Teil IV

Ausgewählte Interaktionsformen

15 Grafische Benutzerschnittstellen am Personal Computer

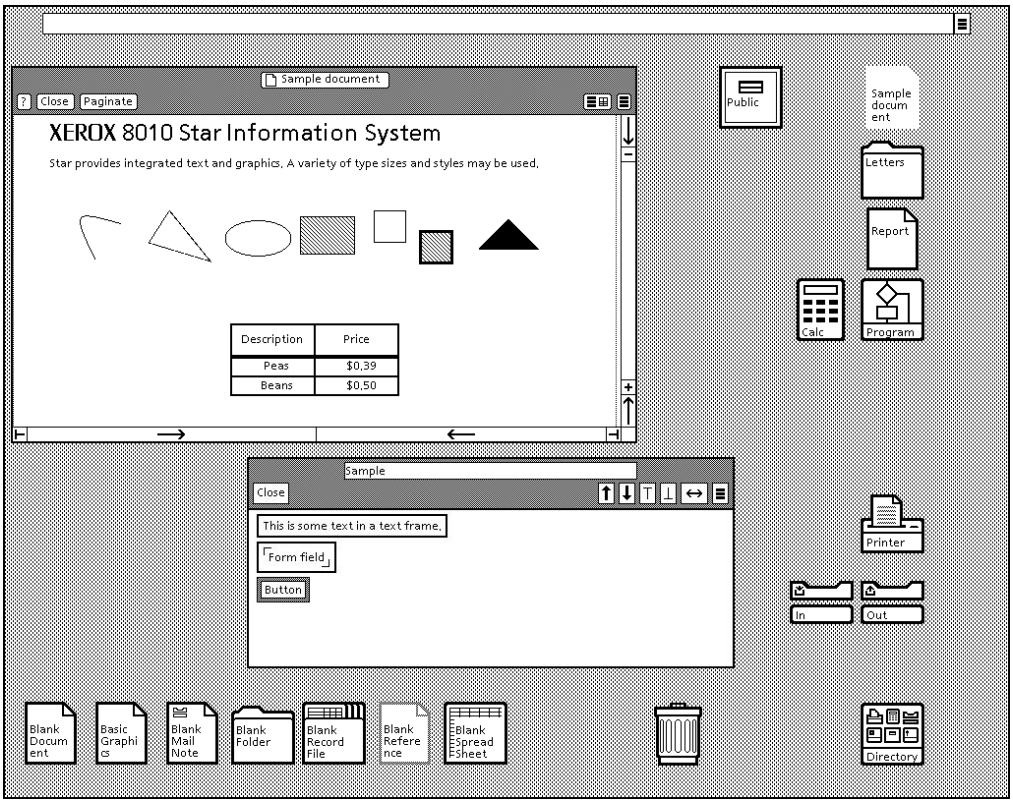
15.1 Personal Computer und Desktop Metapher

Historisch gesehen war der Computer in seiner Anfangszeit (etwa ab den 1940er Jahren) eine große und teure Maschine, die vor allem durch Spezialisten bedient wurde, die sich auch mit Aufbau und Funktion des Computers bestens auskannten. Später, im Zeitalter der **Großrechner** (engl. **Mainframe**) (etwa ab den 1960er Jahren), konnten solche Rechenanlagen dann auch mehrere Benutzer gleichzeitig versorgen. Jeder Benutzer hatte einen Textbildschirm und eine Tastatur zur Verfügung (zusammen ein sogenanntes **Terminal**) und konnte daran Kommandos eingeben, Dateien verwalten und editieren, und auch Programme schreiben und ausführen lassen. Noch heute finden wir diese Kommandozeilen-Umgebungen in viele PC-Betriebssysteme integriert (siehe Abschnitt 8.1 und Abbildung 8.1 auf Seite 86).

Mit der weiteren Entwicklung der Technik wurden die Rechner dann immer kleiner und preiswerter, zunächst entstanden sogenannte **Workstations**, also Arbeitsplätze, die jedem Benutzer seinen eigenen Rechner zur Verfügung stellten, und schließlich (etwa ab den 1980er Jahren) der sogenannte **Personal Computer** oder **PC**, der auch preislich dann für Privatanwender erschwinglich wurde. Seit Beginn dieses Jahrtausends wird der Personal Computer zunehmend durch mobile Rechner (**PDAs** und **Smartphones**) ergänzt. Hinzu kommen etwa seit 2010 auch größere Tablets, die nicht mehr als mobile PCs verstanden werden. Die überwiegende Nutzungsform von Rechenleistung verschiebt sich derzeit weg vom klassischen PC und hin zu diesen meist durch *Touch*-Eingabe bedienbaren mobilen Bauformen (siehe auch Kapitel 17 und 18).

Mit der technologischen Entwicklung der Computertechnik besaßen diese Personal Computer von Anfang an grafikfähige Bildschirme und ein grafisches Eingabegerät, die **Maus**, sowie die aus der Mainframe Zeit verbliebene Tastatur. Sie verarbeiteten Daten in einem flüchtigen *Haupt-* oder *Arbeitsspeicher* und legten sie auf einem nichtflüchtigen Speichermedium wie einer *Festplatte* oder *Diskette* ab. Die Maus war als Eingabegerät relativ gut untersucht und ihr Verhalten mit Gesetzmäßigkeiten wie **Fitts' Law** oder dem **Steering Law** gut zu beschreiben (siehe Abschnitte 4.1 und 4.2).

Konzipiert waren sie zur Erledigung von Büro-Arbeiten, und einer der konzeptuellen Vorläufer des PC, die Star Workstation (siehe Abbildung 15.1 auf Seite 158), wurde auch von einer Firma aus der Büro-Branche, Xerox, entwickelt. Da es darum ging, die Arbeitsabläufe eines damaligen papierbasierten Büros auf den Rechner zu übertragen, lag es auf der Hand, die Objekte und Aktionen dieser Umgebung auf dem Rechner grafisch abzubilden (siehe hierzu auch Abschnitt 7.10). So entstand die **Desktop Metapher**



mmibuch.de/a/15.1

Abbildung 15.1: Das Interface der Xerox Star Workstation (1981) mit den grafischen Elementen für Dokumente, Ordner und Ablagefächer sowie für die Werkzeuge Drucker, Taschenrechner und Papierkorb

mit ihrer Schreibtischoberfläche, den Ordnern, Akten und Dokumenten. Die Elemente dieser Metapher sind uns mittlerweile so vertraut und erscheinen so selbstverständlich, dass sie im alltäglichen Sprachgebrauch oft mit den Begriffen für die technischen Umsetzungen vertauscht werden: Ein *Dokument* ist technisch gesehen als eine *Datei* abgelegt. Ein *Ordner* enthält konzeptuell mehrere *Dokumente*, genau wie ein *Verzeichnis* technisch mehrere *Dateien* enthält. Zum technischen Konzept des *Laufwerks*, also der Festplatte, Diskette oder der heutigen Halbleiterspeicher, existierte kein so richtig passendes Element in der Metapher, höchstens vielleicht der Aktenschrank, daher taucht das *Laufwerk* auch genau als solches in der Desktop-Umgebung auf und bildet damit eine Abweichung von der physikalischen Welt. Die Desktop Metapher ist also ein Beispiel für ein **konzeptuelles Modell**, das sich in manchen Bereichen recht nahe an das **implementierte Modell** anlehnt (vgl. Kapitel 5).

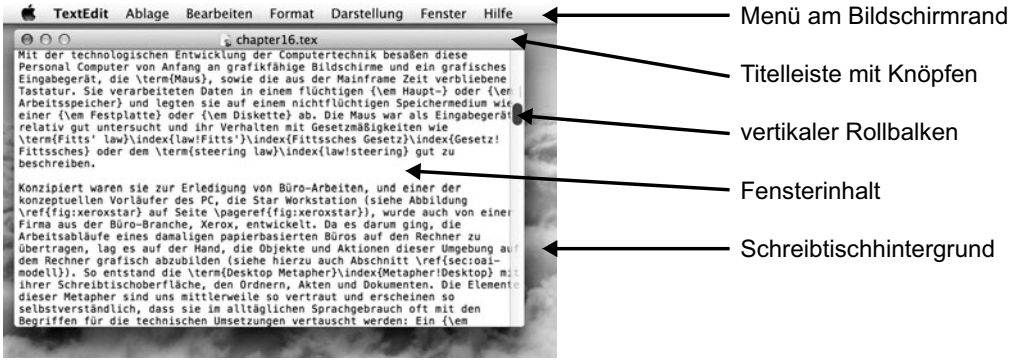


Abbildung 15.2: Grundlegende Elemente eines Fensters in einer heutigen Desktop-Umgebung

mmibuch.de/a/15.2

15.2 Das WIMP Konzept

Die grundlegenden Elemente der grafischen Benutzerschnittstelle am PC sind **Fenster**, **Icons**, **Menüs** und ein **Zeiger**, der den Bewegungen eines **Zeigegeräts** wie z.B. der Maus folgt. Auf Englisch heißen diese Begriffe **Window**, **Icon**, **Menu** und **Pointer**, was zusammen das Akronym **WIMP** ergibt (engl. *wimp* = Feigling, Schwächling). Mit diesen 4 Grundelementen wird das Konzept der **Direkten Manipulation** umgesetzt (vgl. Abschnitt 8.4). Das WIMP Konzept ist jedoch allgemeiner als die Desktop Metapher und kann potenziell auch andere Metaphern umsetzen. Das Prinzip der Direkten Manipulation ist ebenfalls übergeordnet und findet beispielsweise in vielen Computerspielen Anwendung, ohne dass dort von Dokumenten und Ordnern die Rede ist.

15.2.1 Fenster und Leisten

Der Begriff *Fenster* bezeichnet im Zusammenhang mit WIMP einen reservierten Bildschirmbereich, der für eine bestimmte Anwendung reserviert ist. Beispiele hierfür sind komplette Anwendungsfenster, aber auch kleinere Dialogboxen oder Meldungen. Fenster können (müssen aber nicht) beweglich sein. Sie können sich gegenseitig überlappen und verdecken, und ihre Anordnung auf dem Bildschirm ist keine triviale Aufgabe. Jede WIMP Umgebung hat für diesen Zweck eine Programmkomponente, die sich nur um die Verwaltung der Fenster kümmert. Unter **LINUX** gibt es verschiedene, austauschbare **Window Manager**, in Windows und Mac OS ist diese Komponente fester Bestandteil des Betriebssystems. Neben dem reservierten Bildschirmbereich sind Fenster oft mit einer sogenannten **Dekoration** versehen, die es erlaubt, das Fenster zu verschieben, zu skalieren, zu schließen oder zu minimieren. Auch in Betriebssystemen für kleinere Bildschirme, wie **Windows CE** gab es das Konzept von Fenstern noch, was jedoch kaum Sinn ergab, da fast alle Fenster (bis auf kleine Dialogboxen) den vollen Bild-

schirm beanspruchten. Auch in heutigen Smartphone Betriebssystemen existiert das Konzept des Fensters auf Software-Ebene noch, in der grafischen Benutzerschnittstelle fehlen jedoch die klassischen Dekorations-Elemente verschiebbarer Fenster. Benötigt der in einem Fenster dargestellte Inhalt mehr Platz als das Fenster bietet, so zeigt das Fenster üblicherweise nur einen Ausschnitt. Die Position dieses Ausschnitts kann mit den sogenannten **Scroll-Balken** eingestellt werden. Dabei zeigt der Reiter des Balkens die Position und meistens auch die Größe des Ausschnitts im Gesamtdokument.

Neben den Fenstern enthalten aktuelle Desktop Schnittstellen regelmäßig Leisten am Bildschirmrand, wie z.B. die **Menüleiste** in OS X oder die Task-Leiste in Windows. Diese Anordnung am oberen oder unteren Rand des Bildschirms hat den Vorteil, dass die vertikale Position nicht genau getroffen werden muss, sondern die Maus beliebig über den Bildschirmrand hinaus geschoben werden kann und dort hängen bleibt. Aus diesem Grund befinden sich dort häufig benötigte Funktionen, wie z.B. das Anwendungsmenü in der OS X Menüleiste oder die minimierten Fenster in der Windows Task Leiste. Eine weitere bevorzugte Position auf dem Bildschirm sind die Ecken, da dort die Maus gleich in zwei Richtungen hängen bleibt. Aus diesem Grund befinden sich dort die am häufigsten genutzten Funktionen des Betriebssystems, wie das **Startmenü** in Windows und das **Systemmenü** in OS X.

15.2.2 Menü-Techniken

Menüs, die aus einer Leiste am oberen Bildschirmrand herabgezogen werden, heißen aus diesem Grund **Pull-down-Menü** (siehe Abbildung 15.3). Menüs, die an einer beliebigen Bildschirmposition erscheinen, wie das **Kontextmenü** vieler Anwendungen, heißen **Pop-up-Menü**. Beides sind **lineare Menüs**. Dies bedeutet, dass die verschiedenen Einträge des Menüs linear untereinander angeordnet sind. Je weiter entfernt von der Startposition sich ein Eintrag befindet, desto länger dauert es, ihn mit der Maus zu treffen. Dies folgt aus **Fitts' Law** (siehe Abschnitt 4.1). Noch komplizierter wird es, wenn ein Eintrag des Menüs ein Untermenü öffnet. In diesem Fall muss sich der Mauszeiger zunächst zum betreffenden Eintrag bewegen, dann innerhalb des Eintrags bis zur Markierung am rechten Rand, worauf sich das Untermenü öffnet, dann in dieses hinein und hinunter zum Eintrag der ersten Schachtelungsebene (siehe Abbildung 15.3). Dabei ist der Weg nach rechts innerhalb des ersten Menüeintrages oben und unten begrenzt durch die Höhe des Eintrags: verlässt der Mauszeiger auf dem Weg zum rechten Rand den Eintrag, dann öffnet sich das falsche Untermenü. Die für diese Mausbewegung benötigte Zeit lässt sich nach dem **Steering Law** abschätzen (siehe Abschnitt 4.2). Besonders störend werden diese Effekte bei langen Menüs mit vielen und daher verhältnismäßig schmalen und langen Einträgen. Ab einer gewissen Größe werden geschachtelte Menüs schwer bedienbar, wie viele Leser wohl in eigener leidvoller Erfahrung festgestellt haben. Es gibt verschiedene Ansätze, die störenden Effekte geschachtelter Menüs zu entschärfen, beispielsweise indem der Mauszeiger ab einer bestimmten X-Position den Eintrag nicht mehr mehr so leicht verlassen kann.

Neben den linearen Menüs gibt es auch sogenannte **Torten-Menüs**, die ihre Einträge im Kreis anordnen und damit einen Vorteil bezüglich der Auswahlgeschwindigkeit erreichen (Abbildung 15.4) Ein Eintrag eines Tortenmenüs wird ausgewählt, indem der Mauszeiger das Menü in Richtung des Eintrags verlässt. Geschachtelte Tortenmenüs

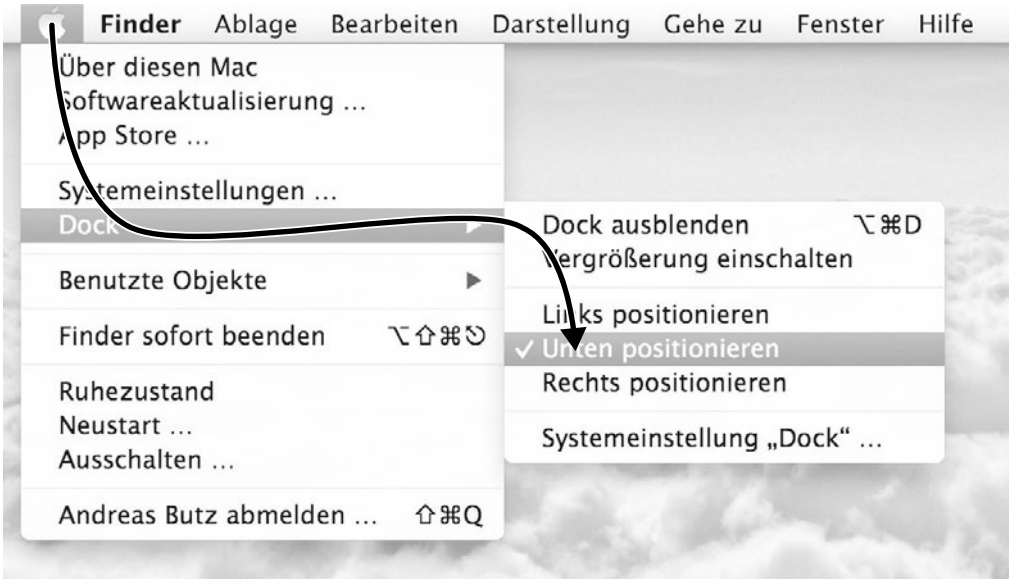
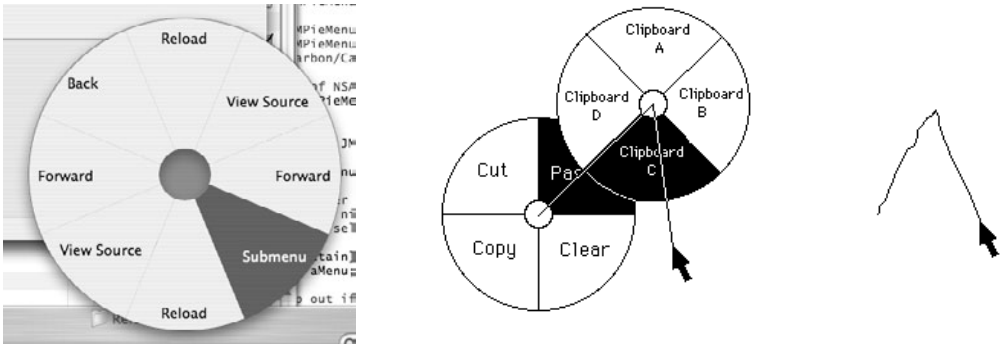


Abbildung 15.3: Ein geschachteltes Pull-down-Menü und der Weg des Mauszeigers zur Auswahl eines Eintrags in der ersten Schachtelungsebene

mmibuch.de/a/15.3

können realisiert werden, indem nach Verlassen des ersten Menüs an der aktuellen Mausposition das zugehörige Untermenü als Pop-up Menü in Tortenform erscheint. Befinden sich alle Menüeinträge stets an der gleichen Stelle, dann kann sich der Benutzer so einen Bewegungsablauf merken und diesen irgendwann ohne Betrachten der Menüs als Geste auf dem Bildschirm blind ausführen. Da für die Auswahl eines Eintrags im Tortenmenü nur die Richtung entscheidend ist, nicht die exakte Weglänge, macht es keinen Unterschied, wie groß diese Gesten ausgeführt werden. Nur die Winkel zwischen den Segmenten sind entscheidend. Der Lernprozess solcher Gesten wird unterstützt, indem der Mauszeiger auf dem Bildschirm eine Spur hinterlässt, und damit die ausgeführte Bewegung als Linienzug zu sehen ist. Diese Art von geschachtelten Tortenmenüs mit Spur des Mauszeigers heißen **Marking Menu** [69] und unterstützen einen fließenden Übergang vom Anfänger zum Experten. Tatsächlich benutzen Experten Menüs generell anders als Anfänger [24]: während Anfänger das Menü zunächst aufklappen, dann in linearer Zeit die Einträge lesen bis sie den gesuchten Eintrag gefunden haben, merken sich fortgeschrittene Benutzer recht schnell die Position der Einträge im Menü und benötigen für die Auswahl nicht mehr lineare Zeit, sondern nach dem **Hick-Hyman Gesetz** nur noch logarithmische Zeit bzgl. der Anzahl der Menüeinträge (siehe Abschnitt 3.6). Zu diesem Zeitanteil für die Entscheidung kommt dann noch der Zeitanteil für die motorische Ausführung, abhängig von der Art des Menüs.



mmibuch.de/a/15.4

Abbildung 15.4: Links: ein Tortenmenü, Rechts: ein geschachteltes Tortenmenü, bei dem der Mauszeiger eine Spur hinterlässt (Marking Menu) [69].

15.3 WYSIWYG

Im Interaktionskonzept der **Direkten Manipulation** werden die relevanten Informationseinheiten am Bildschirm grafisch repräsentiert und mit dem Zeigegerät direkt selektiert oder verschoben. Überträgt man diese Idee auch auf die Bearbeitung visueller digitaler Medien, wie z.B. Text oder Bilder, dann bedeutet das, dass Änderungen an einem solchen Medienobjekt am Bildschirm direkt so vorgenommen werden, wie sie sich später, beispielsweise im Druck, auch auswirken. Das Dokument wird am Bildschirm so gezeigt, wie es auch später auf dem Papier aussehen wird (bis auf die normalerweise niedrigere Auflösung des Bildschirms). Diese Situation lässt sich im Englischen prägnant ausdrücken als *what you see is what you get* (was man sieht, ist was man bekommt) und das wiederum liefert als Akronym den Begriff **WYSIWYG**.

Dieser Begriff tauchte zuerst im Zusammenhang mit Textsatz-Systemen und der Idee des sogenannten **Desktop Publishing** auf. Dabei sollte die Erstellung hochwertiger Druckerzeugnisse direkt vom Schreibtisch aus möglich werden und der WYSIWYG Ansatz sollte sicherstellen, dass auch ohne das Know-How einer Druckerei das Endprodukt jederzeit beurteilt werden konnte. Bereits die frühen Varianten der Desktop Metapher auf dem Xerox Star und dem Xerox Alto enthielten WYSIWYG Editoren für Text und dieser Ansatz scheint uns heute so selbstverständlich, dass wir kaum andere Vorgehensweisen in Erwägung ziehen. Gängige Textverarbeitungen wie MS Word, Apple Pages oder OpenOffice Writer arbeiten nach dem WYSIWYG Prinzip, wobei es Abstufungen gibt: in vielen dieser Softwarepakete kann beispielsweise eine Gliederungsansicht eingestellt werden, die dann die logische Struktur des Dokuments hervorhebt statt der visuellen Gestalt.

Ein extremes Gegenbeispiel ist das Textsatzprogramm TEX, mit dem beispielsweise dieses Buch gesetzt ist: Hier werden die verschiedenen Textteile mit Steuerkommandos versehen und der Text regelrecht *programmiert* (siehe auch Inhalt des Fensters in Abbildung 15.2). Eine Beurteilung des Endergebnisses ist erst nach einer Übersetzung (ähnlich dem

Kompilieren eines Programms) möglich. Der Vorteil dieser alternativen Vorgehensweise ist, dass systematische Änderungen am Dokument und die Sicherstellung von Konsistenz im Drucksatz sehr einfach sind, und dass das Ergebnis drucktechnisch sehr hohen Standards genügt. Das WYSIWYG Konzept wird auch bei der Bearbeitung anderer visueller Medien eingesetzt, beispielsweise in der Bildbearbeitung. Werkzeuge wie **Gimp** oder **Photoshop** zeigen die Auswirkungen einer Operation auf das Bild direkt an. Ein Gegenbeispiel wäre die Bildverarbeitung auf Kommandozeilenebene, beispielsweise mit dem Netpbm[1] Programmpaket.

Übungsaufgaben:

1. Ein echtes Zitat eines Hilfe suchenden Vaters an seinen Informatik studierenden Sohn: *Du hast doch Windows studiert. Ich hab das Internet gelöscht.* Analysieren Sie, was vermutlich passiert ist, welche Konzepte und welche Elemente der Metapher hier verwechselt werden, und wie diese logisch eigentlich zusammen gehören. Zeichnen Sie ein Diagramm dieser Zusammenhänge und erklären Sie diese in Worten, die Ihre Großeltern-Generation versteht. Falls Sie sich über diese recht blumige Aufgabe wundern: Es ist unsere fachliche Verantwortung als Informatiker, mit solchen Begriffen klar und korrekt umzugehen und unsere soziale Verantwortung, sie auch einem nichtfachlichen Publikum nahe bringen zu können.

2. Vergleichen Sie ein lineares Menü und ein Tortenmenü mit jeweils 8 Einträgen. Beide sollen als pop-up Menü implementiert sein. Schätzen Sie mithilfe von Fitts' Law die Zugriffszeiten auf alle 8 Einträge in beiden Menüs ab und erläutern Sie den Unterschied! Welche Gründe könnten dafür gesorgt haben, dass sich Tortenmenüs trotzdem bisher nicht durchgesetzt haben?

[1] `http://netpbm.sourceforge.net`

16 Die Benutzerschnittstelle des World Wide Web

Kaum ein anderes Medium hat unseren Umgang mit Informationen so nachhaltig verändert wie das World Wide Web, kurz **WWW** oder einfach Web. Tatsächlich wird der Begriff *Web* oft mit dem Begriff *Internet* austauschbar verwendet: Nutzer *gehen ins Internet* und nutzen Software mit Namen wie *Internet Explorer*, wenn sie Webseiten besuchen. Anbieter preisen den Zugang zu *Internet und E-Mail* an, wenn Sie den Zugang zum World Wide Web meinen. Das Web ist zum *Gesicht* des Internet geworden und viele Dienste wie E-Mail, News, oder soziale Netze bedienen sich heute seiner technologischen Grundlage. Dieses Kapitel ist ein Versuch, zumindest die grundlegenden Begriffe und Trends mit Bezug zur Benutzerschnittstelle einzuführen, kann aber der Fülle an Informationen zu diesem Thema natürlich nie gerecht werden.

16.1 Technische Grundkonzepte des Web

Technisch gesehen ist das WWW ein (sehr) großes verteiltes System aus vernetzten **Servern** und **Clients**. Auf den Servern sind die **Webseiten** gespeichert, von denen mehrere zusammenhängende Seiten auch als **Website** bezeichnete werden, sozusagen ein *Ort* innerhalb des Web. Im Deutschen herrscht durch den gleichen Klang dieser beiden Begriffe oft Verwirrung über ihren genaue Abgrenzung und sie werden munter durcheinander geworfen. Webseiten werden mit einer Client-software, dem sogenannten **Browser** angezeigt. Dieser Browser lief von Beginn an (bis auf exotische Ausnahmen wie Lynx[1]) immer innerhalb einer grafischen Desktop-Benutzerschnittstelle (siehe Kapitel 15) und wurde auch stets mit einem Zeigegerät wie beispielsweise der Maus bedient. Das ändert sich gerade mit der zunehmenden Nutzung des Web auf neueren Geräten wie Tablets und Smartphones.

Webseiten sind sogenannte **Hypertext**-Dokumente. Das bedeutet, dass sie nicht nur Text und Medienelemente enthalten, sondern auch Verweise (**Hyperlinks**) zu anderen Webseiten. Diese Hyperlinks unterliegen keinen festen Regeln bezüglich ihrer Struktur: Sie können beliebig innerhalb einer Website oder auch darüber hinaus verweisen und führen damit zu anderen Orten innerhalb des Web. Entlang der Hyperlinks begibt sich der Benutzer damit auf eine (abenteuerliche) Reise, was auch in metaphorischen Namen der Browser wie Internet *Explorer* (Engl. für *Erforscher*) oder *Safari* (Swahili für *Reise*) zum Ausdruck kommt. Alle diese Zusammenhänge wissen wir als Informatiker aus unserem täglichen Umgang mit dem Web. Anderen Teilen der Bevölkerung hingegen sind diese Begriffe und Zusammenhänge nicht unbedingt völlig klar, und wenn wir

[1] `http://lynx.browser.org`

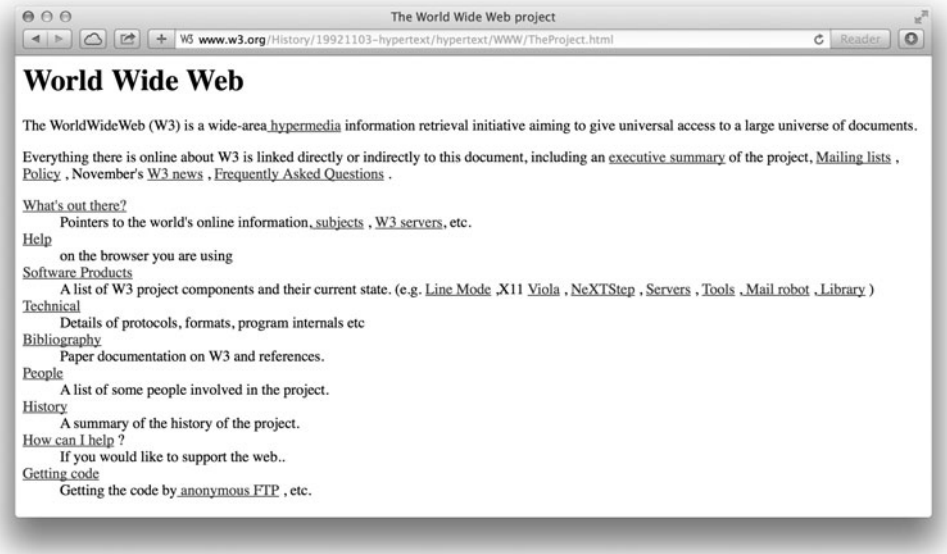


Abbildung 16.1: Replik einer der ersten Webseiten, angezeigt in einem heutigen Browser

einmal wieder die Computer-Probleme unserer (Groß-) Elterngeneration lösen sollen, kann es durchaus sein, dass wir auf die Frage, welcher Browser denn verwendet wurde, nur einen fragenden Blick bekommen.

Entwickelt wurden die grundlegenden technischen Standards des WWW 1989-1991 am **CERN** Institut durch **Tim Berners-Lee**. Er definierte die Beschreibungssprache **HTML** (*Hypertext Markup Language*) und programmierte die erste Server- und Browser- Software. Das so entstandene System wurde zunächst dazu genutzt, wissenschaftliche Dokumente zwischen den Forschern am CERN auszutauschen. Der Fokus lag damit also zunächst auf Text und Abbildungen, sowie den genannten **Hyperlinks**. Abbildung 16.1 zeigt eine der ersten Webseiten im damals typischen undekorierten (*Plain HTML*) Stil. Erst die Kommerzialisierung des WWW brachte dann die medienlastige Gestaltung heutiger Webseiten mit sich. Das ursprüngliche HTML sah die nötigsten Darstellungsmittel für Texte vor, wie Überschriften verschiedener Ebenen und Textauszeichnungen wie kursiv oder fett. Wesentliche Vorteile gegenüber anderen damaligen Internet-basierten Informationssystemen wie Usenet **News** oder **Gopher** waren die freie Struktur der Hyperlinks sowie die Möglichkeit, Bilder in die Texte mit einzubinden. Später kamen andere Medienobjekte wie Audio und Video hinzu, was damals aber noch eine Herausforderung an die Rechenleistung bedeutete.

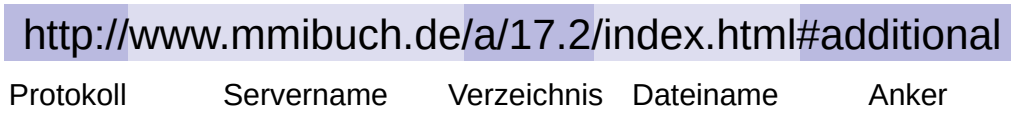


Abbildung 16.2: Aufbau eines Uniform Resource Locator (URL) aus Protokoll, Servername, Verzeichnis auf dem Server, Dateiname und einem optionalen Anker innerhalb der Seite

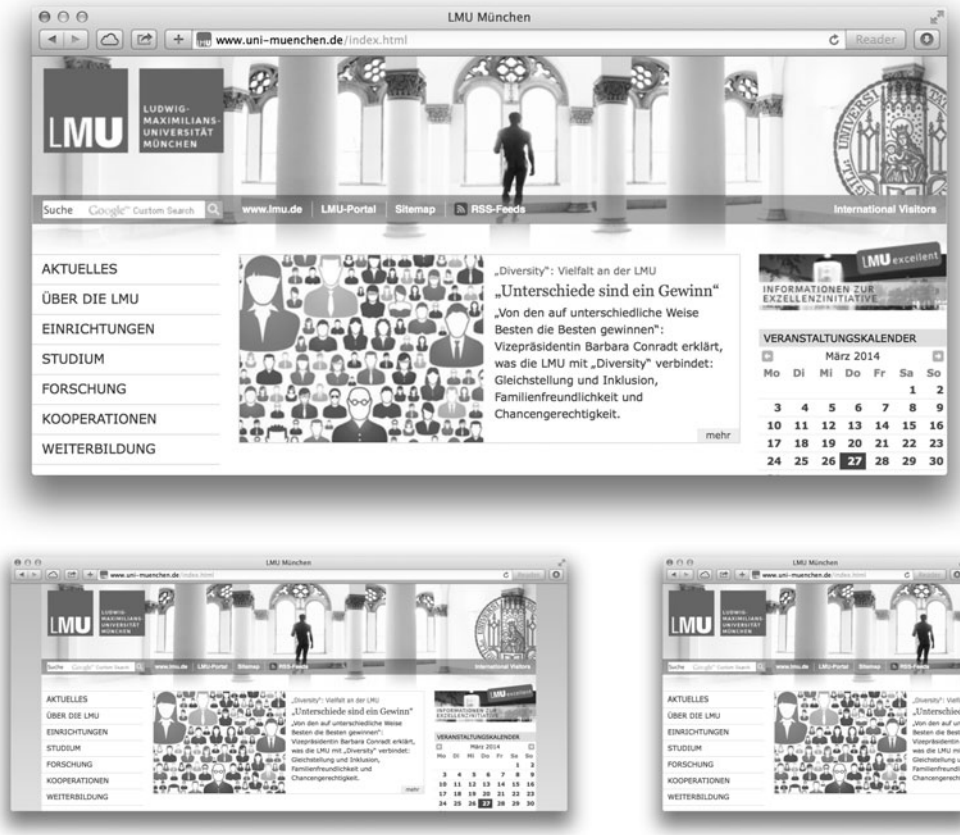
Zur Übertragung der HTML-Seiten definierte Berners-Lee das Hypertext Transfer Protocol **HTTP** zwischen Server und Browser. Eine Übertragung im HTTP besteht grundlegend aus einer Anfrage des clients an den Server (**HTTP request**) und einer Antwort des Servers (**HTTP response**). Eine einzelne Ressource im Web, beispielsweise eine Webseite oder ein Bild, wird durch eine Web Adresse, den sogenannten *Uniform Resource Locator*, **URL** adressiert. Diese[2] besteht aus einem Bezeichner für das verwendete Protokoll, einem Servernamen und dem Namen einer Datei auf diesem Server. Hinzu kommen Konventionen für Sprungmarken innerhalb der Datei oder sonstige Parameter, die an den Server übergeben werden (siehe Abbildung 16.2).

Die technischen Standards des WWW werden vom World Wide Web Consortium (**W3C**) verwaltet und haben seit dem Beginn des WWW teilweise substanzielle Änderungen durchlaufen. Während HTTP immer noch in der Version 1.1 von 1999 verwendet wird, ist man bei HTML mittlerweile bei Version 5 angelangt mit Umwegen und teilweise parallel entwickelten Standards. Dabei war die Weiterentwicklung durch verschiedene Stellen und mit verschiedener Motivation getrieben, was stellenweise durchaus zu einem technologischen Wildwuchs führte. Die Gestaltung und das Layout der Webseiten durchlief dabei verschiedene Modetrends.

16.2 Layout: fließend, statisch, adaptiv, responsiv

Eine ursprüngliche Kernidee von HTML war es, kein festes Seiten-Layout vorzugeben. Frühe Webseiten wie die in Abbildung 16.1 passten ihr Layout an die Fenstergröße des Browsers an und der Browser war dafür verantwortlich, die Inhalte immer sinnvoll und bestmöglich darzustellen. Diese Art des Layouts nennt man **fließendes Layout**, da der Text bei Veränderung der Fenstergröße regelrecht um die anderen Elemente herum fließt. Mit der wachsenden Kommerzialisierung des Web wuchs der Anspruch, auf die visuelle Gestaltung der Seiten mehr Einfluss nehmen zu können. Gestalter aus dem Print-Bereich begannen, mit festen Satzspiegeln und Layout **Grids** zu arbeiten, wie sie bei Druckerzeugnissen wie Büchern oder Zeitschriften verwendet werden. Dabei entstanden **statische Layouts**, die in einer bestimmten Fenstergröße oder Bildschirmauflösung sehr gut aussehen, in anderen Umgebungen aber teilweise völlig versagen.

[2] Obwohl *locator* im Englischen ein männliches Wort ist, hat sich im Deutschen der weibliche Artikel für *URL* eingebürgert, wohl in Anlehnung an *die* Adresse.



mmibuch.de/a/16.3

Abbildung 16.3: Statisches Layout, oben in der optimalen Fensterbreite. Bei einem breiteren Fenster werden rechts und links leere Flächen ergänzt, bei einem schmaleren Fenster werden Inhalte abgeschnitten.

Ein dabei noch recht unkritisches Beispiel zeigt Abbildung 16.3. Durch die wachsende Verbreitung mobiler Endgeräte seit Beginn dieses Jahrhunderts greifen immer mehr Menschen mobil auf Webseiten zu. Dabei ist ein optimales Layout besonders wichtig, da sich die Anzeigefläche und Interaktionsmöglichkeiten von der Desktop-Umgebung doch stark unterscheiden. Ein statisches Layout, das für eine bestimmte Fenstergröße am Desktop optimiert wurde, versagt hier praktisch immer, da bei voller Darstellung alle Inhalte unlesbar klein werden. Zoom und Pan (siehe Abschnitt 9.2) ermöglichen zwar prinzipiell wieder eine Interaktion auf solchen Geräten, sind jedoch umständlich und von einer optimalen Bedienung weit entfernt. Aus dieser Situation heraus entstand das sogenannte **adaptive Layout**. Dabei werden für verschiedene Geräteklassen verschiedene

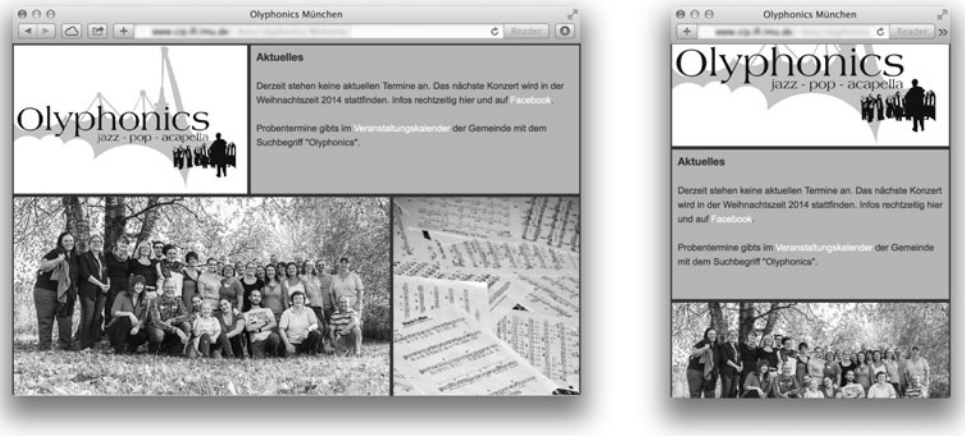


Abbildung 16.4: Responsives Layout: der Designer legt fest, wie die grundlegenden Elemente der Seite in verschiedenen Situationen angeordnet werden, und der Browser übernimmt das detaillierte Layout.

mmibuch.de/a/16.4

statische Layouts vorbereitet und je nach Gerätetyp die passendste Version vom Server geliefert. Technisch wird dies dadurch ermöglicht, dass der **HTTP request** Angaben über das verwendete Betriebssystem, den Browser und das Endgerät enthält, und der Server in Abhängigkeit von diesen Variablen eines der verschiedenen Layouts auswählt und zurückschickt. Offensichtlicher Nachteil dieser Methode ist der erhöhte Arbeitsaufwand und die Tatsache, dass der Web-Entwickler eigentlich dem Gerätemarkt immer hinterher programmiert. Sobald ein neues Gerät erscheint, muss überprüft werden, ob eines der bestehenden Layouts darauf passt, und ob die Auswahlregeln ggf. auch das richtige Layout liefern. Um diesem neuerlichen Missstand zu entgehen, entstand das sogenannte **responsive Layout**. Dieses verlagert die Detail-Anpassung vom Designer wieder zurück auf den Browser: Der Web Designer legt dabei nur die grobe Anordnung verschiedener logischer Elemente der Seite für verschiedene Grundsituationen fest, und der Browser übernimmt wieder das detaillierte Layout auf dem konkreten Endgerät (siehe Abbildung 16.4). Technisch wird dies möglich durch die Trennung von Inhalt und Darstellung in **HTML 5** und **CSS** in Verbindung mit sogenannten **media queries**. So kann der Browser selbst das Layout auf die genaue Fenstergröße nach den im Layout festgelegten Regeln anpassen, nachdem er aus mehreren Grund-Layouts (beispielsweise für Desktop, Smartphone und Druck) das richtige ausgewählt hat.

Mit der rasanten Entwicklung des Web ist abzusehen, dass schon in wenigen Jahren weitere Modetrends beim Layout und weitere technische Standards entstehen werden. Ziel dieses Abschnitts ist es daher nur, die Ideen hinter diesen Trends darzustellen. In die aktuellen Trends und technischen Details wird man sich zum gegebenen Zeitpunkt immer wieder neu einarbeiten müssen.

16.3 Inhalte: statisch oder dynamisch

Neben dem Layout von Webseiten kann auch deren Inhalt sich dynamisch verändern. Dies widerspricht eigentlich der ursprünglichen Idee des Web, bei der ja zu einer URL immer die gleiche **statische Webseite** geliefert wurde. Die Kernidee **dynamischer Webseiten** besteht darin, dass der Server zu einem gegebenen **HTTP request** zuerst ein individuelles Dokument berechnet und dies dann zurückgibt. Diese *serverseitige* Berechnung kann aus einem Datenbankzugriff bestehen, vom Webserver selbst oder auch von einem größeren Softwaresystem auf dem Server ausgeführt werden. Auf diese Weise wurde es erst möglich, Shop-Systeme mit Bestell- und Bezahlvorgängen im Web abzubilden: Der individuelle Einkaufskorb sieht für jeden Benutzer anders aus. Seit dieser Entwicklung spricht man daher nicht mehr nur von Webseiten, sondern gelegentlich von regelrechten **Webanwendungen**.

An dieser Stelle kommt nun zum Tragen, dass das Web ursprünglich nicht für eine solche Architektur konzipiert war: Für die Beantwortung eines HTTP request gibt es keinerlei Zeitgarantien, und beim Klick auf einen Hyperlink kann es je nach Umfeld recht lange dauern, bis die zugehörige Seite vom Server geliefert wird und auch vollständig übertragen ist. Dies widerspricht der in Abschnitt 6.2 aufgestellten Forderung, dass Feedback in interaktiven Systemen idealerweise innerhalb von 100ms gegeben werden sollte. Um diesem Problem zu begegnen und Webanwendungen zeitlich besser kontrollierbar zu machen, entwickelte man Techniken, die die Berechnung *clientseitig* ausführen. Bei **AJAX**[3]-basierten Webanwendungen wird die Benutzerschnittstelle insgesamt (in Form eines **JavaScript** Programms) herunterladen und dann im Browser ausgeführt. Die Verbindung zum Server erfolgt nicht mehr über einzelne HTTP requests, sondern über eine separate Netzwerkverbindung, durch die **XML** Datenstrukturen in beide Richtungen übertragen werden. So kann die Webanwendung zeitnah auf Benutzereingaben reagieren und ist damit unabhängiger von den Unwägbarkeiten des Übertragungsweges.

16.4 Nutzungsarten: Web x.0 (x = 1,2,3,...)

Mit der Entwicklung dynamischer Inhalte wurde es immer einfacher, nun selbst Inhalte für das Web zu generieren ohne dafür selbst HTML-Code schreiben zu müssen. Von Produktbewertungen beim Online-Shopping über Blogs bis zu sozialen Netzen treffen wir heute überall von Benutzern generierte Inhalte an, ohne dass diese Benutzer dafür HTML schreiben oder einen Server betreiben müssen. Dieser qualitative Schritt hat auch zu dem Begriff **Web 2.0** geführt, womit das *alte* statische Web automatisch zum **Web 1.0** wurde. Mit der Zeit wollte man die im Web von so vielen Menschen generierten Inhalte auch durch Maschinen nutzbar machen, um daraus automatisch strukturiertes Wissen ableiten zu können. Technisch gesehen bestehen Webseiten aus reinen Textdateien sowie den eingebetteten Medienobjekten wie Bildern oder Filmen. Diesen automatisch eine Bedeutung zuzuordnen und sie sinnvoll miteinander in Beziehung zu setzen, ist nicht ohne Weiteres möglich, da die tiefe semantische Analyse von Text oder gar Bildern nach wie vor ein algorithmisch schwieriges Problem ist. Die Lücke zwischen dem reinen Dateiinhalt (Syntax) und der Bedeutung (Semantik) (Engl: **semantic gap**)

[3] AJAX = Asynchronous JAvascript and Xml

lässt sich nicht oder nur in eng begrenzten Sonderfällen automatisch schließen. Hätte man zu Webseiten allerdings eine formalisierte und automatisch verarbeitbare Beschreibung ihrer Bedeutung oder zumindest der Art von Information, die sie enthalten, dann wäre die automatische Suche nach sinnvoller und nützlicher Informationen und die Ableitung neuner Information daraus viel besser zu bewerkstelligen.

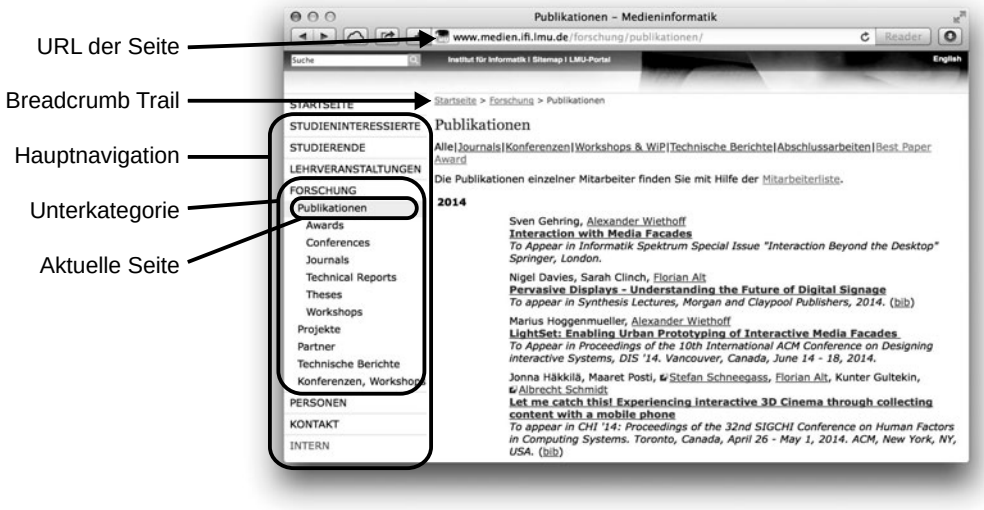
Aus diesem Grund entstanden Formalismen wie das **Resource Description Framework**[4] (**RDF**) und die **Web Ontology Language** (**OWL**), die auf eine solche semantische Beschreibung von Webseiten und ihren Beziehungen untereinander abzielen. Die Vision dahinter ist das sogenannte **Semantic Web** oder **Web 3.0**, in dem Seiten anhand ihrer Bedeutung gefunden und zueinander in Beziehung gesetzt werden können. Eine konkrete Ausprägung solcher semantischer Annotationen ist beispielsweise das Hinzufügen geografischer Information zu Seiten, die mit Orten auf der Erde zu tun haben. Dies ermöglicht es, beim Besuch dieser Orte genau die relevanten Seiten, die sich auf den jeweiligen Ort beziehen, anzuzeigen. Das Semantische Web ist derzeit noch stark in der Entwicklung begriffen und seine Ausprägung in der Zukunft ungewiss.

16.5 Wie Webseiten gelesen werden

Der Umgang des menschlichen Nutzers mit Webseiten unterscheidet sich stark vom Umgang mit anderen Textdokumenten. Während wir in einem Buch seitenweise Text lesen und uns dazu in der Regel die Zeit nehmen, ganze Kapitel und Abätze zu verstehen, fliegt unser Blick über Webseiten in wenigen Sekunden dahin. Beim Betrachten einer Webseite sucht unser Auge dabei nach auffallenden Elementen, wie beispielsweise farblich hervorgehobenen Links und Schlüsselwörtern. In Abschnitt 2.1.3 haben wir gelernt, dass eine farbliche Hervorhebung das Finden von Links mittels **präattentiver Wahrnehmung** ermöglicht, also besonders schnell macht. Dies wurde bereits von den ersten Webseiten unterstützt (siehe Abbildung 16.1). Außerdem nutzen wir beim Überfliegen von Webseiten aus, dass diese mittlerweile oft einer festen Struktur folgen: am linken oder oberen Rand erwarten wir globale Informationen wie Firmenlogo oder Kopfzeile, sowie Navigationselemente, in der Mitte der Seite bedeutungsvollen Inhalt und am rechten Rand optionale Zusatzinformation oder Werbung (siehe Abbildung 16.3).

All dies hat Auswirkungen auf die inhaltliche, strukturelle und grafische Gestaltung von Webseiten: Die Inhalte müssen kurz und prägnant sein. Text muss in wenigen Worten oder Sätzen seine Botschaft vermitteln, umfangreichere Zusatzinformationen sollten dabei optional angeboten und verlinkt werden. Lange Texte werden im Web in den allermeisten Fällen nicht gelesen, sondern nur überflogen und nach Schlüsselwörtern abgesucht. Aussagekräftige Bilder vermitteln auf einen Blick oft mehr als Texte und unterstützen das schnelle Absuchen ebenfalls. Strukturell sollten Webseiten sich heutzutage an bestehende Konventionen halten, damit der Benutzer bestimmte Elemente auch an den erwarteten Orten vorfindet. Dies gilt vor allem für die Navigationselemente. Information, deren Existenz auf der Seite verschleiert werden soll, kann man sogar regelrecht verstecken, indem man sie an Stellen anordnet, an denen sich normalerweise Werbung befindet, und sie dann auch grafisch wie Werbung aussehen lässt. Grafisch

[4] http://www.w3.org/RDF/



mmibuch.de/a/16.5

Abbildung 16.5: Navigationselemente einer Webseite: Links die hierarchisch aufgebaute Hauptnavigation, Oben der Breadcrumb Trail, Ganz oben die strukturell gleiche URL

sollten Webseiten so einfach wie möglich gehalten werden. Die grafische und typografische Gestaltung hat hier vor allem den Zweck, das schnelle Absuchen der Seite nicht zu behindern. Text sollte in gut lesbarer Größe und mit gutem Kontrast dargestellt werden. Links und wichtige Begriffe sollten präattentiv wahrnehmbar sein (siehe Abschnitt 2.1.3). Zudem werden Webseiten in den meisten Fällen auch nicht gescrollt. Was nicht beim ersten Erscheinen der Webseite im Fensterbereich sichtbar ist, wird oft auch nicht mehr durch Verschieben des Scrollbalkens zu Tage gefördert. Auch so lassen sich Informationen (beabsichtigt oder unbeabsichtigt) verstecken. Das Impressum, das jede Webseite heute verpflichtend haben muss, ist ein Beispiel für ein solches oft bewusst verstecktes Element: ein Link dorthin findet sich meist in der Fußzeile in kleiner, unauffälliger Schrift. Damit ist der gesetzlichen Regelung Genüge getan, die Aufmerksamkeit des Benutzers wird jedoch erfolgreich daran vorbei gelenkt.

16.6 Orientierung und Navigation

Drei wichtige Kriterien bei der **heuristischen Evaluation** interaktiver Systeme (siehe Abschnitt 13.2.2) sind *Visibility of system status*, *User control and freedom* und *Recognition rather than recall*. Diese Forderungen werden auf strukturierten Websites in der Regel durch Navigationselemente erfüllt. Die Navigation einer Website soll also erkennen lassen, was alles auf dieser Site zu finden ist (*recognition*) und wo sich

der Benutzer gerade befindet (*status*). Außerdem soll sie natürlich ermöglichen, sich zu anderen Teilen der Site zu bewegen (*control*). Abbildung 16.5 zeigt ein Beispiel für die hierarchische Hauptnavigation einer Website, die diese Forderungen erfüllt: Sie befindet sich zunächst einmal am erwarteten Ort, nämlich dem linken Rand der Seite, und zeigt an, welche Themenbereiche es auf dieser Site gibt (*recognition*). Beim Anwählen einer Unterkategorie wird diese aufgeklappt (*control*) und zeigt die darin verfügbaren Seiten oder weiteren Unterkategorien an. Die aktuell angewählte Seite ist grau hinterlegt (*status*). Auf dieser Seite wird außerdem ein weiteres weit verbreitetes Navigationselement verwendet, der sogenannte **Breadcrumb Trail**. Wie im Märchen von Hänsel und Gretel zeigen Brotkrumen den Weg zurück zum Ausgangspunkt: Die nacheinander aufgeklappten Unterkategorien erscheinen als eine Art Spur immer spezifischerer Begriffe. Interessanterweise deckt sich die logische Hierarchie in diesem Beispiel auch mit der implementierten Ordnerstruktur auf dem Webserver, und weil die Verzeichnisse dort die gleichen Namen tragen wie die Titel der zugehörigen Unterkategorien, stimmt die URL der einzelnen Seite strukturell mit dem angezeigten Breadcrumb Trail überein.

16.7 Die sozialen Spielregeln: Netiquette im Web

Für den Umgang miteinander im Internet entstand schon recht früh eine informelle Sammlung von Regeln für gutes Benehmen. Diese *Etiquette* im Netz wird oft als **Netiquette** bezeichnet. Dabei standen ursprünglich andere Medien im Vordergrund, wie *E-Mail*, *chat* und *Usenet* Foren, und viele der Regeln beziehen sich daher auf die direkte Kommunikation zwischen einzelnen Menschen oder zwischen einem Menschen und einer Gruppe. Einige der Regeln lassen sich jedoch auch sehr gut auf das Web beziehen, und natürlich insbesondere auf die seit einigen Jahren boomenden **sozialen Netze**:

- *Den Menschen im Blick behalten*: Die Grundregel menschlichen Zusammenlebens, andere so zu behandeln wie man selbst behandelt werden möchte, geht durch die distanzierte Kommunikation im Internet leicht verloren. Beim Verfassen von Webseiten oder beim Umgang in sozialen Netzen sollte man immer die Auswirkungen des Geschriebenen auf betroffene Personen und deren Reaktion im Auge behalten. Unangemessenes Verhalten fällt letztlich auch negativ auf den Autor zurück.

- *Gleiche Verhaltensregeln im Web wie im echten Leben*: Obwohl es im Web sehr einfach ist, Gesetze und Spielregeln zu überschreiten, und obwohl die Verfolgung dieser Straftaten oft schwierig ist, gelten doch die gleichen Regeln. Insbesondere das Urheberrecht verpflichtet uns zu einem sorgfältigen Umgang mit dem geistigen Eigentum anderer, beispielsweise in Form korrekter Quellenangaben.

- *Zeit und Bandbreite anderer respektieren*: Statt langwieriger Diskussionen und Ausführungen sollten Webseiten schnell zum Punkt kommen, kurze und prägnante Sprache verwenden, und damit den eingangs geschilderten Lesestil von Webseiten unterstützen. Bandbreite bezieht sich damit einerseits auf die geistige Kapazität des Lesers, andererseits aber auch auf die technische Kapazität: wird ein Foto in voller Auflösung in eine Webseite eingebunden und dann innerhalb des HTML-Codes auf Briefmarkengröße skaliert, dann stellt dies eine Verschwendung von

Bandbreite dar und führt zu unnötig langen Ladezeiten der Seite. Ein vorher passend skaliertes Bild verbraucht womöglich nur ein Hundertstel des Datenvolumens. Gleiches gilt übrigens für E-Mail.

- *Gut aussehen online*: Wer eine Webseite verfasst, wird nach ihr beurteilt. Rechtschreibfehler, schlechte Fotos, veraltete Informationen, tote Links, und technisch oder ästhetisch fragwürdige Seiten sollten allesamt vermieden werden, denn sie bremsen oder stören den Leser und fallen damit negativ auf den Autor zurück. Auch Seiten, die für eine bestimmte Zielplattform optimiert sind und auf anderen Plattformen schlecht oder gar nicht funktionieren, widersprechen dem Grundgedanken des Web.

- *Privatsphäre respektieren*: Informationen über andere, die wir persönlich erlangt haben, haben nichts in der Öffentlichkeit zu suchen! Gerade in sozialen Netzen ist es sehr einfach, private Informationen, Bilder oder Nachrichten an eine große Öffentlichkeit weiterzuleiten. Es sollte sich von selbst verstehen, dass so etwas nur mit dem ausdrücklichen Einverständnis des ursprünglichen Autors bzw. der abgebildeten Person geschieht, oder noch besser durch sie oder ihn selbst.

Diese Regeln sind nur eine Auswahl eigentlich offensichtlicher Spielregeln für den Umgang miteinander. Eine *amtliche* Version der Netiquette findet sich in RFC 1855[5], bezieht sich jedoch in weiten Teilen auf heute nicht mehr so verbreitete Medien. Die Regeln der Netiquette unterliegen auch einem zeitlichen Wandel und sind in verschiedenen Kulturen unterschiedlich ausgeprägt. Es bleibt uns also nichts anderes übrig, als in einer konkreten Situation die jeweiligen Regeln für diesen Fall erneut zu recherchieren oder durch Beobachtung anderer selbst abzuleiten.

Übungsaufgaben:

1. Überprüfen Sie, wie gut Ihre fünf liebsten Webseiten auf verschiedenen Zielplattformen (Desktop PC, Smartphone, Tablet, Ausdruck auf Papier) funktionieren. Welche Layout-Strategien wurden verwendet und wie gut ist die Umsetzung jeweils gelungen?

2. Analysieren Sie ihre private Webseite oder die ihrer Institution (Uni, Firma, Verein,...) unter dem Aspekt der Angemessenheit der Inhalte! Sind Texte kurz und prägnant, Bilder aussagekräftig und ästhetisch ansprechend? Ist die technische Umsetzung sauber und entspricht sie den aktuellen Standards? Werden die Regeln der Netiquette eingehalten? Falls es Ihre eigene Webseite ist, korrigieren Sie die gefundenen Fehler. Falls die Seite Ihrer Institution gehört, weisen Sie die Verantwortlichen höflich und mit sachlichen Begründungen darauf hin. Dokumentieren Sie deren Reaktion.

[5] http://tools.ietf.org/html/rfc1855

17 Interaktive Oberflächen

Spätestens seit der Einführung leistungsfähiger Touchscreenbasierter Smartphones 2007 entwickelt sich die Interaktion mittels Touch-Eingabe zur vorherrschenden Art der Interaktion in vielen Alltagssituationen jenseits des Büros. Tablets, Smartphones und berührungssensitive Bildschirme an öffentlichen Terminals oder Verkaufsautomaten sind die derzeit verbreiteten Arten **Interaktiver Oberflächen**, aber auch interaktive Wandtafeln haben Einzug in unsere Schulzimmer gehalten und interaktive Tische setzen sich zunehmend in Messe- und Museums-Kontexten durch.

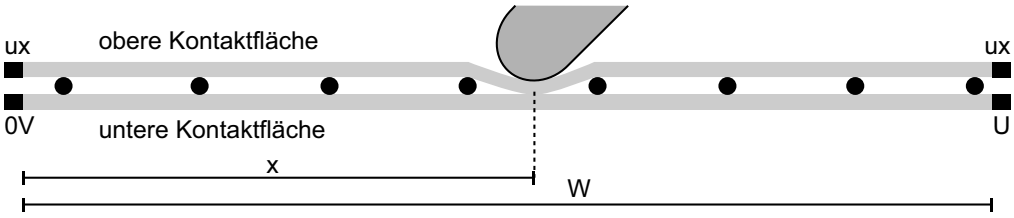
17.1 Grundlagen zu Touch und Multi-Touch

All diesen Geräten gemeinsam ist die Eingabe mittels Berührung mit einem (**Single-Touch**) oder mehreren Fingern oder Handflächen (**Multi-Touch**). Bereits seit den 1980er Jahren existieren Forschungsarbeiten zum Thema Multi-Touch-Eingabe und auf den dort entwickelten Interaktionskonzepten basieren viele heutige Geräte. Dabei war lange Zeit die Sensortechnologie der begrenzende Faktor.

17.1.1 Sensortechnologien für Touch

Die technisch einfachste Vorrichtung zur Erkennung von Touch-Eingaben ist der **resistive Touch-Sensor** (siehe Abbildung 17.1). Dabei sind zwei leitende Folien mechanisch so miteinander verbunden, dass sie im Ruhezustand keinen Kontakt zueinander haben, durch Druck an einer bestimmten Stelle jedoch dort ein Kontakt geschlossen wird. Nun legt man an eine der Folien in horizontaler Richtung eine Spannung an, die über die gesamte Breite der Folie gleichmäßig abfällt. Wird ein Kontakt geschlossen, so kann man an der anderen Folie eine Spannung abgreifen, die sich mit der horizontalen Position des Kontaktpunktes verändert und damit die Bestimmung der X-Koordinate des Kontaktpunktes ermöglicht. Legt man danach an die andere Folie in gleicher Weise eine Spannung in vertikaler Richtung an, so lässt sich an der ersten Folie analog die Y-Position des Kontaktpunktes bestimmen. Erzeugt man mehrere Kontaktpunkte, so liefert der Sensor den Mittelwert der jeweiligen Positionen zurück. Resistive Touch-Sensoren benötigen mechanischen Druck bei der Eingabe und können mit Stiften oder Fingern bedient werden, jedoch keine Objekte oder visuellen Marker erkennen. Sie liefern keine absolute Position, sondern müssen für die korrekte Umrechnung von Spannungen in Positionen kalibriert werden.

Die derzeit am weitesten verbreitete Technologie ist der **kapazitive Touch-Sensor**. Er wird in den heute gängigen Tablets und Smartphones verwendet und es gibt mehrere prinzipielle Funktionsweisen. Allen gemeinsam ist ein in die Sensor-Oberfläche eingear-



mmibuch.de/a/17.1

Abbildung 17.1: Resistiver Touch Sensor: Um die x-Position zu messen, wird an der unteren Schicht die Spannung U angelegt und oben ux ausgelesen, sobald durch Druck ein Kontakt entsteht. Es gilt $\frac{ux}{U} = \frac{x}{W}$ oder $x = W\frac{ux}{U}$.

beitetes Gitter oder Raster aus Leiterbahnen. Bringt man einen Finger oder eine andere größere elektrisch leitende Masse in die Nähe dieser Leiterbahnen, so verändert sich die Kapazität der Bahnen zueinander oder zur elektrischen *Masse*. Durch regelmäßiges Abfragen aller Bahnen oder Kreuzungen kann so ermittelt werden, an welchen Positionen eine Berührung vorliegt. Werden mehrere benachbarte Sensorpunkte abgedeckt, so kann die Position durch Interpolation sogar mit höherer Auflösung als der des Sensors selbst berechnet werden. Die Sensorausgabe kann man wie ein Bitmap-Bild behandeln und durch Bildanalyse lassen sich nicht nur die Mittelpunkte der Kontaktpositionen berechnen, sondern auch komplexere Formen wie Handflächen erkennen. Kapazitive Touch-Sensoren benötigen keinen mechanischen Druck, jedoch ein elektrisch leitendes Material auf der Sensoroberfläche. Sie können Finger, Hände, spezielle Stifte und kapazitive Marker erkennen und müssen bezüglich der Position nicht kalibriert werden.

Bei größeren interaktiven Oberflächen wie z.B. interaktiven Tischen oder Wänden trifft man derzeit auch häufig auf **optische Touch-Sensoren**. Viele relativ einfach konstruierte und in Forschungslabors gebaute interaktive Tische verwenden dabei das **FTIR** oder das **DI** Verfahren. Diese beiden Verfahren arbeiten mit Infrarot-Licht und lassen sich mithilfe preisgünstiger Kameras und Infrarot-LEDs für wenige Euro realisieren. FTIR (siehe Abbildung 17.2) steht für *frustrated total internal reflection* und beruht auf der Tatsache, dass Licht an einer Grenzfläche zwischen Materialien mit unterschiedlichem Brechungsindex (z.B. Plexiglas und Luft) vollständig reflektiert wird. Berührt ein Material mit anderem Brechungsindex (z.B. ein Finger oder eine Silikon-Folie) die Fläche, so tritt an dieser Stelle Licht aus und beleuchtet das Objekt, was im Kamerabild als helle Region zu erkennen ist (Abbildung 17.2).

DI (siehe Abbildung 17.3) steht für *diffuse illumination* und verwendet Lichtquellen, die die Interaktionsfläche von unten diffus ausleuchten (Abbildung 17.2). Berührt ein Objekt die Fläche, so wird es beleuchtet und ist ebenfalls im Kamerabild zu erkennen. FTIR-Sensoren benötigen mechanischen Druck, sind dafür aber recht robust gegenüber wechselndem Umgebungslicht. DI-Sensoren benötigen keinen Druck und können optische Marker erkennen, sind dafür aber viel anfälliger gegen Störlicht wie z.B. direktes Sonnenlicht. Beide Funktionsprinzipien können auch miteinander kombiniert werden, um die robuste Touch- und Marker-Erkennung gleichzeitig zu ermöglichen. Sowohl FTIR

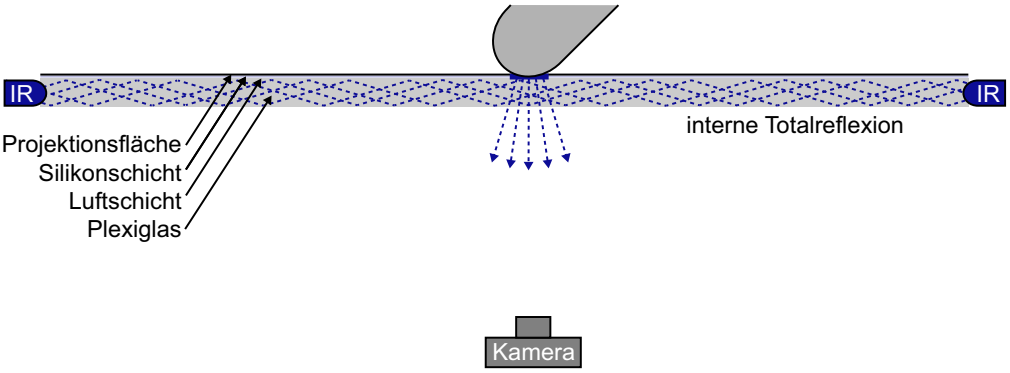


Abbildung 17.2: FTIR Touch Sensor: Licht wird seitlich durch IR-LEDs eingespeist und an der Grenzfläche Plexiglas-Luft totalreflektiert. Drückt ein Objekt die Silikonschicht an das Plexiglas, dann tritt dort Licht aus und erleuchtet das Objekt, das daraufhin im Kamerabild sichtbar wird.

mmibuch.de/a/17.2

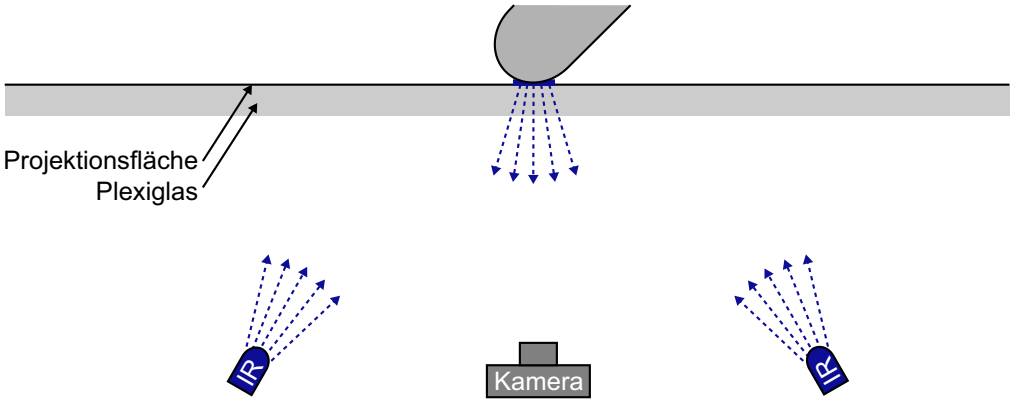
als auch DI-Sensoren müssen bezüglich ihrer Geometrie und der Bildhelligkeit kalibriert werden und der billigen Herstellung steht derzeit noch meist ein Mangel an Robustheit gegenüber. Durch den benötigten optischen Weg für die Kamera lassen sich diese Sensoren auch praktisch nur mit Projektions-Displays kombinieren. Daneben gibt es auch kommerziell verfügbare Interaktive Oberflächen mit optischem Funktionsprinzip, wie z.B. der Microsoft PixelSense[1]. Bei diesem sind optische Sensoren direkt in die Anzeigeelektronik des Bildschirms integriert, wodurch der gesamte Bildschirm auch zu einer Art Bildsensor wird. Diese Technologie kann Finger und Hände, aber auch Objekte und optische Marker erkennen, benötigt keine Kalibrierung und verhält sich recht robust gegenüber Störlicht.

Vor allem die FTIR Technologie hat bei ihrer Vorstellung durch **Jeff Han** 2006 in der Forschung einen wahren Boom für Interaktive Oberflächen ausgelöst. Seit es möglich wurde, für wenige Euro einen Multi Touch Tisch zu bauen, begannen viele Forschungsgruppen, die Interaktion damit zu untersuchen. So entstanden eine Reihe von mehr oder weniger experimentellen Konzepten. Formal lassen sich diese beispielsweise mit einem bereits weit über 20 Jahre alten Modell beschreiben.

17.1.2 Buxtons Modell der 3 Zustände

Bill Buxton [18] beschreibt die Übergänge zwischen verschiedenen Zuständen bei der Interaktion in grafischen Benutzerschnittstellen in Form von Zustandsautomaten. Ausgangspunkt ist die **Maus** (siehe Abbildung 17.4 Oben links). Ist die Maustaste nicht gedrückt, dann befindet sich das System in Zustand 1 (Tracking). Wird die Maus in Zustand 1 bewegt, dann führte das damals nur zu einer Positionierung des Mauszeigers, löste aber keine Funktion in der grafischen Schnittstelle aus. In heutiger Terminologie

[1] www.pixelsense.com



mmibuch.de/a/17.3

Abbildung 17.3: DI Touch Sensor: Licht wird von unten durch die Plexiglasscheibe gestrahlt. Sobald ein Objekt die Oberfläche berührt oder ihr sehr nahe kommt, wird es davon erleuchtet und im Kamerabild sichtbar.

wird dieser Zustand 1 oft als **Hover** (vom englischen *hover* = schweben) bezeichnet und löst sehr wohl Reaktionen der Schnittstelle aus, wie beispielsweise das Einblenden kurzer Erklärungstexte zu UI-Elementen, sogenannter **Tooltips**. Wird die Maustaste gedrückt, dann wechselt das System in den Zustand 2 (Dragging). Fand das Drücken über einem grafischen Objekt, beispielsweise einem Icon, statt, dann hängt dieses Objekt nun am Mauszeiger und wird mit ihm bewegt, solange sich das System in Zustand 2 befindet. Lässt der Benutzer die Maustaste los, so wechselt das System zurück in den Zustand 1 und das Objekt wird an der aktuellen Position des Mauszeigers losgelassen.

Obwohl die Interaktion mittels Touch zunächst sehr ähnlich erscheint, funktioniert sie doch bei genauerer Betrachtung ganz anders (siehe Abbildung 17.4 Oben rechts). Solange kein Finger den Sensor berührt, befindet sich das System im Zustand 0 (Out of Range), in dem auch keine Position des Fingers ermittelt werden kann. Berührt nun ein Finger den Sensor, beispielsweise ein **Touch-Tablet**, so wechselt das System in den Zustand 1 (Tracking), in dem die Fingerposition ermittelt wird und der Mauszeiger ihr am Bildschirm folgt. Da sich der Finger nun aber schon auf dem Bildschirm befindet, gibt es zunächst keine Möglichkeit, in den Zustand 2 (Dragging) zu gelangen. Hierfür werden zusätzliche Eingabevorrichtungen gebraucht, beispielsweise ein druckempfindlicher Bildschirm, der durch stärkeren Druck in den Zustand 2 wechseln könnte oder ein Stift mit einer Taste (siehe Abbildung 17.4 Unten). Da viele Sensoren, beispielsweise die kapazitiven Sensoren heutiger Smartphones, diese Möglichkeit nicht bieten, muss man sich mit anderen Tricks behelfen, um die bekannten Operationen wie **Zeigen** (engl. **pointing**) und **Auswählen** (engl. **selection**) zu unterstützen.

Ein solcher häufig angewendeter Trick ist die sogenannte **Lift-Off-Strategie**. Sie funktioniert für grafische Schnittstellen, in denen keine Positionsänderung in Zustand 2 notwendig ist und arbeitet wie folgt: Zustand 1 ist wie oben beschrieben der Tracking

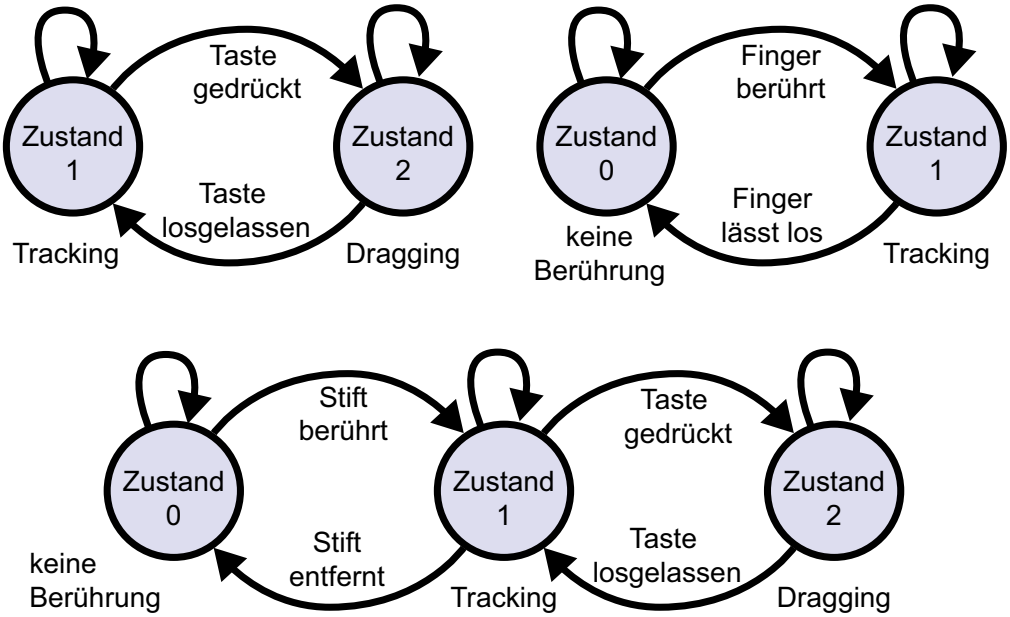


Abbildung 17.4: Bill Buxtons Modell der 3 Zustände für Maus (Oben links), Single Touch (Oben rechts) und einen Stift mit Taste (Unten), in Anlehnung an Buxton [18]

mmibuch.de/a/17.4

(oder *hover*) Zustand. Verlässt der Finger den Sensor über einem grafischen Objekt (Wechsel in den Zustand 0), so wird dieses Objekts ausgewählt. Dieser Strategie folgen die allermeisten heutigen grafischen Schnittstellen auf Touchscreens, beispielsweise die Schnittstellen der gängigen Smartphone Betriebssysteme iOS und Android. Berührt man beispielsweise deren Bildschirm über dem Icon der falschen App, dann kann man den Finger immer noch zum richtigen Icon bewegen und dort loslassen und hat damit keine falsche Eingabe getätigt.

17.1.3 Das Midas Touch Problem

Die eben analysierte Situation, dass bei der Touch Eingabe mit handelsüblichen Sensoren nur zwei Zustände unterschieden werden können, führt zu dem sogenannten **Midas Touch Problem**. Dieser Name geht zurück auf die Sage des Königs Midas von Phrygien. Ihr zufolge wünschte sich Midas vom Gott Dionysos, dass alles was er anfasste, unmittelbar zu Gold würde. Als ihm der Wunsch erfüllt wurde, stellte er die unerwünschten Nebeneffekte fest, beispielsweise, dass auch sein Essen und Getränke in seinen Händen direkt zu Gold und damit ungenießbar wurden. Genau so unerwünscht ist es in vielen Situationen, dass beim Berühren eines Touch Sensors das an dieser Stelle befindliche Objekt sofort selektiert wird. In einfachen Fällen (beispielsweise Aus-

wahl einfacher Funktionen mittels Sensortasten) mag dies tolerierbar sein, in komplexeren Benutzerschnittstellen möchte man jedoch oft in einen dritten Zustand gelangen oder zumindest eine fehlerhafte Selektion korrigieren können. Zur Umgehung des Midas Touch Problems gibt es verschiedene Ansätze.

Der gängigste Ansatz, der ohne zusätzliche Eingabevorrichtungen auskommt, ist der **Verweildauer** (engl. **dwell time**) Ansatz. Dabei ist eine bestimmte Verweildauer mit dem Finger über einem Objekt nötig, um es zu selektieren. Bewegt man den Finger vorher weg, so wird das Objekt nicht selektiert. Die Korrektur einer fehlerhaften Selektion ist somit innerhalb der nötigen Verweildauer noch möglich. Nach Ablauf der Verweildauer wird entweder das fragliche Objekt selektiert oder es erfolgt ein Moduswechsel, beispielsweise um das Objekt zu verschieben. Dieser Ansatz löst das Problem zwar ohne zusätzliche Hardware, benötigt aber für jeden Interaktionsschritt die Zeit der Verweildauer, wodurch die Interaktionsgeschwindigkeit grundlegend beschränkt wird. Ein weiterer Ansatz ist die oben beschriebene Lift-off Strategie, die sich zwar zur Selektion, nicht aber zum Verschieben eignet. Eine vollständige Lösung des Problems lässt sich nur durch eine weitere Eingabemöglichkeit erreichen, beispielsweise eine Taste, ein Pedal, oder einen anderen Sensor.

Das Midas Touch Problem existiert übrigens auch in anderen Eingabemodalitäten: Für querschnittsgelähmte Menschen, die nur noch die Augen bewegen können, kann man Beispielsweise eine Interaktionsmöglichkeit schaffen, indem man die Blickrichtung ihrer Augen mittels eines **Eyetrackers** verfolgt. Zeigt man nun auf einem Bildschirm eine Tastatur mit hinreichend großen Tasten an, dann kann eine solche Person Texte eingeben, indem sie die zugehörigen Tasten nacheinander anschaut. Zur Selektion der Taste wird hier normalerweise die **dwell time** Strategie verwendet, was zwar die Geschwindigkeit beschränkt, dafür die Eingabe aber grundlegend erst ermöglicht. Auch die anderen genannten Möglichkeiten lassen sich anwenden: als zusätzlicher Sensor zur Eingabe kann beispielsweise die Erkennung des Lidschlags genutzt werden.

17.1.4 Das Fat Finger Problem

Ein typischer Touch Bildschirm eines aktuellen Smartphones hat eine Breite von etwa 6cm. Der menschliche Zeigefinger ist an seiner Spitze etwa 2cm breit. Berührt man mit ihm den Bildschirm, so lässt sich nicht mit großer Genauigkeit sagen, an welcher Position der zugehörige Berührungspunkt erkannt wird, der ja konzeptuell keine Breite besitzt. Wir können nur davon ausgehen, dass er irgendwo unter der Fingerspitze sein muss, beispielsweise in deren Mitte. Berücksichtigt man diese Unsicherheit, dann lassen sich auf der Bildschirmbreite eigentlich nur drei bis vier verschiedene interaktive Elemente (z.B. Buttons) nebeneinander anordnen. Vertikal gilt grob gesehen das Gleiche. Diese Problematik, dass der Finger im Verhältnis zum Touch Bildschirm relativ groß ist, bezeichnet man in der englischsprachigen wissenschaftlichen Literatur als **Fat Finger Problem**. Ihm wird beispielsweise dadurch Rechnung getragen, dass das Startmenü der gängigen Smartphone Betriebssysteme iOS und Android nicht mehr als ca. 4 App-Icons nebeneinander anordnet.

Ein damit verwandtes Problem ist das der Verdeckung: Bei der Interaktion mit grafisch detaillierten Inhalten, wie z.B. Kartenmaterial oder Texten verdeckt der Finger immer

genau den aktuell selektierten Teil. Dem kann man beispielsweise entgegenwirken, indem man den verdeckten Teil vergrößert neben dem Finger nochmals anzeigt [118], wie dies auch auf aktuellen Smartphones bei der Texteingabe implementiert ist. Ein anderer Ansatz besteht darin, den Finger nicht von vorne auf den zu steuernden Bildschirm zu legen, sondern beispielsweise auf der Rückseite des Gerätes [7].

17.1.5 Interaktionskonzepte für Touch

Interaktionskonzepte für Touch Eingabe sind ein aktuelles Forschungsgebiet, in dem recht viel Bewegung ist. Da die anfänglichen Touch-Sensoren (vgl. Abschnitt 17.1.1) lediglich die Erkennung eines einzelnen oder mehrerer Berührungspunkte erlaubten, verwendeten die zugehörigen Interaktionskonzepte auch nur solche Punkte, ähnlich der Zeigeposition eines Mauszeigers. Auf diesem Konzept eines einzelnen Zeigers (*single pointer*) lassen sich bereits vielfältige Interaktionstechniken aufbauen, wie beispielsweise die verschiedenen Menütechniken aus Kapitel 15.

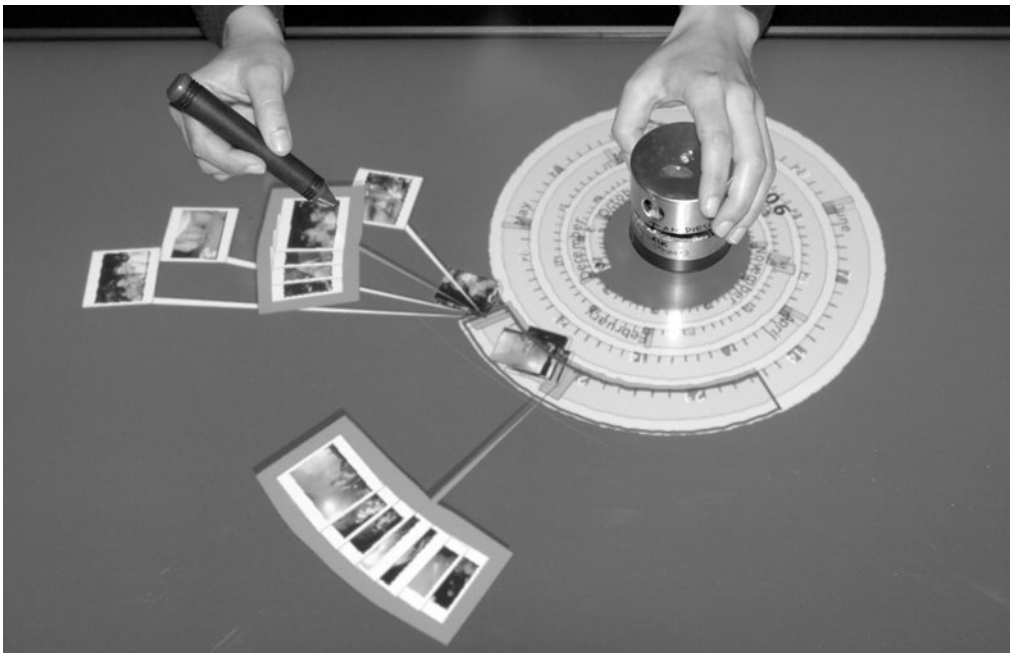
Mit der Erkennung mehrerer Kontaktpunkte wurden zusätzliche Techniken möglich, wie beispielsweise die mittlerweile allgegenwärtige **Pinch** Geste zum Zoomen grafischer Darstellungen oder Mehrfinger-Gesten auf Touch-Sensoren, die beispielsweise zum Blättern oder Scrollen verwendet werden. Bei großen interaktiven Oberflächen können mehrere erkannte Kontaktpunkte auch zu verschiedenen Händen oder verschiedenen Benutzern gehören, was neue Probleme, aber auch neue Möglichkeiten schafft (Abschnitt 17.2).

Geht man schließlich über einzelne Kontaktpunkte hinaus und betrachtet komplexere Kontaktflächen zwischen Hand, Arm, oder Objekten und der interaktiven Oberfläche, dann sind die darauf aufbauenden Interaktionskonzepte weitgehend unbekannt bzw. noch zu erforschen. Aktuelle Formalismen zur Beschreibung von Touch Eingabe, wie z.B. das **TUIO** Protokoll[2] ermöglichen es immerhin, solche Kontaktflächen durch ihren Umriss zu beschreiben. Ein anderer möglicher Ansatz [130] beschreibt die Kontaktflächen durch die von ihnen belegten Pixel, modelliert die Elemente der grafischen Benutzerschnittstelle als physikalische Objekte und simuliert sodann die physikalischen Auswirkungen der Berührungsfläche auf die Schnittstellenobjekte. Ein solches Eingabekonzept bietet die gleichen Vorteile bezüglich Erlernbarkeit und Bedienbarkeit wie die in Abschnitt 7.8 beschriebenen physikanalogen Interface-Konzepte.

17.2 Große Interaktive Oberflächen

Neben den weit verbreiteten Smartphones tauchen immer häufiger relativ große interaktive Oberflächen auf. Beispiele hierfür in unseren Alltagsumgebungen sind interaktive Schultafeln und Tische, in der Forschung auch ganze Wände oder Fußböden [4]. Auf solch großen Flächen lassen sich neue Effekte bei der Interaktion beobachten. Die mit der Größe der Finger verbundenen Probleme der Verdeckung oder mangelnden Auflösung (siehe Abschnitt 17.1.4) treten dabei immer mehr in den Hintergrund und Probleme der Koordination beider Hände und mehrerer Benutzern werden bedeutsamer.

[2] http://tuio.org



mmibuch.de/a/17.5

Abbildung 17.5: Photohelix: Ein gemischt physikalisches und grafisches Interface mit unterschiedlichen Rollen für beide Hände [47]

17.2.1 Beidhändige Interaktion

In Abschnitt 4.3 wurde Guiards Modell der beidhändigen Interaktion vorgestellt. Die darin beschriebene Rollenverteilung hat direkte Auswirkungen auf die Gestaltung von Interaktionstechniken für beide Hände auf großen interaktiven Oberflächen: Interaktionen, die einen höheren Anspruch an Präzision und Koordination stellen, gehören in die dominante (also meist die rechte) Hand, gröbere Aufgaben, die den Interaktionskontext oder -modus bestimmen, gehören in die nichtdominante (also meist die linke) Hand. Ein Beispiel für ein solches grafisches Interface zeigt Abbilddung 17.5. Das hier gezeigte Interface namens PhotoHelix [47] dient zum Durchsuchen einer Fotosammlung auf einem interaktiven Tisch: Die linke Hand bedient eine spiralförmig aufgewickelte Zeitleiste und dreht diese jeweils so, dass verschiedene Abschnitte davon unter ein Fenster gebracht werden, das im festen Winkel zur Hand verbleibt. Durch Drehen an dem physikalischen Drehknopf kann sich der Benutzer somit entlang der Zeitleiste bewegen. Auf dem Tisch erscheint dabei zu jedem Ereignis, das im Zeitfenster auftaucht, ein aufgefächerter Bilderstapel. Aus diesem können mit der rechten Hand einzelne Bilder herausgezogen und neu gruppiert werden. Die rechte Hand kann darüber hinaus Bilder aus der gesamten Anordnung heraus auf den Tisch bewegen, skalieren und rotieren. Die Linke stellt also

den Kontext ein (Zeitfenster), während die Rechte feingranulare Aufgaben wie Manipulation und Koordination übernimmt. Durch die physikalischen Werkzeuge Drehknopf und Stift und deren **Affordances** wird diese Rollenverteilung zudem unterstützt: Der Drehknopf bietet sich lediglich zum Drehen an, sowie zum Bewegen des gesamten Interfaces an eine andere Stelle des Tisches. Der Stift mit seiner feinen Spitze vermittelt die Möglichkeit, auch genauere Selektionen und Manipulationen vorzunehmen.

17.2.2 Mehrere Benutzer

Bei der gleichzeitigen Interaktion mehrerer Benutzer tritt zunächst einmal das **Identifikationsproblem** auf. Der Touch Sensor kann in der Regel nicht unterscheiden, wessen Hand oder Finger eine bestimmte Operation ausgeführt hat. Für viele nichttriviale Interaktionen zwischen mehreren Benutzern ist dies aber durchaus wichtig, um beispielsweise Zugriffsrechte auf digitale Objekte zu ermöglichen. In bestimmten Sonderfällen lässt sich dieses Problem entschärfen, indem man gewisse *Common-Sense*-Annahmen macht: Stehen sich beispielsweise an einem relativ großen interaktiven Tisch zwei Benutzer gegenüber, dann wird jeder bevorzugt auf seiner Seite des Tisches interagieren, und mit der simplen Annahme, dass alle Interaktionen rechts der Tischmitte zu Benutzer A, und links davon zu B gehören, sind bereits die meisten Fälle korrekt abgedeckt. Diese Heuristik lässt sich einfach auf vier Nutzer an einem viereckigen Tisch erweitern.

Gerade die Interaktion an Tischen wirft aber ein anderes Problem auf, nämlich das **Orientierungsproblem**. Sobald die dargestellten grafischen Elemente richtungssensitiv sind, also z.B. auf dem Kopf stehend nicht mehr erkannt oder gelesen werden können, muss ihre Orientierung angepasst werden. Die einfachste Art, dies automatisch zu tun, verwendet die gleiche Heuristik wie bei der Identifikation beschrieben. Jedes Objekt wird so ausgerichtet, dass es von der nächstgelegenen Tischkante aus lesbar ist. Dahinter steht die Annahme, dass es am ehesten für denjenigen Nutzer wichtig ist, zu dem es am nächsten liegt. Eine andere Vorgehensweise ist, die Orientierung dem Benutzer freizustellen. Indem wir ein Objekt für einen anderen Benutzer passend hin drehen, kommunizieren wir nämlich auch eine Art Übergabe des Objektes an den anderen. Die Orientierung von Objekten erfüllt damit auch kommunikative und koordinative Aufgaben [68].

17.2.3 Raumaufteilung

Sobald sich mehrere Nutzer eine interaktive Oberfläche teilen, teilen sie sich auch deren Raum untereinander auf. Dabei ist der eigene Aktionsradius zunächst einmal durch die pure Erreichbarkeit, also z.B. die Länge der Arme bestimmt. In der Regel überschneiden sich aber die erreichbaren Bereiche, so dass andere Konventionen zum Tragen kommen. Eine Studie [99] über das gemeinsame Lösen von Problemen, die Koordination benötigen (im Beispiel das Lösen von Puzzle-Aufgaben) zeigte dabei, dass jeder Benutzer für sich einen persönlichen Interaktionsbereich definiert, der in der Regel zentral vor dem eigenen Körper liegt. Darüber hinaus gibt es einen weniger stark genutzen Ablage-Bereich direkt um den Interaktionsbereich herum, sowie einen gemeinsam genutzten Austausch-Bereich in der Mitte des Tisches. Bei kurzem Nachdenken entspricht dies auch genau der Raumaufteilung bei einer andern wichtigen gemeinsamen Tätigkeit am

Tisch, nämlich dem Essen: Die gemeinsam genutzten Schüsseln und Platten befinden sich in der Tischmitte, wo sie von jedem erreichbar sind. Das persönliche Essen wird auf dem Teller direkt vor dem Körper zerteilt und auf Löffel oder Gabel geladen, und rund um den Teller ist Platz für weiteres Besteck, Trinkgläser und Serviette. Eine solche Aufteilung in der Benutzerschnittstelle unterstützt also unsere im Alltag erlernten Gewohnheiten.

Übungsaufgaben:

1. Warum werden in den grafischen Benutzerschnittstellen heutiger Smartphones selten verschiebbare Icons wie in Desktop-Schnittstellen verwendet? Argumentieren Sie mithilfe des 3-Zustände-Modells.
2. Finden Sie ein Gegenbeispiel (verschiebbare Icons auf einem Smartphone oder Tablet) und analysieren sie, wie dies umgesetzt ist, um den Einschränkungen des 3-Zustände-Modell zu entkommen.
3. Erläutern Sie warum das Fat-Finger Problem bei Touchscreens und nicht bei Touchpads auftritt. Wie kann man diese Erkenntnisse verwenden, um bei Touchscreens das Fat-Finger Problem zu entschärfen? Diskutieren Sie in diesem Sinne existierenden Lösungen zur Interaktion mit Smartphones.

18 Mobile Interaktion

Das **Smartphone** hat in den letzten Jahren eine rasante Verbreitung erfahren - seit der Einführung des iPhone im Jahre 2007 ist diese Geräteklasse aus vielen Haushalten nicht mehr wegzudenken. Seine weltweite Bedeutung wird weiter zunehmen, denn auch in Entwicklungs- und Schwellenländern gewinnt das Smartphone zunehmend an Bedeutung [38] und hilft dabei, ganze Technologiegenerationen zu überspringen, da die Anforderungen an die Infrastruktur, wie z.B. ein funktionierendes Stromnetz und eine kabelbasierte Kommunikationsinfrastruktur bei Smartphones teilweise entfallen.

Das Smartphone wird aufgrund seiner einfachen Verfügbarkeit (man hat es immer dabei) und auch aufgrund seines günstigeren Preises im Vergleich zu Desktop- oder Notebook-PCs für immer mehr Benutzer der wichtigste Computer im Alltagsleben. Das Smartphone kennzeichnet allerdings nur den Beginn einer technologischen Entwicklung, die zu immer leichteren und tragbareren Geräten führt, die der Benutzer immer bei sich trägt und auf deren Dienste schnell und einfach zugegriffen werden kann, wie z.B. Datenbrillen oder intelligente Uhren oder Armbänder. Besondere Bedeutung hat in diesem Zusammenhang das Forschungsfeld **Wearable Computing**. Auch wenn wir uns in diesem Kapitel auf das Smartphone als wichtigste mobile Plattform konzentrieren, betreffen viele der besprochenen Aspekte auch die Interaktion mit anderen tragbaren Geräte.

Einige Besonderheiten der mobilen Interaktion unterscheiden diese maßgeblich von klassischen Desktop-PC-Systemen (siehe Kapitel 15). Ein offensichtlicher Unterschied ist die Mobilität: Mobile Geräte sind immer dabei, passen in die Hosentasche, Informationen können schnell und einfach abgerufen werden. Mobile Geräte werden dadurch häufiger für kürzere Zeitspannen verwendet als Desktop-Systeme, so dass wir häufiger bei anderen Aufgaben unterbrochen werden und diese wieder aufnehmen müssen. Mobile Interaktion muss desweiteren an den Kontext angepasst werden. Ein bekanntes Beispiel ist die Interaktion mit **ortsbasierten Diensten**, wie z.B. einem Navigationsdienst. Neben dem Ortskontext spielen aber noch weitere Kontextarten, wie z.B. der soziale oder technische Kontext eine große Rolle. Eine große Herausforderung für die Interaktion bedeutet der in der Regel kleine Bildschirm und das Fehlen einer angemessenen Tastatur. Schließlich verfügt ein mobiles Gerät heute auch über eine Vielzahl von Sensoren, die zur Interaktion genutzt werden können.

18.1 Unterbrechbarkeit

Wie in Abschnitt 3.4 schon diskutiert ist der Mensch in der Lage, bei vielen Aufgaben seine Aufmerksamkeit zu teilen und damit mehre Aufgaben parallel zu bearbeiten. Das Beispiel aus Abschnitt 3.4 beschrieb die geteilte Aufmerksamkeit des Autofahrers, der zwischen der Aufgabe der eigentlichen Fahrzeugführung und anderen wie z.B. der

mmibuch.de/a/18.1

Abbildung 18.1: Die Android Notification Bar wird zum Sammeln und Anzeigen von Notifikationen benutzt, um so die Unterbrechung während der Verwendung von anderen Applikationen des mobilen Gerätes zu reduzieren.

Bedienung des Navigationsgerätes oder des Radios hin und her wechselt. Eine andere Sichtweise auf diese geteilte Aufmerksamkeit ist die der Unterbrechung von Aufgaben, die insbesondere bei der mobilen Interaktion eine sehr große Rolle spielt. Mobile Geräte verfügen über eine Vielzahl von Funktionen und Applikationen, die potenziell für eine Notifikation des Benutzers verantwortlich sein können und damit auch den Benutzer bei einer anderen Tätigkeit unterbrechen können. Desweiteren besitzt bei der mobilen Interaktion der Interaktionsontext (siehe Abschnitt 18.2) einen viel größeren Einfluss als bei einer vergleichbaren Interaktion mit einem Desktop-PC.

Eingehende Telefonanrufe, Erinnerungen an Termine im elektronischen Kalender, E-Mail-Notifikationen, SMS- und Messenger-Benachrichtigungen versuchen durch visuelle, akustische und taktile Signale auf sich aufmerksam zu machen ohne zu berücksichtigen in welcher Situation und in welchem Kontext der Benutzer sich befindet. Während dies bei der Desktop-Interaktion im schlimmsten Fall lästig ist, kann es bei der mobilen Interaktion zu sozial peinlichen Situationen (Telefon klingelt im Kino) oder sogar zu gefährlichen Situationen führen, wenn eine Unterbrechung eine benötigte motorische Fertigkeit stört (z.B. wenn eine SMS-Benachrichtigung beim schnellen Treppensteigen eintrifft und der Benutzer im Lauf stolpert). Umso wichtiger ist es beim Entwurf mobiler Benutzerschnittstellen, die Unterbrechbarkeit für den Benutzer kontrollierbar zu gestaltet. Dazu gehört die Kontrolle über die Modalität der Unterbrechung (z.B. akustisch oder visuell) aber auch die Art und Weise, wie Notifikationen durch die Benutzerschnittstelle präsentiert werden. Das Android-Betriebssystem verwendet zu diesem Zweck die in Abbildung 18.1 gezeigte **Notification Bar**, in der Benachrichtigungen angezeigt und gesammelt werden. Diese Ansicht kann dann vom Benutzer durch Interaktion erweitert werden, um auf Details der unterschiedlichen Benachrichtigungen zugreifen zu können.

Formal können die Kosten, die mit der Unterbrechung einer Aufgabe einhergehen, durch den zeitlichen Mehraufwand definiert werden, der durch die Unterbrechung verursacht wird. Wir bezeichnen die Zeit T_a als die Zeitspanne, die ein Benutzer zur Bearbeitung einer Aufgabe ohne Unterbrechung benötigt. Wird der Benutzer bei der Aufgabe beispielsweise durch einen eingehenden Anruf unterbrochen, die Aufgabe also unterbrochen und nach dem Anruf wieder aufgenommen, teilt sich die Zeitspanne T_a in zwei Zeitspannen auf. Dabei bezeichnet die Zeitspanne T_v die Zeit, die *vor* der Unterbrechung auf die Aufgabe verwendet wurde und die Zeitspanne T_n, die Zeit, die *nach* der Unterbrechung zur Vollendung der Aufgabe benötigt wird. Hinzu kommt noch die Zeitspanne T_u, welche die Zeit der Unterbrechung kennzeichnet. Diese Spanne kann von sehr kurz (z.B. eine SMS-Notifikation) bis sehr lang (z.B. einem Telefonat) reichen.

Die neue Gesamtzeit, inklusive Unterbrechung T_g berechnet sich aus der Summe der drei Zeiten: $T_g = T_v + T_u + T_n$. Hätte die Unterbrechung keinerlei (zeitliche) Kosten, so müsste gelten: $T_a = T_v + T_n$, d.h. die Unterbrechung hat keinerlei Verlängerung der Bearbeitungszeit zur Folge. In der Regel ist jedoch $T_a < T_v + T_n$, d.h. die Unterbrechung verursacht einen zusätzlichen zeitlichen Aufwand T_o (insbesondere gilt: $T_a = T_v + T_n + T_o$), der sich in der reinen Bearbeitungszeit der Aufgabe widerspiegelt. Größere Studien mit Computerbenutzern am Desktop-PC haben gezeigt, dass dieser zusätzliche Aufwand T_o erheblich sein kann. Benutzer haben bei gängigen Office-Anwendungen im Durchschnitt über 16 Minuten mehr benötigt, wenn sie bei einer Aufgabe unterbrochen wurden [54]. Studien zur Unterbrechung bei der Verwendung mobiler Geräte kommen grundsätzlich zu ganz ähnlichen Schlüssen [13].

Zur Reduktion der Negativeffekte einer Unterbrechung, also zur Verringerung von T_o lassen sich grundsätzlich zwei unterschiedliche Strategien verfolgen: die **präventive Strategie** und die **kurative Strategie**. Die *präventive* Strategie zielt darauf ab, die negativen Effekte der Unterbrechung zu reduzieren indem das System die Unterbrechung abhängig vom Benutzerkontext (siehe auch den nächsten Abschnitt 18.2) kontrolliert durchführt. Am Beispiel eines unterbrechenden Telefonanrufes kann dies bedeuten, dass die Unterbrechung durch das System verzögert wird bis der Benutzer die Bearbeitung der Aufgabe (zumindest teilweise) zu Ende geführt hat.

Die *kurative* Strategie setzt bei der Wiederaufnahme der Aufgabe nach der Unterbrechung an. Sie zielt darauf ab, den Benutzer möglichst schnell wieder in die Lage zu versetzen, die ursprüngliche Aufgabe zu Ende zu führen. Eine solche kurative Strategie haben zum Beispiel Kern et al. für Navigationssysteme im Auto entwickelt [58]. Die Strategie *gazemarks* verwendet den letzten Blickpunkt des Benutzers auf der elektronischen Karte des Navigationsgeräts bevor dieser wieder auf die Straße blickt. Dieser letzte Punkt wird dann durch eine Visualisierung hervorgehoben. Die Kosten der Unterbrechung durch Ereignisse im Straßenverkehr für die Aufgabe, Navigationshinweise auf der elektronischen Karte zu verstehen, wurden so nachweislich reduziert.

18.2 Interaktionskontext

Die Interaktion am Desktop-PC findet in der Regel in einer wohldefinierten Umgebung statt: Die Lichtbedingungen sind nahezu konstant, die Position des Benutzers ist mehr oder weniger unveränderlich (meistens sitzend), die Interaktionsgeräte sind überschaubar und wohl bekannt (Maus und Tastatur) und der Benutzer arbeitet meistens alleine. Dieser starre Interaktionskontext hat Vorteile. Er erlaubt es, eine Vielzahl von Annahmen in den Designprozess zu integrieren, z.B. wie fokussiert ein Benutzer eine Applikation bedienen kann, wie hoch die Bildschirmauflösung ist und welche ergonomischen Rahmenbedingungen während der Interaktion gelten (z.B. bezüglich der Lage der Maus auf dem Tisch und der Bewegungsfreiheit des Arms).

Im Fall der mobilen Interaktion gelten viele dieser Annahmen nicht. Der mobile **Interaktionskontext** ist vielschichtiger und weniger vorhersagbar. Einerseits macht dies den Designprozess komplexer und in vielen Aspekten verschieden vom Desktop-PC Designprozess. Andererseits ergeben sich durch die Berücksichtigung des Kontexts völlig

neue Möglichkeiten in der mobilen Interaktion, die am Desktop-PC nicht zu realisieren sind. Die Idee, vielschichtige Faktoren in der Umwelt des Benutzers in die Interaktion miteinzubeziehen, ist eng verknüpft mit dem Begriff des **Ubiquitous Computing** (dt. allgegenwärtiges Rechnen). Mark Weiser bezeichnete mit diesem Begriff Ende der 1980er Jahre die dritte Ära von Rechnern und deren Interaktionsprinzipien, die nach der ersten Ära der Zentralrechner und der zweiten Ära des Desktop-PC nun bevorstehe [125].

Das Ubquitous Computing ist geprägt durch die weitere Miniaturisierung von Computern und deren Verteilung und Einbettung in unserer Umgebung. Dabei werden mobile Computer ausdrücklich miteinbezogen. Schon heute bewegen wir uns in Umgebungen, die hochinstrumentiert und mit Computer durchdrungen sind. Ein typisches Beispiel sind moderne Fahrzeuge der Luxusklasse, in denen sich mehr als 100 Prozessoren für unterschiedlichste Aufgaben befinden. Mobile Computer ergänzen diese Umgebungen bereits heute, wenn z.B. das Smartphone sich mit dem Auto verbindet, um Musik über die Autostereoanlage abzuspielen oder Daten- und Sprachdienste im Auto zur Verfügung zu stellen. Aus Sicht der MMI kennzeichnet das Ubiquitous Computing einen Meilenstein und Paradigmenwechsel in der Art und Weise, wie interaktive System entwickelt werden müssen. Dies wird insbesondere durch das Verhältnis zwischen der Anzahl der Benutzer und der Anzahl der verwendeten Geräte deutlich. Während der Zentralrechner von vielen Benutzern verwendet wird (Verhältnis n:1), der Desktop-PC von genau einem Benutzer (1:1), kehrt sich dieses Verhältnis beim Ubiquitous Computing weiter um: ein Benutzer verwendet viele Computer (1:n). Die damit einhergehende Orchestrierung verschiedenster technologischer Komponenten ist eine große Herausforderung an die MMI und weiterhin Gegenstand intensivster Forschung.

Offensichtlich spielt der Kontext der Verwendung von Computern im Ubiquitous Computing und damit auch in der mobilen Interaktion eine große Rolle. Interaktive mobile Systeme, die den Kontext ignorieren, können nicht vernünftig funktionieren. Soll der Kontext in der Interaktion berücksichtigt werden, stellt sich zunächst die Frage, was in diesem Zusammenhang unter Kontext verstanden werden soll, und wie dieser für die mobile Interaktion formalisiert werden kann. Zwei wegweisende Projekte zur Untersuchung der Rolle des Kontexts auf die mobile Interaktion waren die Anfang der 1990er Jahre durchgeführten Projekte *Active Badge* [120] und *Parctab* [121]. In beiden Systemen wurde der Kontext des Benutzers während der Interaktion des Systems berücksichtigt und floss in die Interaktion mit ein. In beiden Projekten wurde Kontext vor allem durch die drei *Ws* charakterisiert: *Wo* ist der Benutzer? *Wer* befindet sich in der Nähe des Benutzers? *Welche* Dienste kann der Benutzer nutzen? Wichtig ist dabei festzuhalten, dass diese Fragen zu unterschiedlichen Zeitpunkten unterschiedlich beantwortet werden. Nichtdestotrotz stellt sich immer die Frage, welcher Kontext tatsächlich für eine bestimmte mobile Anwendung *relevant* ist, da ja nicht alle möglichen Antworten zu den obigen Fragen zu jeder Zeit in einem mobilen System berücksichtigt werden können. Der Forscher **Anind Dey** schlägt daher folgende Definition von **Kontext** vor:

> Kontext bezeichnet jegliche Art von Information, die verwendet werden kann, um die Situation einer Entität zu charakterisieren. Unter einer Entität verstehen wir eine Person, einen Ort oder Objekte die für die Interaktion zwischen Benutzern und Anwendungen relevant sind. Benutzer und Anwendungen sind damit auch Teil des Kontexts.

Unter Berücksichtigung der drei *W*-Fragen und der obigen Definition kann man grundsätzlich drei Kontextarten unterscheiden: den **physischen Kontext**, den **sozialen Kontext** und den **diensteorientierten Kontext**. Diese drei Kontextarten sind nicht unabhängig voneinander, sondern bedingen sich teilweise gegenseitig.

18.2.1 Physischer Kontext

Die wichtigste Eigenschaft des **physischen Kontexts** ist der geografische Ort des Benutzers. Ihn zu kennen ist der Schlüssel zu weiterer Information, die im Rahmen der mobilen Interaktion nützlich sein kann. Mobile Navigationssysteme benötigen offensichtlich den Ort des Benutzers, um sinnvolle Weghinweise geben zu können (siehe dazu auch den Abschnitt 18.2.4). Weitere physische Eigenschaften sind Teil dieser Kontextart, dazu gehören beispielsweise die Helligkeit/Lichtintensität in der Umgebung, die Lautstärke und die Art der Hinter- und Vordergrundgeräusche, die Temperatur am Ort, die Tages- und Jahreszeit, die momentane Geschwindigkeit und Beschleunigung des Benutzers, seine Orientierung im Raum, sowie das Transportmittel, das er benutzt (z.B. zu Fuß oder öffentliche Verkehrsmittel). Viele dieser Eigenschaften lassen sich aus dem Muster der zeitlichen Veränderung der Position ableiten [75]. Desweiteren sind veränderliche technische Rahmenbedingungen ebenfalls Teil des physischen Kontexts, beispielsweise die Verfügbarkeit drahtloser Netzwerke oder die Bandbreite der verfügbaren Datenverbindung, genauso wie weitere Geräte, auf die der Benutzer zugreifen kann.

18.2.2 Sozialer Kontext

Der **soziale Kontext** beschreibt, wer sich in der Nähe des Benutzers befindet. Dies beinhaltet einerseits die Ortsinformation der Personen in der Umgebung, andererseits ihren sozialen Stellenwert für den Benutzer, z.B. ob es sich um Freunde oder Familienmitglieder handelt oder ob Arbeits- oder Studienkollegen in der Nähe sind. Soziale Netzwerke wie z.B. Facebook bieten die Möglichkeit an, sich befreundete Nutzer in der unmittelbaren Umgebung anzeigen zu lassen, um sie gegebenenfalls gezielter kontaktieren zu können. In anderen Situationen kann die Auswertung des sozialen Kontexts das genaue Gegenteil bezwecken: Bei der mobilen Navigation kann eine erkannte Ansammlung von vielen Menschen und damit auch Autos auf der Autobahn zu einer Vermeidungsstrategie führen, wodurch ein Stau erfolgreich umfahren wird. Beim Wandern in der Wildnis kann beides erwünscht sein, abhängig vom sozialen Kontext: entweder das Auffinden von Gleichgesinnten zum gemeinsamen Erleben einer schöner Wanderung, oder die Vermeidung von Gleichgesinnten, um die Natur alleine erleben zu können. In diesem Sinne kann die Kenntnis über den sozialen Kontext einen wichtige Rolle bei der Kontrolle der eigenen Privatsphäre sein. Bei mobilen Spielen mit mehreren Benutzern spielt der soziale Kontext eine sehr große Rolle, insbesondere wenn in Teams gespielt wird. Ein originelles Beispiel ist das mobile Spiel *Human Pacman*, bei dem der Spieler in einer alltäglichen Umgebung in die Rolle der Spielfigur Pacman schlüpft und die Gegenspieler als Monster versuchen, ihn zu fangen [22].

18.2.3 Diensteorientierter Kontext

In Abhängigkeit vom physischen Kontext können zahlreiche Dienste für den Benutzer von Bedeutung sein. Die beiden oben erwähnten Projekte *Active Badge* und *Parctab* wurden in Büroumgebungen realisiert und so standen dort Bürodienste im Vordergrund, wie z.B. die Verwendung von Druckern im Bürogebäude, das automatische Weiterleiten von Telefonaten und die Erteilung von Zugangsberechtigungen. Darüber hinaus konnten Benutzer auf Webdienste zu greifen und Kalenderdienste nutzen, die abhängig von anderen Kontextfaktoren die Terminvereinbarung zwischen verschiedenen Benutzern vor Ort vereinfachten. Typische mobile Dienste sind das Finden von geeigneten öffentlichen Verkehrsmitteln und ihren Verbindungen, Übersetzungsdienste wie z.B. Dienste, die Schilder in der Umgebung mithilfe von Augmented Reality- Technologie übersetzen [77][1]. Die Anzahl der nutzbaren Dienste steigt immer stärker an. Die Appstores der weit verbreiteten Smartphone Betriebssysteme enthalten mehrere 100.000 Apps und viele von diesen Apps bieten kontextabhängige Dienste an. Zahlreiche Forschungsarbeiten untersuchen zur Zeit die Möglichkeit, Benutzern sinnvolle Apps abhängig von ihrem jeweiligen Kontext vorzuschlagen [12].

18.2.4 Kontextsensitivität

Interaktive mobile Systeme, die ihren Kontext im Sinne des vorherigen Abschnitts bei der Interaktion berücksichtigen, werden als **kontextsensitive Systeme** bezeichnet. In diesem Abschnitt wollen wir anhand des Beispiels mobiler Navigationssysteme diese **Kontextsensitivität** diskutieren. Das Beispiel der Navigation und wie diese technisch unterstützt werden kann wurde in diesem Buch schon häufiger verwendetet, da eine Reihe von Aspekten der MMI betroffen sind. Navigation ist eine typische kognitive Leistung des Menschen, die zunehmend durch Computer unterstützt wird und jeden betrifft. Es gibt viele unterschiedliche Ausprägungen von Navigationssystemen: man kann entweder auf spezielle Geräte zur Navigation zurückgreifen, Software für ein Smartphone oder den PC verwenden, oder aber ein fest in das eigene Auto integriertes Navigationssystem nutzen. Für diese Fälle gelten aus Perspektive der Mensch-Maschine Interaktion unterschiedliche Rahmenbedingungen, die zu berücksichtigen sind.

Im Folgenden wollen wir uns außerdem mit einer ganz speziellen Kategorie von Navigationssystemen beschäftigen, die erst in den letzten Jahren an Bedeutung gewonnen haben, den **Fußgängenavigationssystemen**. Bei diesen Systemen handelt es sich um eine Kombination von Hard- und Software, die mit dem Zweck entwickelt wurde, die Navigation unter der besonderen Berücksichtigung der Belange von Fußgängern zu unterstützen. Während Navigationssysteme für Automobile seit Beginn der 1980er Jahre kommerziell zur Verfügung stehen, sind Navigationssysteme für Fußgänger eine verhältnismäßig neue Erscheinung im kommerziellen Bereich. Die Forschung beschäftigt sich bereits seit 15-20 Jahren mit dem Entwurf von Fußgängernavigationsystemen und ihren besonderen Herausforderungen für die MMI. Abbildung 18.2 zeigt drei Beispiele von Visualisierungen für Fußgängernavigationssysteme verschiedener Generationen.

[1] Ein entsprechender Dienst wurde schon 1993 von **Wendy Mackay** in ihren Forschungsarbeiten beschrieben. Seit 2010 existieren entsprechende kommerzielle Dienste, wie z.B. der Dienst World Lens für das iPhone und für Android-Smartphones.

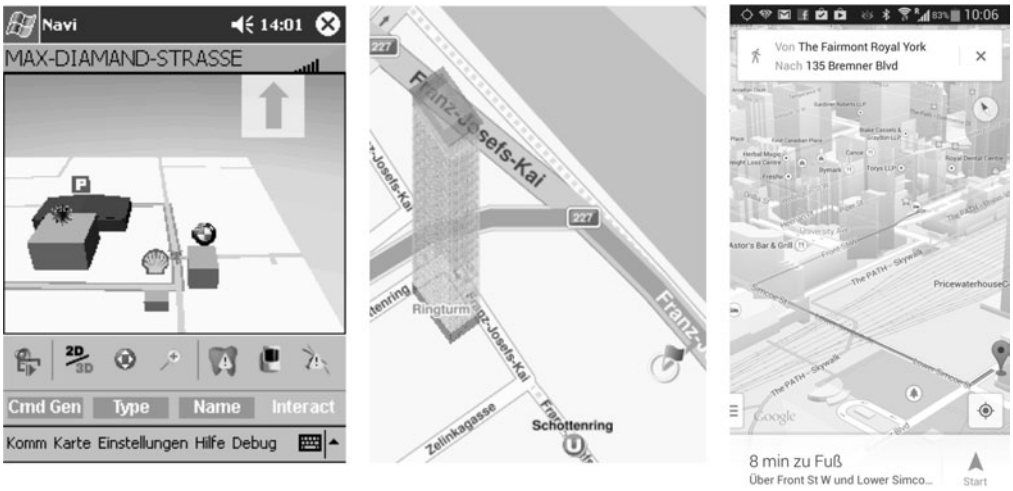


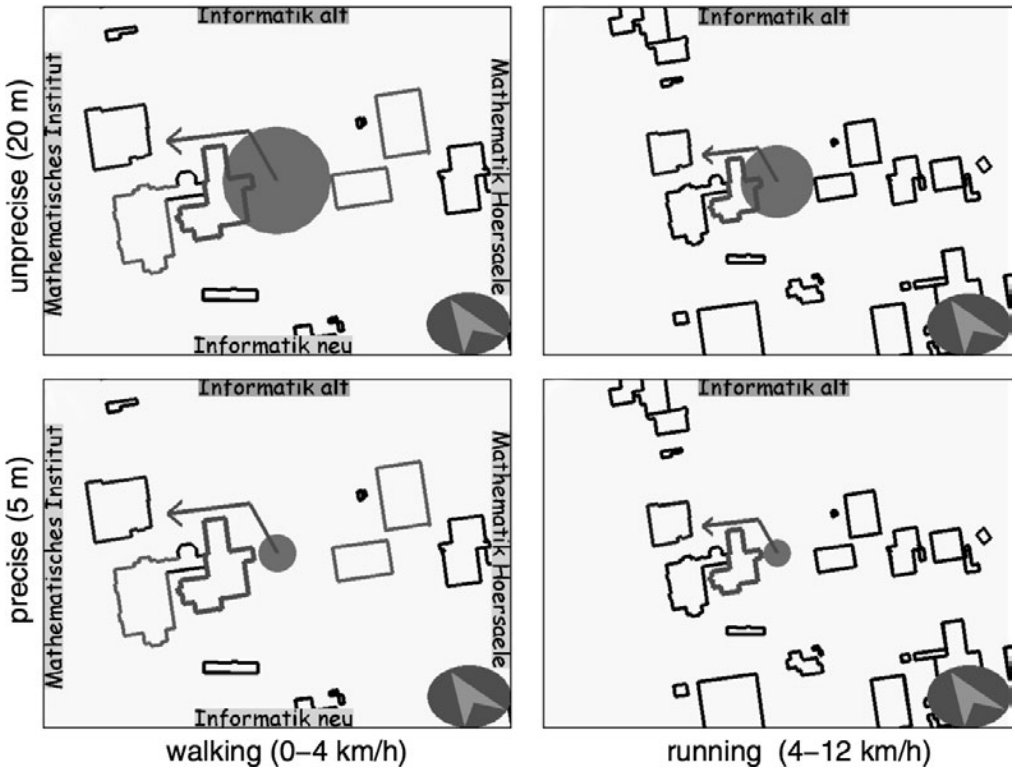
Abbildung 18.2: Beispiele für Kartenvisualisierungen, die für Fußgänger entwickelt wurden, Links: ein früher Forschungsprototyp von 2003 [122], Mitte: Nokia Maps, Version 4 von 2009, Rechts: Google Maps von 2014

mmibuch.de/a/18.2

Was auffällt, ist die frühe Einbettung von 3D-Grafiken, da diese gerade Fußgängern als Landmarken im urbanen Bereich sinnvolle Orientierungshilfe bieten. Im Gegensatz zur Autonavigation ist der Fußgänger deutlich langsamer unterwegs, hat also mehr Zeit, sich in der Umgebung zu orientieren.

Der Fußgänger kann seine Bewegungsgeschwindigkeit den örtlichen Begebenheiten besser anpassen. Er kann z.B. stehen bleiben, um sich zu orientieren. Dies ist während der Autonavigation prinzipiell auch möglich, aber mit mehr Kosten verbunden (um zu gewährleisten, dass der fliessende Verkehr nicht gestört wird). Der Fußgänger hat mehr Möglichkeiten sich zu orientieren, da er den Kopf während des Gehens freier bewegen kann und er ist nicht so starr festgelegt auf bestimmte Straßen. Im Gegensatz zum Autofahrer kann der Fußgänger z.B. kleine Wege zwischen Gebäuden und in Parks verwenden und darüber hinaus sich nicht nur im Freien, sondern auch in Gebäudekomplexen bewegen. Diese Tatsachen erhöhen die Anforderungen an die MMI eines Navigationssystems für Fußgänger erheblich und sollen Gegenstand der nächsten Abschnitte sein. Zunächst wird die Adaptivität im Mittelpunkt stehen, d.h. die Fähigkeit des Systems zur Anpassung und welche Konsequenzen sich daraus für die MMI ableiten. Anschließend werden verschiedene Modalitäten diskutiert, die für Fußgängernavigationssysteme zweckmäßig verwendet werden sollten. Schließlich werden wir hybride Navigationssysteme betrachten, d.h. Navigationssysteme, die unterschiedliche technische Komponenten verwenden, die im Sinne des Ubiquitous Computing in der Umgebung eingebettet sind.

Ein System, welches in der Lage ist, sich unterschiedlichsten Kontexten anzupassen, wird als **adaptives System** bezeichnet. Wie im vorherigen Abschnitt erwähnt müssen im



mmibuch.de/a/18.3

Abbildung 18.3: Beispiel für die grapische Adaptionsleistung des Fußgängernavigationssystems ARREAL aus dem Jahre 2000. Abhängig von der Positionierungsgenauigkeit und der Benutzergeschwindigkeit werden Benutzerposition und Umgebung unterschiedlich dargestellt.

Fall der Fußgängernavigation viele Kontextfaktoren berücksichtigt werden. Im Folgenden soll exemplarisch auf zwei Faktoren eingegangen werden: Benutzergeschwindigkeit und Benutzerumgebung. Üblicherweise wird bei Fußgängern von einer durchschnittlichen Geschwindigkeit von ca. $5\frac{km}{h}$ ausgegangen. In der Realität schwankt die Geschwindigkeit allerdings erheblich, wenn Fußgänger sich z.B. in großen Menschenmassen bewegen, Treppen steigen, stehen bleiben oder auch einmal schneller laufen. Dabei muss der Fußgänger seine unmittelbare Umgebung im Sinne der geteilten Aufmerksamkeit (siehe Abschnitt 3.4) im Blick behalten. Untersuchungen haben auch einen deutlichen Effekt des Alters des Fußgängers und der Beschaffenheit des Weges ergeben [61]. Ein adaptives Fußgängernavigationssystem muss sich diesen Sachverhalten anpassen und z.B. seine Visualisierungen entsprechend korrigieren. Im Forschungsprojekt ARREAL wurden Konzepte für Fußgängernavigationssysteme entwickelt, welche die Visualisierung von Weg- und Umgebungsinformation der Benutzergeschwindigkeit und der Genauig-

Abbildung 18.4: Multimodale Interaktion mit dem mobilen Fußgängernavigationssystem M3I. Links: die kombinierte Verwendung von Gesten auf dem touchsensitiven Bildschirm mit Spracheingaben, Mitte: die Verwendung von Gesten mit dem Gerät selbst, Rechts: ein Ausschnitt der Grammatik, die bei der Spracherkennung in Abhängigkeit vom Kartenausschnitt vorausgewählt wird und so die mobile Erkennungsleistung verbessert [122].

mmibuch.de/a/18.4

keit der Positionserkennung anpassen [9]. Abbildung 18.3 zeigt vier unterschiedliche Visualisierungen der gleichen Navigationssituation abhängig von unterschiedlichen Benutzergeschwindigkeiten (Gehen, bzw. Laufen) und der Genauigkeit, mit der das System die Positionierung des Benutzers vornehmen kann (im Bild 5 bzw. 20 Meter).

Im Gegensatz zur Situation des Autofahrens, wo die Positionierung durch GPS und zusätzliche Sensoren im Auto, wie z.B. Radlaufsensoren, verhältnismäßig robust gelingt, ist die Situation des Fußgängers häufig dadurch gekennzeichnet, dass er sich in Umgebungen befindet, die keinen zuverlässigen GPS-Empfang zulassen, z.B. in Häuserschluchten oder in Gebäuden. Führt dies zu einer deutlich erhöhten Ungenauigkeit bei der Positionierung, ist dies dem Benutzer von Seiten des Systems in geeigneter Art und Weise zu kommunizieren, damit Fehler in der Navigationsführung vermieden werden. Selbst beim vollständigen Ausfall der Positionierungstechnologie sollten solche Systeme in der Lage sein, Benutzern eine Selbstpositionierung zu ermöglichen, indem das System z.B. nach potenziell sichtbaren Landmarken fragt und so in einem Dialog dem Benutzer hilft, seine Position selbst zu bestimmen [65].

Fußgängernavigationssysteme werden sinnvollerweise **multimodal** konzipiert, da sie einer Reihe von Nebenbedingungen unterworfen sind. Diese greifen schon bei der Eingabe des Navigationsziels, was vorzugsweise per Spracheingabe geschehen sollte. Aber auch bei der Interaktion während der Navigation selbst, z.B. wenn der Benutzer weitergehende Informationen aus der elektronischen Karte ableiten möchte, stößt dieser schnell an die Grenzen der Benutzbarkeit, wenn nur eine Modalität eingesetzt wird (z.B. nur Sprache oder nur Gestik). Besser geeignet sind in diesem Fall multimodale Eingabekonzepte, die mehrere Modalitäten miteinander kombinieren. Das Fußgängernavigationssystem M3I (MultiModal Mobile Interaction) [122] erlaubt die Kombination

von Spracheingabe und Zeigegesten, die auf der elektronischen Karte ausgeführt werden (siehe Abbildung 18.4). Hierdurch wird einerseits die Erkennung wesentlich robuster, da zwei Eingabemodalitäten sich ergänzen, und andererseits natürlicher, da die kombinierte Verwendung von Gesten und Sprache in der menschlichen Kommunikation eine bedeutende Rolle spielen. Auch die Ausgabe sollte im besten Fall multimodal erfolgen. So gibt es Ansätze, Navigationshinweise haptisch über Vibrationen am Körper, z.B. durch einen Vibrationsgürtel [115] oder durch aktive Steuerung der Schuhform, die eine eindeutige Gehrichtung vorgeben [123] zu übermitteln. Im letzten Fall werden die *Affordances* (siehe Abschnitt 7.1) des Schuhs abhängig von der Gehrichtung manipuliert und führen zu einer schnellen Erfassung der neuen Gehrichtung durch den Benutzer.

Die Verwendung eines Smartphones bei der Fußgängernavigation hat auch einige Nachteile. Das Gerät muss aus der Tasche genommen werden, um damit zu interagieren. Der Bildschirm ist sehr klein und häufig im direkten Sonnenlicht, also schlecht ablesbar. Desweiteren möchte man eventuell die Reiseplanung selbst nicht auf dem mobilen Gerät durchführen und sich nicht auf ein Fortbewegungsmittel beschränken sondern kombiniert mit dem Auto, öffentlichen Verkehrsmitteln und schließlich zu Fuß auf die Reise gehen. Hier ist also ein Navigationssystem gefragt, welches verschiedene Bausteine besitzt und unterschiedliche Technologien für die Navigation verwendet. Ein System, welches aus unterschiedlichen Komponenten besteht, und diese sinnvoll miteinander ergänzt, wird **hybrides System** genannt. Die Fußgängernavigation kann in solchen Systemen eine Teilkomponente sein, wie z.B. in dem Forschungsprojekt BPN (BMW Personal Navigator) in dem die Vorbereitung der Navigation am Desktop-PC erfolgte und eine nahtlose Synchronisierung des Fußgängernavigationssystems und des Autonavigationssystems gewährleistet wurde [67].

Neuartige Ausgabemedien, wie z.B. Brillensysteme, intelligente Uhren und Armbänder werden mehr und mehr in die Fußgängernavigation miteingebunden und funktionieren häufig nur in Kombination mit weiteren Geräten, wie z.B. einem Smartphone. Elektronische Displays in der Umgebung können ebenfalls Teil eines hybriden Navigationssystems sein und in Kombination mit einer tragbaren Komponente schnelle und effektive Navigationsunterstützung leisten [89, 66].

18.3 Explizite vs. Implizite Interaktion

Die im vorherigen Abschnitt beschriebene Berücksichtigung des Kontexts in der mobilen Interaktion verändert die Art und Weise, wie wir mit dem mobilen Geräte interagieren, maßgeblich. Das klassische Interaktionsparadigma beruht darauf, dass ein Benutzer Eingabegeräte verwendet, um Befehle zu spezifizieren, die dann vom Computer ausgeführt werden (beim Desktop-PC im Wesentlichen durch die Verwendung von Maus und Tastatur). Diese Eingabe erfolgt in einer gewissen Art und Weise willentlich, da die Verwendung der Eingabegeräte gezielt zur Spezifizierung der Befehle an den Computer verwendet werden. Die Eingabe eines Kommandos oder die Auswahl eines Menüeintrags mithilfe des Mauszeigers geschieht alleine zur Kommandoausführung. Man spricht daher auch von **expliziter Interaktion**, da die Interaktion mit dem Computer unmittelbar und direkt mit einem expliziten Befehl des Benutzers in Verbindung gebracht wird.

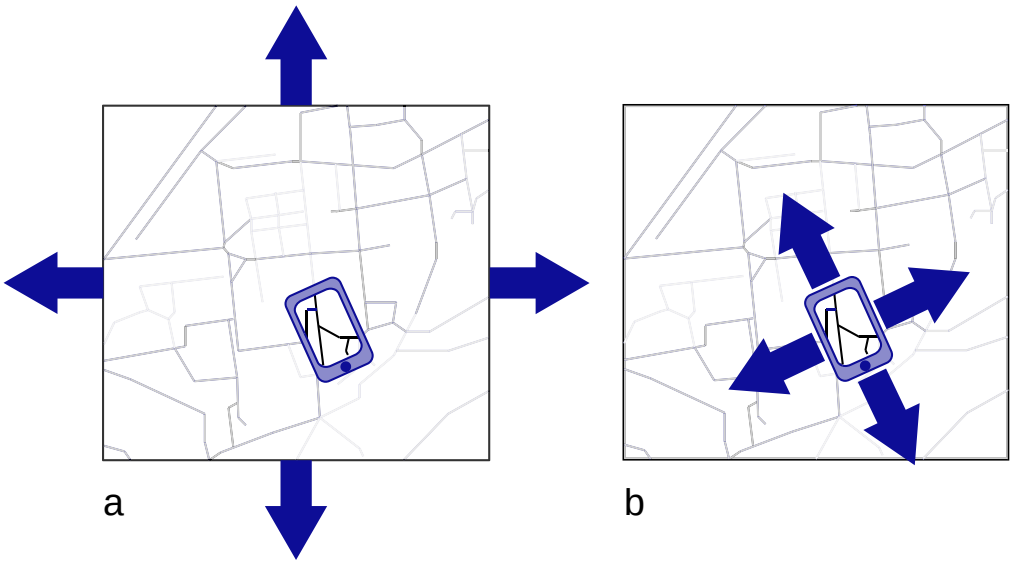
Wird nun Kontext bei der Interaktion mitberücksichtigt, wie z.B. die Verwendung einer geografischen Koordinate im Rahmen eines mobilen Navigationsdienstes, kann nur noch bedingt von einem expliziten Zusammenhang zwischen der Veränderung des Ortes und dem Willen des Benutzers, mit dem mobilen Navigationssystem zu interagieren, gesprochen werden. Der Benutzer bewegt sich während der Navigation vom Startpunkt in Richtung des Zielpunktes. Diese Bewegung ist üblicherweise dadurch motiviert, das Ziel zu erreichen und nicht, um mit einem mobilen System zu interagieren. Trotzdem führt die Bewegung zu einer Veränderung des Kontexts, welcher das Navigationssystem maßgeblich steuert und schließlich dazu führt, dass Weghinweise zum richtigen Zeitpunkt und am richtigen Ort gegeben werden. Man spricht in diesen Fällen von **impliziter Interaktion**, da die Eingabe des Systems über den Kontext erfolgt, der in der Regel nicht explizit durch den Benutzer vorgegeben oder verändert wird, obwohl die Kontextveränderung durchaus durch das Verhalten des Benutzers (bei der Navigation durch die Bewegung zum Zielort hin) bedingt wird [98].

Der Einfluss der impliziten Interaktion auf die Entwicklung mobiler Systeme ist nicht zu unterschätzen und kann Segen oder Fluch sein. Als positive Aspekte sind die nebenläufige Bedienung und Anpassung zu nennen. Man stelle sich z.B. ein Navigationssystem vor, welchem der Benutzerort immer explizit mitgeteilt werden muss. Aktuelle Forschungsarbeiten beschäftigen sich mit dem ultimativen Ziel, dass Benutzer nur noch implizit mit Systemen interagieren. **Proactive Computing** bezeichnet eine Forschungsrichtung, welche versucht, Verfahren zu entwickeln, die abhängig vom Kontext die Pläne und **Ziele** von Benutzern erkennen, um dann diese Ziele vollautomatisch zu unterstützen [112]. Der Erfolg solcher Systeme ist stark davon abhängig, mit welcher Präzision die Ziele der Benutzer erkannt werden können und wie viele Fehler die Systeme in diesem Zusammenhang machen. Die Verbesserung der Erkennungsleistung ist weiterhin wichtigstes Ziel.

Negative Aspekte der impliziten Interaktion hängen mit dem Kontrollverlust zusammen, den der Benutzer erfährt. Im Rahmen vieler Anwendungen ist gar nicht klar, welche impliziten Interaktionen akzeptiert werden oder in welcher Qualität diese erfolgen. Die automatische Berücksichtigung des Ortes in einer Anwendung kann so zu einem ernsthaften Verlust der Privatsphäre führen, wenn z.B. dem Benutzer nicht bewusst ist, dass eine implizite Interaktion vorliegt, die weitreichende Folgen haben könnte. Je vielfältiger und komplexer die implizite Interaktion, desto schwerer kann diese vom Benutzer kontrolliert werden. Die Anwendungen entwickeln dann ein Eigenleben, welches aus Sicht des Benutzers nicht mehr transparent ist. Umso wichtiger ist es, bei der Entwicklung der Benutzerschnittstellen diese Aspekte zu berücksichtigen, und dem Benutzer zu ermöglichen, die verwendete Kontextinformation einzusehen und zu kontrollieren.

18.4 Visualisierungen für kleine Bildschirme

Ein bedeutsamer Nachteil der mobilen Interaktion wurde in diesem Kapitel schon mehrfach erwähnt: der kleine Bildschirm. Im Gegensatz zum großen Bildschirm des Desktop-PC lassen sich großflächige Inhalte nur durch das *Schlüsselloch* des Mobilgeräts betrachten und auch die Interaktion mit dem kleinen Bildschirm ist weniger ergonomisch (siehe auch die Diskussion zum Fat Finger Problem in Abschnitt 17.1.4). Diese Form der Inter-



mmibuch.de/a/18.5

Abbildung 18.5: Visualisierungen, die über die Ausdehnung und Auflösung eines kleinen Bildschirms (Peephole) hinausgehen, können auf zwei Arten mit kleinen Displays exploriert werden. Links ist ein statisches Peephole zu sehen, hinter dem sich der Hintergrund verschiebt, Rechts ein dynamisches Peephole, welches vor einem statischen Hintergrund verschoben wird.

aktion wird **Peephole Interaction** genannt. Grundsätzlich existieren zwei Möglichkeiten, auf visuelle Inhalte zuzugreifen, wenn diese größer als das Peephole sind: entweder man verschiebt den visuellen Inhalt gegen das Peephole oder man verschiebt das Peephole gegen den Inhalt. Im ersten Fall handelt es sich um ein sogenanntes **statisches Peephole** und es ist der Fall, der heutzutage auf fast allen Smartphones beim Explorieren einer größeren digitalen Karte Verwendung findet. Abbildung 18.5a zeigt diese Situation. Durch die Verwendung von Tasten oder Wischgesten wird der digitale Inhalt im Peephole verschoben - so entsteht der Eindruck, das Peephole würde *statisch* über dem digitalen Inhalt fixiert und der Inhalt dynamisch darunter bewegt.

Der zweite Fall ist in Abbildung 18.5b visualisiert: Hier wird das Peephole verschoben, um den nicht sichtbaren Bereich der Visualisierung zugänglich zu machen. So entsteht der Eindruck, das Peephole sei *dynamisch* gegenüber der Visualisierung (**dynamisches Peephole**). Untersuchungen haben gezeigt, dass die dynamische Variante beim Auffinden und Vergleichen von visueller Information der statischen deutlich überlegen ist [82]. Beide Varianten bedienen unterschiedliche **mentale Modelle** (vgl. auch Abschnitte 5.1 und 5.2.2). Die Umsetzung eines dynamischen Peephole erfordert allerdings einen höheren technischen Aufwand, da die Bewegung des dynamischen Peephole durch Sensorik erfasst und interpretiert werden muss, während das statische Peephole durch einfache Touchgesten realisiert werden kann. Dynamische Peepholes können auch aus ergono-

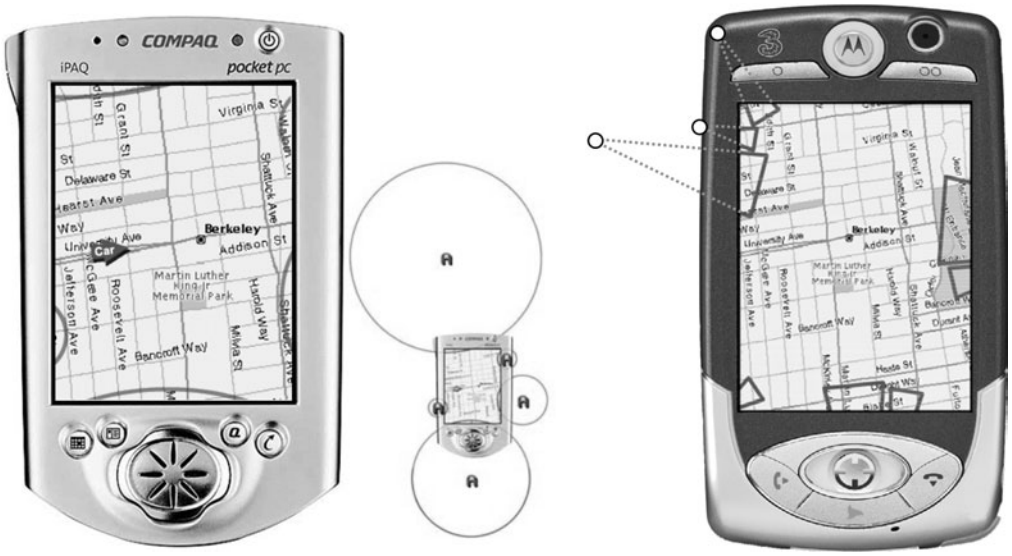


Abbildung 18.6: Links: die Methode *Halo* zur Visualisierung von off-screen Inhalten. Inhalte außerhalb des Bildschirms werden durch einen Kreisausschnitt auf dem Bildschirm visualisiert. Rechts: die Methode *Wedge*, die statt eines Kreisbogen eine Keildarstellung verwendet, deren nicht-sichtbare Spitze auf off-screen Elemente zeigt (Bilder aus Baudisch und Rosenholz [8] und Gustafson et al. [41]).

mmibuch.de/a/18.6

mischer Sicht problematisch sein, da das Gerät bewegt werden muss, was bei längeren Interaktionszeiten ermüdend wirken kann.

Alternativ können größere visuelle Inhalte auch durch die Verwendung eines Zoomable UI (siehe Abschnitt 9.2) realisiert werden. Bei kleinen Bildschirmen verschärft sich allerdings das Fokus & Kontext Problem erheblich, da die Unterschiede zwischen Fokus und Kontext noch stärker ausgeprägt sind (siehe auch Abschnitt 9.3). Eine alternative Lösung für das Fokus & Kontext Problem bei mobilen Geräten mit kleinen Bildschirmen besteht darin, den Benutzer auf Inhalte, die im Peephole nicht sichtbar sind und die sich damit außerhalb des Bildschirms befinden, aufmerksam zu machen. So kann dem Benutzer ein Überblick gegeben werden, wo sich relevante Inhalte außerhalb der momentanen Bildschirmansicht befinden, was die Suche nach Inhalten auf einer digitalen Karte deutlich vereinfacht. Eine entsprechende Visualisierung wird als **off-screen Visualisierung** bezeichnet.

Eine gut funktionierende Lösung zur Darstellung von off-screen Inhalten sollte dem Benutzer zwei wesentliche Aspekte vermitteln: die Richtung und die Entfernung des jeweiligen off-screen Inhaltes gemessen vom Bildschirmrand. Baudisch und Rosenholtz schlagen hierzu die *Halo* Methode vor [8]. Bei dieser Visualisierungsmethode wird um jedes relevante Objekt, welches sich außerhalb des Peephole befindet, ein Kreis geschla-

gen, dessen Radius so gewählt wird, dass er einen Teil des Peephole schneidet. So wird die Richtung des Objektes über den Teil des Peephole definiert, der geschnitten wird, und die Entfernung über die Krümmung des sichtbaren Kreisbogens. In Abbildung 18.6 Links ist zu sehen, dass dies bei nahen Objekten zu kleineren, stark gekrümmten Kreisbögen führt, während bei fernen Objekten die Visualisierung eine schwache Krümmung aufweist, die sich über einen größeren Teil des Bildschirms erstreckt. Die Verwendung von *Halo* führt zu einer Verbesserung der räumlichen Wahrnehmung von Objekten und hat sich bei Suchaufgaben gegenüber einer naiven Pfeildarstellung als vorteilhaft erwiesen. Für wenige off-screen Objekte funktioniert diese Methode sehr gut. Bei vielen off-screen Objekten und entsprechender Anzahl von Kreisbögen wird die Darstellung schnell unübersichtlich. Gustafson et al. schlagen zur Verbesserung statt der Visualisierung durch Kreisbögen die Verwendung von Keildarstellungen vor [41]. Wie in Abbildung 18.6 Rechts gezeigt, werden die stumpfen Keilenden im peephole visualisiert. Der Ort der nicht sichtbaren Keilspitze wird durch die Position des off-screen Objekts definiert. Die Visualisierung des Keils besitzt drei Freiheitsgrade, die zur Verhinderung von Überlappungen verwendet werden können. Der Keil kann leicht um das off-screen Objekt rotiert, die Öffnung der Keilspitze angepasst und die Keillänge modifiziert werden. Im Gegensatz zum verhältnismäßig starren Kreisbogen lassen sich die Keil-Visualisierungen so wesentlich besser anpassen, ohne dass der Informationsgehalt verloren geht. Während *Halo* zu starken Überlappungen der Kreisbögen führt, kann *Wedge* diese durch geeignete Parametrisierung von Rotation, Öffnung und Länge der Keile vermeiden.

18.5 Mobile Interaktionskonzepte

Die Besonderheiten der mobilen Geräteplattform spiegeln sich in den Interaktionskonzepten wider, die zur Bedienung mobiler Applikationen benutzt werden. Klassische Eingabekonzepte verlieren an Bedeutung, wenn mobile Geräte mit kleinem Bildschirm unterwegs bedient werden sollen. Das WIMP-Konzept für den Desktop-PC (siehe Abschnitt 15.2) lässt sich aufgrund der fehlenden und schlecht zu ersetzenden Eingabegeräte Tastatur und Maus nicht ohne weiteres auf die mobile Interaktion übertragen, auch wenn dies z.B. mit Windows CE bei der ersten Generation von mobilen PDA-Geräten (Personal Digital Assistant) mit wenig Erfolg versucht wurde. Maus-äquivalente Eingaben erfolgen bei mobilen Geräten in der Regel über den touch-sensitiven Bildschirm, lassen sich aber nicht 1:1 übertragen, da die Eingabepräzision der Interaktion mit Finger oder Stylus deutlich schlechter ist als die sehr guter PC-Mäuse (siehe hierzu auch die Diskussion zum Fat-Finger Problem in Abschnitt 17.1.4 und zum Modell der drei Zustände in Abschnitt 17.1.2).

Auch wenn die klassische QWERTZ-Tastatur weiterhin als Teil mobiler Betriebsysteme zu finden ist, so ist ihre Ergonomie im Vergleich zur Desktop-PC Tastatur deutlich schlechter, insbesondere weil sie meist als virtuelle Tastatur auf dem touchsensitiven Bildschirm realisiert wird. Ohne physikalischen Druckpunkt und in ihrer üblicherweise reduzierten Größe fällt die Performanz einer virtuellen Tastatur deutlich hinter die der physikalischen Variante zurück[2]. Die **mobile Spracheingabe** wurde schon im

[2] Während erfahren Benutzer mit einer physikalischen Tastatur bis zu 200 Wörter pro Minute tippen

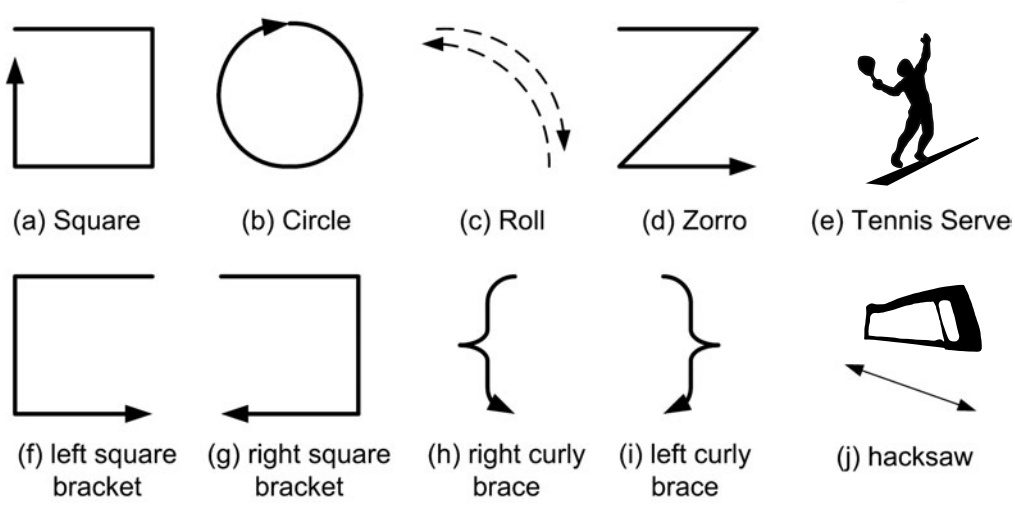
vorherigen Abschnitt 18.2.4 im Rahmen der Fußgängernavigation kurz angesprochen. Spracheingabe besitzt den großen Vorteil, dass sie ohne Einsatz der Hände verwendet werden kann und sich so für die Interaktion mit mobilen Geräten besonders eignet. Die aktuellen mobilen Betriebssysteme iOS und Android besitzen eine integrierte Sprachverarbeitung, die sowohl zur allgemeinen Texteingabe (z.B. zum Diktieren von E-Mails) oder zur Steuerung des mobilen Gerätes verwendet werden kann (z.B. zum Starten einer Anwendung auf dem Smartphone). Aufgrund der hohen Anforderungen der Sprachverarbeitung an Rechenleistung und Speicherplatz wird diese nicht auf dem mobilen Gerät selbst durchgeführt, sondern auf einen speziellen Server ausgelagert. Aus diesem Grund steht diese Funktionalität auch nur bei einer bestehenden Netzwerkverbindung des mobilen Gerätes zur Verfügung.

Sprachverarbeitungssysteme können entweder **sprecherunabhängig** oder **sprecherabhängig** konzipiert sein. Die sprecherunabhängige Sprachverarbeitung kann direkt von beliebigen Personen mit unterschiedlichen kulturellen Hintergründen ohne Trainingsphase verwendet werden. Diesen Vorteil erkauft man sich durch eine höhere Fehlerrate der Erkennung [73]. Die sprecherabhängige Variante hingegen erfordert eine Trainingsphase durch den Benutzer, erkennt dann aber Spracheingaben des spezifischen Benutzers mit hoher Zuverlässigkeit, häufig unter der Verwendung von **Hidden-Markov-Modellen** [52]. In der Regel werden sprecherunabhängige Systeme in offenen Diskursdomänen eingesetzt, die eine potenziell große Menge an möglichen Eingaben erlauben, wie dies insbesondere bei der Bedienung von mobilen Betriebssystemen erforderlich ist. Sprecherabhängige Systeme hingegen finden Verwendung in hochspezialisierten Domänen, in denen auf hohe Erkennungsraten Wert gelegt wird und eine Trainingsphase in Kauf genommen werden kann. Ein typisches Einsatzbeispiel ist die Unterstützung von Lagerarbeitern oder Radiologen durch ein sprachbasiertes System [72].

Der touch-sensitive Bildschirm, über den die meisten mobilen Geräte verfügen, erlaubt den Einsatz unterschiedlichster Touch-Gesten. In Kombination mit Visualisierungsmethoden für kleine Bildschirme (siehe Abschnitt 18.4) können Benutzern effektive Interaktionstechniken zur Verfügung gestellt werden. Mobile Geräte eigenen sich eher zur einhändigen Touch-Interaktion, da sie mit der anderen Hand gehalten werden müssen und reduzieren so die Anzahl der möglichen Gesten. Eine interessante Variation der einhändigen Interaktion tritt ein, wenn das Gerät mit der Hand bedient wird, mit der es gehalten wird. Dies kann dann üblicherweise nur mit dem Daumen erreicht werden und in Abhängigkeit von der individuellen Anatomie und der Größe des Bildschirmes kann in einem solchen Fall nicht der gesamte Bildschirm zur Interaktion verwendet werden. Methoden des maschinellen Lernens werden eingesetzt, um in diesen Fällen eine höhere Präzision der Eingabe zu erreichen [124].

Neben Gesten auf dem touch-sensitiven Bildschirm bietet sich noch ein zweites gestenbasiertes Interaktionskonzept bei mobilen Geräten an: die **Gesteninteraktion** mit dem Gerät selbst. Eingebaute Sensoren (siehe auch Abschnitt 18.6) erlauben die Erfassung einer Bewegungstrajektorie des mobilen Gerätes, die durch geeignete Algorithmen zur Erkennung einer Geste herangezogen werden kann [5]. Abbildung 18.7 zeigt zehn Gesten, die mit dem mobilen Gerät einfach ausgeführt werden können. Einige der Gesten finden

lassen sich bei speziellen virtuellen Tastaturen, die nur mit einer Hand bedient werden können, bis zu 50 Wörter pro Minute erzielen.



mmibuch.de/a/18.7

Abbildung 18.7: Eine Menge von einfachen Gesten, die mit einem mobilen Gerät ausgeführt werden können und mit einer Erkennungsrate von 80 Prozent verhältnismäßig robust zu erkennen sind [63].

sich auch bei gestenbasierten Spielekonsolen, wie z.B. der Nintendo Wii, der Microsoft Xbox und der Sony Playstation. Diese Menge von Gesten lässt sich durch einen robuste Methode auch auf mobilen Geräten recht verlässlich mit 80 Prozent erkennen [63].

Gesten können neuartige Interaktionskonzepte ermöglichen, wenn sie nicht nur mit dem Gerät selbst, sondern mit einer oder beiden freien Händen über dem Gerät oder um das Gerät herum ausgeführt werden. In diesen Fällen spricht man von **Around-Device Interaction** (dt. Geräteumgebungsinteraktion). Hierbei nutzt das Smartphones Sensorik, die seine Umgebung erfasst. Dies können entweder Kameras sein (2D oder 3D) oder Abstandssensoren, die z.B. Infrarotlicht zum Messen der Handgesten neben oder über dem mobilen Gerät verwenden [15, 62]. Der Einsatz einer **Tiefenkamera**, die eine Handgeste neben dem Gerät in den drei Raumdimensionen erfasst, erlaubt es sogar, eine Hand als 3D-Maus einzusetzen, um zum Beispiel mit komplexen 3D-Grafiken auf mobilen Geräten zu interagieren [64]. Die weitere Miniaturisierung mobiler Geräte und die Verbesserung der Performanz von Erkennungsalgorithmen wird in Zukunft weitere mobilen Interaktionskonzepte ermöglichen. Neben der oben erwähnten Verwendung von Tiefenkameras hat insbesondere die **mobile Erfassung von Blickbewegungen** großes Potenzial für die Interaktion mit mobilen Geräten. Die Blickbewegungserfassung alleine reicht in vielen Situationen zu einer robusten Interaktion nicht aus, da es zu Fehlinterpretationen über das gewünschte Ziel des Benutzers kommen kann (ganz in Analogie zum Midas-Touch-Problem, welches in Abschnitt 17.1 beschrieben wird). Allerdings kann sie in Kombination mit anderen Modalitäten (z.B. einer Geste) durchaus erfolgreich auch in mobilen Szenarien eingesetzt werden [55].

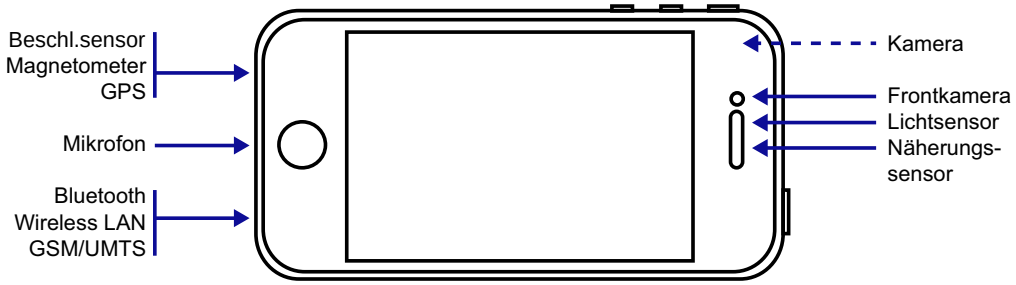


Abbildung 18.8: Beispiel von integrierter Sensorik in einem modernen Smartphone - Hier am Beispiel des iPhone 4, welches mehr als zehn Sensoren besitzt.

mmibuch.de/a/18.8

18.6 Mobile Sensorik

Wie im vorherigen Abschnitt beschrieben, benötigen viele der mobilen Interaktionskonzepte integrierte Sensorik, die in einem modernen Smartphone bereits zahlreich vorzufinden ist. Moderne mobile Geräte enthalten häufig mehr als zehn Sensoren oder technische Sende- und Empfangskomponenten, die auch als Sensorik eingesetzt werden können. In Abbildung 18.8 sind diese exemplarisch am Beispiel des iPhone 4 aufgelistet. Ein **GPS**-Chipsatz wird zur Erkennung des Geräteortes außerhalb von Gebäuden mit hoher Genauigkeit (bis zu wenigen Metern) eingesetzt, der insbesondere bei der impliziten Interaktion zur Steuerung von Navigationssystemen Verwendung findet (siehe Abschnitt 18.2.4). Ein **Gyroskop** oder Kreiselinstrument wird zur präzisen Bestimmung der Lage eingesetzt und misst die Orientierung im Raum. Der eingebaute **Beschleunigungssensor** misst hingegen die Beschleunigung und damit die Bewegung des Geräts. Beide Sensoren werden häufig kombiniert, da sich ihre Informationen zu einer vollständigen Positions- und Orientierungsbeschreibung ergänzen lassen [6].

Ein **Magnetometer** misst das Erdmagnetfeld und wird häufig zur Bestimmung der Orientierung des Gerätes bei der Navigation verwendet. Theoretisch können auch mobile Gesten mit dem Magnetometer erkannt werden, allerdings wird hierzu häufiger das präzisere Gyroskop eingesetzt. In Forschungsansätzen wird das Magnetometer auch zur Lokalisierung in Innenräumen verwendet. Dazu müssen allerdings magnetische Markierungen in der Umgebung installiert werden [109].

Ein **Lichtsensor** misst die Umgebungshelligkeit und versetzt das Gerät in die Lage, die Bildschirmhelligkeit der Umgebungshelligkeit automatisch anzupassen. Die integrierten **Kameras** sind besonders leistungsstarke Sensoren. Viele moderne Geräte besitzen zwei oder mehr solcher Kameras, die üblicherweise unterschiedliche Auflösungen und Funktionsumfänge aufweisen. Für die mobile Interaktion sind Kameras sehr wichtig, da sie nicht nur zur Aufnahme von Bildern verwendet werden können, sondern eine Vielzahl von weiteren Informationen liefern. So lässt die Bewegungsrichtung der Bildpunkte eines Kamerabildes in vielen Fällen auf die Eigenbewegung des mobilen Gerätes schließen.

Dies wird z.B. verwendet, um Panoramabilder zu erstellen oder mobile Gesten zu erkennen. Im nächsten Abschnitt wird darüber hinaus beschrieben, welche Rolle die Kamera eines Mobilgerätes zur Erkennung der Umgebung spielen kann. Das eingebaute Mikrofon (häufig in Verbindung mit einem zweiten Mikrofon zur Rauschunterdrückung) dient nicht nur der Sprachtelefonie sondern ermöglicht eine Reihe mobiler Dienstleistungen, wie z.B. die automatische Erkennung von Musiktiteln, die gerade im Radio laufen.

Jedes mobile Gerät verfügt über Kommunikationstechnik, wie z.B. ein GSM/UMTS-Modul zur Sprach- und Datenkommunikation. Wireless Lan-Komponenten sind ebenso häufig vorzufinden wie die Möglichkeit, Peripheriegeräte mittels Bluetooth zu verbinden. Während der primäre Zweck dieser Komponenten die Übertragung von Daten ist, lassen sie sich auch als Sensoren einsetzen, da durch sie das Mobilgerät mit einer Infrastruktur kommuniziert. Da die Sendeleistung eines mobilen Gerätes beschränkt ist und die Signalstärke mit der Entfernung quadratisch abnimmt, lässt sich der Aufenthaltsort relativ zu den Empfangs- und Senderichtungen in der Umgebung grob abschätzen. Dies funktioniert im Prinzip mit allen Kommunikationsarten, wird aber insbesondere bei Wireless Lan auch zur Positionierung in Gebäuden eingesetzt, da dort in der Regel kein verlässlicher GPS-Empfang möglich ist [76].

Einige Mobilgeräte verwenden **Near Field Communication** (**NFC**) zur Kommunikation bzw. zur Identifikation von Geräten und Objekten. Die Besonderheit von NFC ist die sehr geringe Reichweite (wenige Zentimeter) und die damit verbundene Abhörsicherheit, die bei bestimmten Einsatzgebieten, wie z.B. dem mobilen Bezahlen, von großer Bedeutung sind. NFC kann aber auch zur Interaktion eingesetzt werden, indem NFC-Tags in die Umgebung eingebracht werden und die Bewegung des mobilen Gerätes über diesen Tags gemessen wird. So kann z.B. mit einem Poster interagiert [14] oder eine Gebäudenavigation realisiert werden [90].

18.7 Mobile Augmented Reality

Das Konzept der **Augmented Reality** (**AR**, dt. Erweiterte Realität) beschreibt die Überlagerung von digitalen Inhalten über der realen Welt durch ein technisches Hilfsmittel, z.B. durch eine halbdurchlässige Datenbrille wie in Abbildung 18.9a zu sehen oder durch ein tragbares Gerät, z.B. ein Smartphone entsprechend der Abbildungen 18.9b und 18.9c. Während der Benutzer bei der halbdurchlässigen Brille die Realität direkt wahrnimmt, wird diese bei PDA und Smartphone indirekt durch die Kamera des Geräts aufgenommen. Man spricht in diesem Fall auch von **video-based AR**.

Mithilfe von AR lässt sich eine Vielzahl von mobilen Diensten implementieren, z.B. zur Unterstützung von Arbeitern, die am Werkstück direkt die nächsten Bearbeitungsschritte sehen können, oder wie in Abbildung 18.9c zu sehen, um einem Touristen Information zum nächsten Restaurant anzuzeigen[3]. Damit AR gelingen kann, muss die digitale Information passgenau zur Realität angezeigt werden, d.h. die digitalen Inhalte müssen präzise an der richtigen Stelle erscheinen. Bei video-based AR muss dem System bekannt sein, wie die digital zu erweiternden realen Objekte (im Beispiel 18.9c das Restaurant) im Bezug zur Kamera liegen. Dieser Vorgang wird **Registrierung** genannt und kann auf

[3] Wikitude, siehe www.wikitude.de

Abbildung 18.9: Mobile Augmented Reality in drei Generationen. Links: eines der ersten tragbaren Systeme [50] Anfang der 2000er Jahre, Mitte: ein markerbasiertes AR-System, welches in der Hand getragen werden kann [97], Rechts: ein aktuelles Smartphone AR-System.

mmibuch.de/a/18.9

unterschiedlichen Wegen durchgeführt werden. Die ersten Ansätze verwendeten **Visuelle Marker**, deren geometrische Form, Ausprägung und Größe bekannt sind, und die daher durch gängige Bildverarbeitungsverfahren leicht im Videobild zu erkennen sind. Wird ein solcher Marker im Bild erkannt, kann aus den Eigenschaften der Kameralinse und der perspektivischen Verzerrung die relative Lage (Entfernung und Orientierung) des mobilen Gerätes zum Marker bestimmt werden. Mit dieser Information lässt sich dann ein digitales Objekt relativ zum Marker genau positionieren. In Abbildung 18.9b wurde ein solcher Marker verwendet, um eine digitale Figur über dem Marker zu platzieren. Der Nachteil der Verwendung von visuellen Markern liegt auf der Hand, da eine Platzierung von visuellen Elementen in der Umgebung nicht immer möglich und häufig aus ästhetischen Gründen unerwünscht ist. In einigen speziellen Fällen ist es sinnvoll, unsichtbare Marker zu verwenden, die einen nicht sichtbaren Spektralbereich des Lichtes verwenden (z.B. im infraroten Bereich [91]). Selbst in diesem Fall müssen die Marker vorher in der Umgebung platziert werden, und damit die Registrierung erfolgreich gelingt, muss der Marker sich im Sichtbereich der Kamera befinden.

Eleganter lässt sich das Problem der Registrierung lösen, wenn ausschließlich Eigenschaften der Umgebung selbst zur Registrierung verwendet werden. Dies bezeichnet man als **natural feature tracking** (dt. Verfolgung natürlicher Strukturen). Natural Feature Tracking funktioniert sehr gut, wenn genügend Struktur in der Umgebung vorzufinden ist. Mobile AR Anwendungen, die *natural feature tracking* verwenden, werden von einigen Printmedien eingesetzt. Solche Anwendungen erlauben es, zusätzliche digitale Inhalte, wie z.B. ein Video, direkt über einem Zeitungsartikel oder einem Reklameprospekt einzublenden und können damit eine Brücke bilden zwischen klassischen

und digitalen Medien. Auch wenn es gelingt, diesen Ansatz auf einen größeren Teil der Realität zu übertragen, ist dies noch eine offene Forschungsfrage, so dass ein flächendeckender mobiler Augmented Reality Dienst wohl noch als Zukunftsmusik bezeichnet werden muss.

mmibuch.de/l/18

Übungsaufgaben:

1. Diskutieren Sie das Problem der Unterbrechbarkeit anhand des Beispiels eingehender Telefonanrufe auf dem Smartphone. Nennen Sie drei Situationen bei denen die Unterbrechung durch einen Telefonanruf Probleme bereiten könnte. Überlegen Sie sich jeweils eine präventive und eine kurative Strategie (Abschnitt 18.1), die helfen könnten die Kosten der Unterbrechung abzumildern. Beschreiben Sie welche Rolle der Interaktionskontext (Abschnitt 18.2) in diesen Situationen spielt.
2. Diskutieren Sie die Vor- und Nachteile einer einfachen Gestensteuerung für Smartphones (wie in Abbildung 18.7 gezeigt). Welche Funktionen des Smartphones könnten grundsätzlich einfach, welche schwierig und welche gar nicht mit Hilfe von Gesten ausgeführt werden?
3. Analysieren Sie die Usability von mobilen Augmented Reality Systemen, die mit Hilfe von Smartphones realisiert werden (siehe Abbildung 18.9b und18.9c). Welche Vor- bzw. Nachteile bestehen im Vergleich zu brillenbasierten Augmented Reality Systemen (Abbildung 18.9a)?