted to cater for the needs and possibilities of their electronic counterparts.

This change from printed documents to on-line systems will be divided into several steps: First, we will identify some special characteristics of interactive hypertext documents. After that, we will try to classify Web documents, and give separate design rules for each type.

The basic design schedule in conventional document layout comprises the following steps (see also [Gulbins93]):

1. *Page format:* Width and height of the document pages have to be determined. The most widely used formats (width:height) range from the DIN A series $1 : \sqrt{2}$ to $1 : \sqrt{5}$ as a very slim format.

2. *Page layout:* In this step, the area of the page that is to be "used" is defined, i.e., its margins are specified. Classical ratios are 2:3:4:5 for inner:outer:head:foot margins, with the margins taking up $\frac{1}{3}$ of the page width for a spacious layout.

3. *Page pattern:* The arrangement of information inside the page layout framework has to be defined next: how many columns will be used? This directly influences line length which should be in the range from 45 to 65 characters per line.

   Since the human eye moves over text in chunks of several words, shorter lines would mean that the reader could grasp less text per "read cycle" than possible, and that more line changes would be necessary, slowing down reading speed. Justification should be avoided with very short lines; ragged right formatting is more readable in those cases.

   Extremely long lines, on the other hand, require too many read cycles per line, which tires the eye, and makes it harder for the reader to locate the beginning of the following line, again impairing reading speed.

   For vertical spacing, professional document layouts actually use a very rigid grid. The vertical line spacing of the body text is the grid distance, and the bottom line of all elements, even headings and pictures, has to lie on this grid.

4. *Fonts and Styles:* The fonts and sizes to use for body text, headings, tables, etc. have to be determined and used consistently throughout the whole document. Classical values are: 24pt bold for chapter headings, 18pt bold for section headings, 14pt bold for subsection headings, 12pt bold for paragraph headings, 10pt for body text, and 8pt for annotations and footnotes.

   We will not repeat all the typographical rules here. There are some comprehensive books (see [Parker95]) that not only teach the design rules, but also tell the reader how to break them when appropriate. In any case, a maximum of two different font families should be used, and they should be compatible to each other concerning line thickness, capital letter height, etc. Emphasizing elements (italics, bold face, small capitals) should never be used in combination on the same word.

   Choice of a font class can have a strong influence on readability: As soon as small print or inverse coloring (white on black) are used, fonts without serifs (like Helvetica) are to be preferred. The reasons for this lie in the simple shape and the equal width of the character contours. For character-wise indentation, of course, a non-proportional font is necessary. On the other hand, most people claim that proportional serif fonts (like Times-Roman) can be read fastest, which is why they are used in the body text of most printed media (books, newspapers, etc.)

   The overall task, when choosing font sizes and styles, is to create a page appearance with an even "grey value", without particularly dark or light areas.

5. *Contents:* Only after finishing this layout, the actual text is entered into the document frame, and corrected from the contents point of view.

6. *Aesthetic correction:* Professional documents now go through a final "fine tuning" stage. Here, minor changes to kerning etc. are made. For example, a right margin does not look properly justified when adjusted mathematically correctly: lines that end with, e.g., a colon have to be extended slightly to the right to appear "in line" with the rest of the margin.

4 Layout Rules for Web Documents

Web documents are electronically accessible documents with hypertext characteristics. They consist of so-called "pages" of text and graphics that contain cross-references which can be followed interactively. The sum of all cross references can be seen as a structure which is often much more complex than the linear page-after-page structure of a book (see Figure 8). In this paper we will also deal with emerging representation techniques using animations and 3-D sceneries, while we will not consider sound and video as layout components.

Most Web documents are written in HTML, and animation effects are usually implemented using Java applet code for temporal changes. The major problem of HTML from a layout point of view is that it defines a *logical* layout rather than an exact one. This means that certain display attributes or features are supposed to be specified logically, not by physical descriptions. The environment: <em> Text <em>, for example, is only a hint that 'Text' should be emphasized when displayed. The browser decides, however, whether this is done by using blinking, boldface or italic print.

4.1 Web Document Types

Several attributes of Web documents are quite crucial for their layout design. One of those attributes is related to the question whether the document is *passive*, *active*, or even *dynamic*. Some short definitions are useful here:

So far, most pages are *passive*: They merely display information in the form of text, graphics, and hyperlinks. Interaction is limited to following those links.

Active pages contain not only passive, but also interactive GUI (Graphical User Interface) components like forms and buttons that make more complex user input possible. The processing of those inputs, however, still takes place on the server side, usually resulting in a new static page being generated and sent to the client "on the fly".

Dynamic pages finally change over time "on their own", for example by displaying an animation, or modifying their contents and/or layout.

As already indicated, these three types do not necessarily exclude each other. On the contrary: From a designer's point of view, dynamic pages should always contain passive parts that establish a certain consistency throughout the entire document.

4.2 Why Do People Create Web Documents?

Looking at the motivations that drive people and companies to create Web pages enables us to another classification which is very helpful for layout decisions. We detect the following reasons for Web document creation:

being present and making contact

Currently, one of the most appealing reasons to create Web documents is that people or institutions can present (usually non-commercial) information about themselves.

These documents usually consist of passive pages with brief and objective information on a limited subject.

providing information services

Information service pages have to be up-to-date, objective, and easy-to-grasp. They should be mostly passive and well-structured.

Their primary goal is to provide information in a usually limited subject field. An example is a railway traffic information service where a user can find a train connection for a chosen destination. Users that visit those services are motivated already because they need some information. However, they will usually give up if they do not succeed in finding it quickly.

In those systems, pages should be laid out clearly, with constant parts and a stringent user guidance, so that visitors do not have to stay in the system longer than necessary. Document design should try to *minimize* average session duration. Text elements may be dominating the appearance of pages if necessary to provide the information requested. Since on-line documents do not exist in a closed environment such as a printed and bound timetable, the age of the information has to be indicated on each document.

offering transactions and retrieval

Transaction and retrieval systems are designed to be active, unambiguous, brief, and complete.

These systems are similar to information documents, with additional emphasis on information flow from the user to the service. Examples are hotel reservation systems, where the user may have to enter certain data (like name and address) to book a room, and this information is passed on to the hotel computer. The user interface often contains complex virtual input devices.

User motivation is comparable to information services, although success expectations will usually be even higher as the customer has a clear task in mind ("book hotel room") which he hopes to complete using the system. Design should again try to *minimize* session time.

advertising products and services

These documents are dynamic, rarely require much activity, and are quite subjective.

Advertising pages are installed by companies or institutions to present themselves or their products to the on-line public in an attractive and innovative way. The missing initial motivation of potential users has to be compensated for by a visually attracting design, especially of the entry document (or “home page”). Furthermore, the longer the user stays in the system, the higher its success will be in the eyes of the provider.

For that reason, it makes sense for subsequent pages to present new visual stimuli, e.g., by changing to a different overall layout and appearance. The contents should be interesting, entertaining, and keep up suspense so that the user is motivated to explore it further. User guidance should be less stringent. In short, design should try to *maximize* average session duration. Graphical elements should dominate the layout, with only as much text as necessary (people prefer “pushing” a nicely drawn start button to just clicking on a highlighted word saying “Start”.)

to entertain and to have fun

Web documents for entertainment are expected to be colorful, dynamic and in general terms attractive.

Layout has to follow the same rules as for advertising systems or those encountered in the area of video games. Initial motivation is curiosity which the system has to create first. Session duration should be maximized (within reasonable limits). The interface has to be visually appealing *and* simple at the same time, which often leads to design conflicts.

Of course, most documents will belong to two or more of the above classes. Many documents that supply information or services also have the goal to advertise the provider. There will, however, usually be a primary task which should be identified, if not for the whole document, then at least for its individual pages.

4.3 Applying Layout Rules to Web Documents

The following modifications are necessary when applying conventional layout steps to Web documents:

1. *Page format:* Most screens have landscape rather than portrait format, which is not really suitable to display information in pages. Most browsers therefore suggest portrait windows to display pages.

2. *Page layout:* To come close to traditional media sizes even with landscape monitors, there is a choice between leaving side parts of the screen unused, or using several columns/several windows in the layout. Both alternatives are not entirely satisfactory, however.

   Using standard screens as-is with a single-column layout results in text lines that are much too long to be readable.

The extra space can be used for extra windows of other applications or for other Web browser windows, e.g. multiple, partly overlapping pages. What might look appealing at first sight is quite dangerous. People get lost easily when fighting with multi-layer windows. Therefore window overlapping should be used very restrictively and be limited to at most two levels (for more hints see [Klingert96]). Since this is at least partly in the hands of the browser, it is hard to influence through design parameters.

3. *Page pattern:* Hypertexts can be accessed in more ways than ordinary documents. Because of this, a consistent page pattern with an informative heading showing where in the document the user is currently located is essential to avoid the dreaded "lost in hyperspace" syndrome.

   There should be a consistent and easily understandable arrangement of the author's name and address, as well as creation and modification dates. Means to leave the pages by hyperlinks (for example, using a jump-list with Last, Next, Up, Down, and Home links) should be placed in sight. Therefore long pages require several such patterns or a visible and simple way to get back to the jump-list. These requirements are results of Fitts' Law (see [MacKenzie92]) in 2D which correlates the pointing time directly to size and distance of the target object. In fact, some browser designers have recognized this need and introduced *frames* which are separate windows that can display constant banners or other information and links that have to remain accessible all the time while moving through a document (an example is *Netscape 2.0*, see http://home.netscape.com/assist/net_sites/frames.html).

4. *Fonts and Styles:* Electronic documents can use color much easier than printed media. Even though color is the most striking way to emphasize text parts, only a few different colors should be used, and they should be applied conservatively. They can replace the use of different font sizes, e.g., for headings, but it must be remembered that, for example, 8 percent of males in Europe and North America have some form of color blindness. Color should be used consistently, and not in pairings like blue-on-red (spectral opposites lead to focus problems of the eye). Also, color displays will always have a much lower resolution than monochrome or gray-level displays, so for some text-intensive applications a black-and-white or grey level layout might be preferrable (not to speak of the fact that, on the World-Wide Web, actual color capabilities of a browser are unpredictable anyway).

   One should be aware, by the way, that people read slower on screen than on paper. The difference is significant especially when low-resolution devices are used: traditional print offers 300–2400 dpi, while computer screens offer

70–110 dpi. Therefore, using non-serif fonts can often be more appropriate, especially for small font sizes.

5. *Contents:* Here probably lies the most important difference to conventional documents. Hypertexts have to be split into small chunks that can be displayed individually, and organized in a graph-like or a tree-like structure. This usually means that conventional documents have to be restructured completely. Unfortunately, many authors instead choose to just provide an "index layer" of links that point to various sections in their still linear document.

   In fact, those authors ignore the fact that the very way of taking up information appears to change with electronic media becoming widely available: Just as oral cultures became print cultures after Gutenberg [Ong82], print cultures are now becoming electronic cultures, where linear reading is no longer the rule, but rather the exception, and media design has to ensure, above all, quick and easy access to information, supporting browsing and searching techniques [McAdams95].

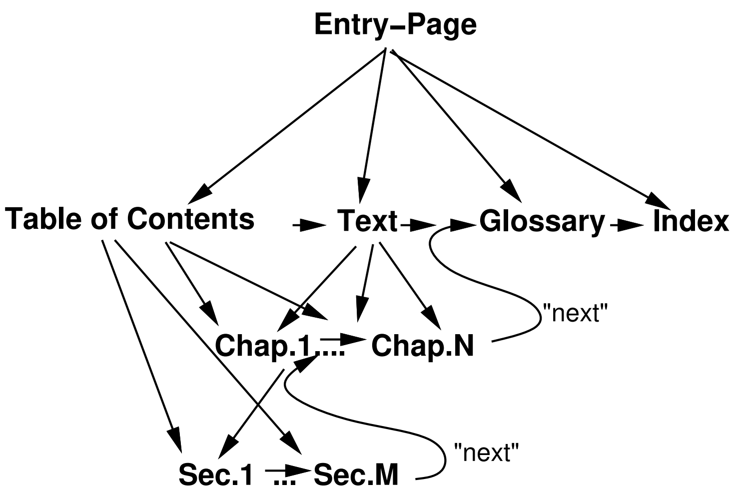


Figure 8: A 'simple' tree-like structure of a book-oriented Web document

   Figure 8 shows how confusing the hyperlinks of a book-oriented document can appear to be.

   The problem is that fast and direct access from the entry-page to a certain section or text page requires many links. Some links cross more than one layer, so that authors finally end up with many overlapping trees, instead of just one. What is supposed to make interaction simple is itself hard to understand: for the user this "forest" is beyond imagination.

Apart from structural considerations, the world-wide availability of documents should also influence their contents: they should be available in English and, if applicable, in the local language.

6. *Aesthetic correction:* The significantly lower resolution of today's computer screens in comparison to printed media means that the fine correction described earlier is hardly possible for documents that are viewed on a screen.

   The multiple-browser problem leaves many fine-tuning parameters to the unknown browser. Viewing Web Documents written for a graphical sound browser with a less equipped program reveals whether the layout was chosen carefully.

In addition to this, active Web documents are read interactively using a computer, which means that all the rules from the area of human-computer interaction and design of graphical user interfaces have to be kept in mind (see [Shneiderman92] for a comprehensive treatment of user interface design).

5 HTML and the Design of Standard Web Documents

HTML, the Hypertext Markup Language, was initially designed to facilitate cooperative work on technical documents [Berners94], and has since become the standard document description language of the World-Wide Web.

HTML is based on SGML, the Standard Graphics Markup Language [Goldfarb94], and is not really designed to specify layout, but a logical document markup (structuring) language. As mentioned above, transforming those structures into a layout is the task of the browser. For example, the exact positioning of pictures on a page cannot be specified in HTML unless the browser is known in advance, and even then arbitrary author-defined positions of pictures are impossible.

But apart from this general problem, HTML in its version 2.0 had several other major drawbacks:

- tables were not part of the standard,
- colors for arbitrary elements could be specified,
- underlaying pages with background images were not possible,
- page layout (two-column text, etc.) could be defined.

HTML development has continued, however, and after the intermediary definition of HTML+, the specification of HTML 3.0 is now in its last stages [Raggett95]. It offers the following improvements:

- tables and justification, and therefore page layout (two-column text, etc.),
- background images,
- inline images with many different, arbitrarily shaped sensitive areas,
- concrete layout specifications via cascading style sheets.

Special layout wishes can be implemented using HTML itself, or custom style sheets. At the same time, HTML itself remains largely a logical markup language, since style sheets are not part of the language itself but special instructions that can be obeyed by certain browsers, but ignored by others.

It has to be mentioned, however, that the specification of HTML 3.0 is not complete yet, and some of the above features might still get improved further.

6 Java and the Design of Interactive Documents

Java [Gosling95], originally designed by Sun Microsystems Inc. for the development of consumer electronics software, is an object-oriented programming language very similar to C++. What makes it interesting is its runtime conception: The Java compiler translates Java source code into a machine- and platform-independent intermediate code for a virtual machine. For most of the common architectures and platforms a sufficiently fast interpreter of this intermediate code format exists. The independence from architecture and platform makes it possible to offer uniform software and functionality via the World-Wide Web.

Another interesting feature of the conception is that the interpreter determines which code modules are required for the application, and which modules have been loaded already. Only missing code modules will be downloaded via the Web. Each interpreter initially has several code modules for the output of text, sound and graphics that are required frequently, so these modules do not have to be downloaded over the network at runtime anymore.

Some Web browsers are able to interpret Java program look-alikes (so-called Java *applets*) that are embedded into HTML documents. This way objects can be integrated into Web documents which directly interact with the user, or which are continuously changing themselves – without any need for network communication once the applet has been downloaded and started.

Authors can integrate objects which show animated or even three-dimensional views of data that has been downloaded, and which can be viewed interactively with high-speed feedback through the downloaded applet that executes locally on the client (browser) side. If the applet itself interacts with other resources regularly, an information provider could even integrate a stock market ticker, or a diagram that shows current financial indices, into his service pages.

Currently there are two Web browsers which can handle Java applets:

HotJava from Sun Microsystems Inc. is a Java application itself. Therefore, it can modify and extend itself at runtime: it can adapt itself, for example, to a new HTML version simply by downloading the functionality. In the same way, Hotjava may "learn" new communication protocols.

Netscape 2.0 from Netscape Communications Inc. is not a Java application; it is distributed in a regular binary format. However, Netscape can interpret Java applets in HTML documents. The availability of Java functionality in a widely used browser as Netscape might be an indictation of the coming relevance of Java as a programming and authoring language.

The specification of Java is not finished yet. Although there are a number of Web documents with integrated Java applets already, it is not widely used in the World-Wide Web at this moment. The language appears, however, to be quite stable by now. For details on Java, HotJava, and sample applets, Sun's Java Web page at http://java.sun.com/ is the best place to start.

6.1 Layout for Dynamic Pages

Using Java, it is possible to create whole documents as Java applets, making it possible to work around the restrictions of HTML. On the other hand, dynamic pages create additional challenges concerning the clear and appealing layout: In a way, another dimension, time, has to be taken into account.

The rule of consistency, for example, has to be applied to the temporal changes of the document: The document should keep a consistent framework even though parts of it may now change constantly. The "lost in hyperspace" syndrome also gets another dimension: Not only must it be possible to get back to the previous or entry document at any time, running applets also have to provide for a clearly accessable "cancel point" where their execution can be interrupted in a controlled way, bringing the user back the state before he started the applet.

With complex time-based documents, the design gets much more complicated, and rules will have to be derived from fields like movie creation etc. In practice, however, most dynamic pages limit their animations to clearly restricted areas. This means that the layout is largely determined by static objects, which makes the rules described beforehand applicable.

7 VRML

With the abundant amount of information available through the existing World-Wide Web and its dynamic extensions, and completely new and useful applications and information servers coming up every day, it might be questionable why there should be a need for a 3-D environment to present information, entertainment, or any of the other contents classified earlier in this paper.

There are, however, a number of applications for which presentation using 3-D graphics can increase their attractiveness considerably. These applications can be classified into the following groups:

Data Visualization: Researchers in the natural, technical, and social sciences, as well as many branches of industry (e.g., fincancial businesses) often have to handle large amounts of statistical or other data that can be visualized more clearly through an abstract 3-D model. The translation into a visible structure can help people to "see" certain regularities or trends.

Object Visualization: Some research areas deal with 3-D structures that can be visualized directly (e.g, in molecular biology).

Information Visualization: While data has no semantics attached to it yet, information is "data with a meaning". Some structural information is presented more naturally as 3-D graphs (take, for example, the organizational pyramid of people working for a software company, with hierarchy going top-down, divisions like engineering and marketing going from left to right, and different projects displayed from front to back).

Reality Visualization: Finally, we can try to visualize "real" everyday objects and environments. 3-D worlds can be a more natural environment for social interaction via networks (telepresence) than text-only systems like chat groups, etc.

7.1 Conceptual Problems With 3-D Interfaces

The most prominent problem with 3-D documents (or "worlds", as we will call them in this section) is navigation: While 2-D input is supported nicely through technologies like the mouse, 3-D navigation can only be accomplished in one of the two following ways:

- using specialized *3-D devices* like the Spaceball, the Spacemouse, or 3-D hand-gesture input [Bröckl-Fox95], with the problem that those devices are not generally available and often unsuitable for other, 2-D oriented tasks;

- or by using *3-D metaphors* that interpret the input from standard 2-D devices as navigational commands in the application's 3-D space [Kettner93]. The disadvantage of this approach is that navigation is often less intuitive because the missing third input dimension has to be generated in some way (this is the main task of a 3-D metaphor).

Moreover, 3-D worlds have to define the ways in which the user is allowed to move through them. Here the choice is between two basic models:

Simulative: If virtual worlds model real worlds, then it makes sense to constrain the possible movement of the user to those degrees of freedom he knows from real life: He may walk along the open 2-D ground, but he is not allowed to fly up into the air or walk through walls.

Innovative: If, however, the system does not try to simulate real environments, then it makes sense to take advantage of the additional degrees of freedom that distinguish virtual realities from physical reality, like the absence of gravity, or weather, and to pass those possibilities on to the user through a more liberal navigation model.

In either case, we have far more possible *views* that the user can take onto our objects than with a 2-D document. It is quite a challenge to create a 3-D environment in which all possible views are also aesthetically pleasing, informative, and navigable! In fact, most existing worlds contain ugly "backyards" or "holes" where the three-dimensional illusion collapses if it was trying to keep up an image of a "real" world. This problem is made worse through the fact that 3-D scenes usually have to be mapped to a 2-D output device (computer screen) with limited resolution: In contrast to traditional Web documents, objects can now be too far away, or viewed under an angle too large to be rendered readable.

From user reactions, another problem with 3-D worlds has been identified, especially when used as social spaces: They are often rejected because they "feel unnatural" — even though, at first sight, they seem to offer a much improved "realism" over text-only interfaces! The reasons for this are twofold: First, virtual reality environments usually create very high expectations towards their realism in appearance which they cannot meet with today's technology. The second reason is less obvious: Like all media, VR environments can be classified according to their *vividness*, or degree of realism in appearance, and *interactivity*, or degree of realism in behaviour (see [Steuer94]). Steuer's research showed that realistic behaviour is more important than realistic appearance for the perceived socialness. For example, textual MUDs with complex interaction models are often perceived more "realistic" than 3-D worlds with only simple means of interaction.

7.2 Some Existing Systems And Standards

In spite of these problems, a number of 3-D environments and standards have been created: Apart from numerous, mostly academic, pure scene description languages, *Worlds Chat* and *AlphaWorld* by Worlds Inc. (http://www.worlds.net/) are quite successful, but proprietary commercial projects for social navigable 3-D worlds.

The *Information Visualizer* [Card91] is a system that focuses on the presentation of information structures using a sophisticated set of three-dimensional layout prinicples. The underlying project aims at developing a user interface paradigm going beyond the desktop metaphor, making use of the emerging technological possibilities to simplify finding, manipulating, and understanding large amounts of information.

VRML is the open standard for the description of 3-D object worlds that has been developed for incorporation into the existing World-Wide Web. The VRML specification [Bell95] is basically nothing more than a definition of an ASCII file format for the description of three-dimensional scenes. It is actually largely derived from, and identical to, the *Open Inventor* file format defined by Silicon Graphics Inc. for the description of 3-D scenes in their Open Inventor System.

In VRML, Scenes are defined as *scene graphs*, a hierarchy of *nodes* that represent either objects (*shape nodes*), attributes like material, transformation, light, or camera (*property nodes*), or hierarchies (*group nodes*).

Shape nodes include the usual basic set of objects (spheres, cones, cubes, line sets, text, etc.). An interesting feature for our purposes are Level-Of-Detail nodes who allow changing (detailing) the representation of an object when the ecamera comes close to it, and the possibility to include links to other VRML scenes or external Web documents into the graph.

Apart from that, VRML does not support interactive behaviour yet; however, this feature is scheduled for inclusion into the VRML 2.0 specification, and is also addressed by extensions like *VRML+* from Worlds Inc.

Since VRML is the most promising approach to the definition of a standard for 3-D environments on the World-Wide Web, we will focus on this technology in the rest of this section.

7.3 Layout Rules for 3-D Worlds

As we have seen in previous sections, designers of electronic text documents can derive many heuristics or even exact rules from their historical “ancestors”, traditional typography experts. (Even though it has to be stressed again here that on-line text has nothing in common with printed text except the use of textual symbols, and the new limitations and possibilities of electronic documents have to be taken into account by modifying traditional rules.)

Designers of simulative 3-D worlds have much less historical background to borrow from: Although Architecture as a scientific discipline should serve nicely as supplier of theories, rules, and guidelines for the design of “user-friendly spaces”, reality is quite different: Architectural knowledge is mostly passed on in the form of good examples, guidelines tend to be rather contemporary and vary enormously over time, and different schools often promote contrasting guidelines (if any).

Moreover, it is extremely tedious to create object simulations with a high degree of detail since all the real-world characteristics (little irregularities, stains and scratches) would have to be created by hand or with very complex software.

In this case, it is probably a better idea not to try to simulate real-world objects (why should houses in an electronic community have saddle roofs when there is no rain?), but to create genuinely virtual, innovative spaces. They can take advantage of the newly gained freedom, and use new, artificial, but appropriate representations for objects in them.

To make these virtual worlds interesting, a high level of detail can then be created not by copying real-world complexity, but by creating appropriate complexity automatically: Instead of putting a texture of real-world marble onto a surface, for example, it can be “painted” using a fractal-generating algorithm that creates aesthetic and attractive textures automatically based on mathematical systems (real-time constraints would mean precomputation of such patterns, of course).

It has to be kept in mind that 3-D environments usually re-imlement the spatial distance between different information sources which had disappeared through the hypertext link in standard Web documents. Designers should make sure that “walking” from one point to another in a virtual world is not a boring, unnecessary delay (*transport*), but that on this way users can experience interesting events, meet other people, or come across something new, making the journey itself rewarding (*travel*) [Möller94].

If, however, the 3-D environment is mainly aimed at presenting structural information, then a number of rules become applicable that are often derived from the 2-D case.

One example is *Fitt’s Law* that has been described in an earlier section; it can basically be extended to the 3-D case, although experimental results still have to be gathered. Arno?

Similarly, most Gestalt Laws are easily transferred to the 3-D case: The Law of Proximity, for example, applies equally to 3-D arrangements of objects. When carrying over these laws, it has to be kept in mind, however, that the actual view onto the layout can have many different perspectives, and that perspective distortion may disturb the perception of otherwise perfect layouts.

Furthermore, 3-D worlds are often created as “wrappers” around 2-D objects:

Many environments enable the visitor to move through them in order to reach points of interest that are actually standard Web documents. Of course, in this case all the rules for the design of conventional pages apply for those embedded documents.

This situation brings about the additional challenge of creating a seamless transition between the 3-D framework and its 2-D contents. This can be achieved by displaying the document inside a three-dimensional frame that mimicks, e.g., a "wall-mounted monitor" or other reading device which appears as part of the 3-D environment. Similar techniques can often be encountered in video games.

Finally, the navigation engine of a 3-D environment viewer usually supports different modes of movement, or *metaphors*: The *VRweb* viewer [Pichler95], for example, supports five different navigational modes:

Flip where the object or scene can be moved while the viewer remains stationary;

Walk with forwards/backwards motion plus up/down "climbing", sidestepping and head (camera) turning;

Fly with an aircraft-piloting-like set of controls for climbing/diving and accelerating/decelerating;

Fly To which automatically moves the user to a selected Point Of Interest (POI) [Mackinlay90];

Heads Up which makes most of those modes accessible by overlaying the scene display with semi-transparent control buttons.

This sample list shows that scene designers, especially when they design scenes for a standard like VRML where multiple clients exist, cannot rely on a certain type of navigation to be available – or on certain types of movements (like free flight) *not* to be available! This is just the same situation as for Web document designers who cannot predict on what sort of browser their pages will be displayed. For those reasons, 3-D models have to take into account that users can potentially move around arbitrarily in their scene.

With authoring tools support still largely missing, it becomes clear from these observations that designing 3-D worlds is still a much more complex task than creating other Web documents.

8 Summary

This paper has shown that layout questions are an important issue in the design of on-line Web documents. Guidelines for good layout can be drawn from conventional typography, Gestalt psychology, architecture, film-making, human-computer interaction, and many other areas. For serious projects, however, it should have become clear that the above areas are complex enough to justify consulting experts in the respective field.

To design an optimal layout requires classification of the documents to be created according to their intention. Moreover, on-line documents can be created using a number of different media, like static or dynamic graphics, and 2-D or 3-D presentation techniques. Those new media require adaptation of many rules to their new characteristics, like hypertext structure, navigational features, etc.

While HTML 3.0 is suitable only for text and graphics documents that do not require animation or interaction, Java applets offer this functionality. 3-D Web environments are still in their infancy; many user interface issues have not been solved yet. The most widespread standard is VRML which in its current version, however, does not offer interaction.

In all, the design of aesthetically pleasing Web documents is easily possible, and supported by tools but only in the case of passive 2-D text/graphics pages. More elaborate presentation forms still need authorware that will support content providers in their task to create interesting and appealing documents.

Parts of the ideas presented in this article, with special emphasis on kiosk systems, were published in [Borchers95].

References

[Bell95] Bell, Gavin; Parisi, Anthony; Pesce, Mark: *The Virtual Reality Modeling Language – Version 1.0 Specification (clarified)*, November 1995 (http://www.oki.com/vrml/vrml10c.html)

[Berners94] Berners-Lee, Tim: *The World-Wide Web*, in: CACM, Vol. 37, No. 8, August 1994

[Borchers95] Borchers, Jan; Deussen, Oliver; Knörzer Clemens: *Getting it Across: Layout Issues for Kiosk Systems*, SIGCHI Bulletin, Vol. 27, No. 4, October 1995

[Bröckl-Fox95] Bröckl-Fox, Ulrich: *Untersuchung neuer, gestenbasierter Methoden für die 3D-Interaktion*, dissertation, University of Karlsruhe, Germany 1995

[Card91] Card, Stuart K.; Robertson, George G.; Mackinlay, Jock D: *The Information Visualizer – An Information Workspace*, Proceedings of ACM CHI'91 Conference on Human Factors in Computing Systems, Information Visualization, 1991, pp. 181-188

[Goldfarb94] Goldfarb, Charles F.: *The SGML Handbook*, Oxford University Press, Oxford 1994

[Gosling95] Gosling, James; McGilton, Henry: *The Java Language Environment – A White Paper*, Sun Microsystems, Inc., Mountain View,CA 1995

[Gulbins93] Gulbins, Jürgen: *Mut zur Typographie*, Springer-Verlag, Berlin 1992

[Kay95] Kay, Alan: *When Can We Take The Training Wheels Off?*, keynote speech, Third International World-Wide Web Conference, Darmstadt 1995

[Kettner93] Kettner, Lutz: *Mathematisch-Informationstheoretische Untersuchung von 3D-Metaphern*, final thesis (Diplomarbeit), University of Karlsruhe, December 1993

[Klingert96] Klingert, Arnold,: *Einführung in Graphische Fenstersysteme*, Springer-Verlag, Heidelberg 1996

[Koehler29] Köhler, Wolfgang: *Gestalt Psychology*, Liversight, New York 1929

[MacKenzie92] MacKenzie, I.S., Buxton, W., *Extending Fitts' Law to Two-Dimensional Tasks*, in: Proceedings of the HCI, 1992.

[Mackinlay90] Mackinlay, Jock D.; Card, Stuart K.; Robertson, George G.: *Rapid Controlled Movement Through a Virtual 3D Workspace*, in: Proc. ACM SIGGRAPH '90, Dallas, Texas, New York August 1990, pp. 171–176

[McAdams95] McAdams, Melinda: *Information Design and the New Media*, interactions, vol. 11, no. 4, ACM Press, October 1995, pp. 36–46

[McLuhan67] McLuhan, Marshall: *The Medium Is the Massage* (sic), Random House 1967 (reprint: Simon and Schuster 1989)

[Möller94] Möller, Christian: *Die Inszenierung des Zwischenraumes*, in: Nonlocated online, Medien.Kunst.Passagen, Passagen Verlag, Wien 1994

[Ong82] Ong, W.J.: *Orality and Literacy: The Technologizing of the World*, Routledge, London 1982

[Parker95] Parker, Roger C.: *Looking Good in Print*, 2nd germ. Edition, Midas-Verlag, Switzerland 1995

[Pichler95] Pichler, Michael; Orasche, Gerbert; Andrews, Keith: *VRweb: A Multi-System VRML Viewer*, Proc. VRML '95, San Diego, CA, December 1995

[Raggett95] Raggett, Dave: *HyperText Markup Language Specification, Version 3.0*, Internet Draft, March 1995 (http://www.w3.org/pub/WWW/MarkUp/html3/CoverPage.html)

[Sassoon94] Sassoon, Rosemary (ed.): *Computers and Typography*, Intellect, Oxford 1993

[Schmitt94] Schmitt, Alfred: *Mensch-Maschine-Dialog I*, lecture notes, University of Karlsruhe, 1994

[Shneiderman92] Shneiderman, Ben: *Designing the User Interface*, Second edition, Addison-Wesley, Reading,MA, 1992

[Steuer94] Steuer, Jonathan: *Vividness and Source of Evaluation as Determinants of Social Responses Toward Mediated Representations of Agency*, dissertation, Stanford University, December 1994 (http://www.cyborganic.com/People/jonathan/Academia/Dissertation/)

About The Authors

Jan Borchers graduated in computer science at the University of Karlsruhe in 1995, and currently works on his doctoral thesis at the Telecooperation Research

Group at the University of Linz. His main interests lie in the design of user interfaces for all types of new media, from electronic books and mobile devices to virtual realities.

Telecooperation Research Group
Institute of Computer Science
University of Linz 4040 Linz, Austria
email: jan@tk.uni-linz.ac.at
http://www.tk.uni-linz.ac.at/~jan

Oliver Deussen graduated in computer science at the University of Karlsruhe in 1991, and currently works on his doctoral thesis at the University's Institute for Operating and Dialog Systems. He is interested in simulation and modelling aspects of computer graphics and the optimized design of graphical user interfaces. He also likes efficient and easy to use algorithms of computational geometry.

Institute for Operating and Dialog Systems
University of Karlsruhe
76128 Karlsruhe, Germany
email: oliver@informatik.uni-karlsruhe.de
http://www.uni-karlsruhe.de/~oliver

Clemens Knörzer received his master's degree in computer science from the University of Karlsruhe in 1992, and currently works at the "Zentrum für Kunst und Medientechnologie" Karlsruhe (ZKM) in a project called "Forschungsverbund Medientechnik Südwest" where he designs a multimedial video database. He is also interested in data compression and multimedial user interfaces.

Institute for Operating and Dialog Systems
University of Karlsruhe
76128 Karlsruhe, Germany
email: knoerzer@informatik.uni-karlsruhe.de
http://www.uni-karlsruhe.de/~knoerzer

Arnold Klingert received his PhD from the University of Karlsruhe in 1993. He currently works there as a senior assistant. Recently he published a book about graphical window systems. As areas of special interest he considers: tools and platforms for user interface design, medical visualization, and mobile computing.

Institute for Operating and Dialog Systems
University of Karlsruhe
76128 Karlsruhe, Germany
email: ajk@informatik.uni-karlsruhe.de
http://www.uni-karlsruhe.de/~ajk