FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Potentiale und Einsatzmöglichkeiten einer Tablet-App zum intuitiven und kollaborativen Annotieren von Video

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur/in

im Rahmen des Studiums

Medieninformatik

eingereicht von

Philipp Kastner
Matrikelnummer 0625504

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung
Betreuer/in: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Peter Purgathofer

Wien, 27.01.2014 _____ _____

(Unterschrift Verfasser/in) (Unterschrift Betreuer/in)

Erklärung zur Verfassung der Arbeit

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 27.01.2014

(Unterschrift Verfasser/in)

Hiermit möchte ich mich herzlich bei Peter Purgathofer
für die zahlreichen Anregungen und die freundliche Betreuung bedanken.

*

Ein besonderer Dank gilt meinen Eltern Brigitte und Friedrich,
die mich immer unterstützen und mir einen sicheren Rückhalt bieten.

*

Vielen Dank an meine Geschwister und meine Freundin Viktoria,
die mir in vielen Situationen weitergeholfen haben.

Kurzfassung

Die Annotation dient der intensiven Auseinandersetzung mit unterschiedlichen Inhalten. In dieser Arbeit wird speziell das Annotieren im Zusammenhang mit Videos und einer dafür entwickelten Tablet-App genauer betrachtet. Dafür wird zunächst ein grundlegendes theoretisches Verständnis im Bereich der Interaktion mittels Gesten und dem Umgang mit dem Medium Video erarbeitet. Durch den Einsatz von Touchscreens bieten Tablets neue Möglichkeiten im Bereich des Interaktionsdesigns. Inwiefern sich die Eingabe mittels Finger oder Stylus auf das User-Interface Design auswirkt wird untersucht und in weiterer Folge auf die Veränderung des Medium Video in den letzten Jahren und dessen Einsatzgebiete eingegangen.

Es existieren bereits einige Anwendungen zum Annotieren von Videos allerdings wurde diese meist für ganz spezielle Einsatzgebiete entwickelt und sind einigermaßen komplex. Durch die Erarbeitung von möglichen Einsatzgebieten einer intuitiven Tablet-App zum Annotieren von Video wird gezeigt, dass in vielen unterschiedlichen Bereichen ein Bedarf für eine solche App besteht. Außerdem werden die Möglichkeiten des kollaborativen Arbeitens betrachtet. In vielen Fällen werden Annotationen nämlich nicht nur für den eigenen Gebrauch erstellt, sondern auch mit anderen ausgetauscht. Was bei der Entwicklung von Software, die die Zusammenarbeit von mehreren Personen ermöglicht beachtet werden muss und welche Chancen dabei entstehen sind dabei zentrale Fragestellungen.

Neben der theoretischen Auseinandersetzung wurde im Rahmen der Arbeit auch ein Prototyp für das iPad umgesetzt. Dieser wurde verwendet um diverse Konzepte zu validieren. Zum Abschluss werden noch Verbesserungsmöglichkeiten für eine zukünftige iterative Weiterentwicklung der Tablet-App aufgeführt.

Abstract

To be able to annotate different kind of content is essential for different tasks like research, teaching or learning. This thesis specially examines the annotation of video as well as the potentials of a tablet-app to do so. In the beginning some fundamental theoretical understanding in the area of gesture interaction as well as the use of video is developed. Because of the use of touchscreens tablets offer various new opportunities in area of interaction design. Especially the differences between touch input and the use of a stylus regarding the user interface design are interesting. Also the recent changes of video and the usage of video in general will be discussed.

There already exist some desktop applications to annotate videos. The problem is that most of them were developed with a specific use case in mind and that they are not easy to use. The research of various different application domains for video annotation shows that there is definitely a demand for such a tablet-app.

The opportunities of collaborative work is discussed as well. A lot of times annotations are not only created for personal use but also shared with other people. The important factors of developing collaborative software as well as the resulting changes are also research topics of this thesis.

Beside the theoretical examination a prototype for the iPad was developed. It was used to demonstrate and validates various concepts. At the end a set of improvements for the future iterative development of the tablet-app is presented.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

KAPITEL 1

Einleitung

Das Annotieren von Texten hat eine lange Tradition und wurde bereits in Manuskripten der Antike betrieben. Anmerkungen können unter anderem verwendet werden, um Wichtiges hervorzuheben, Bestehendes zu ergänzen oder auf Fehler hinzuweisen. Annotieren beschränkt sich allerdings nicht nur auf Text alleine. Andere Inhalte wie Bilder, Videos oder Audioaufnahmen können auch annotiert werden. Insbesondere im Videobereich gibt es nur sehr umfangreiche, teilweise sehr komplexe Software, die meistens auf ganz spezielle Anwendungsfälle ausgelegt ist. Jedoch ist das Annotieren von Videoinhalten vermutlich für ein viel breiteres Publikum von Interesse. Studenten könnten sich mit Hilfe eines solchen Tools Notizen zu aufgezeichneten Vorlesungen machen, Firmen wären in der Lage Verkaufsgespräche zu analysieren und Sporttrainer könnten auf eine einfache Weise eine Videoanalyse durchführen. Im Rahmen der Lehrveranstaltung From Design to Software wurde bereits ein Prototyp in Form einer iPad Applikation zum einfachen Annotieren von Videos entwickelt. Dieser Prototyp soll als Grundlage für diese Diplomarbeit dienen.

In den letzten Jahren haben mobile Geräte mit integrierten Touchscreens immer mehr an Bedeutung gewonnen. Dadurch ergaben sich neue Möglichkeiten sowie Schwierigkeiten bei der Interaktion mit User-Interfaces. Der Finger, und somit der Mensch selbst, ersetzt die bei bisherigen Computersystemen notwendigen Eingabegeräte. Diese neuen Voraussetzungen dürfen bei der Entwicklung, sowie beim Design von Interaktion und User-Interfaces nicht vernachlässigt werden. Auch das Medium Video selbst bringt gewisse Herausforderungen im Bezug auf die Interaktion mit sich. So gibt es zwar beispielsweise interessante Lösungsansätze [1] für die feine Positionierung innerhalb eines Videos aber noch keine Standardlösung. Ein weiterer Trend bei der Entwicklung von Software ist die Unterstützung des kollaborativen Arbeitens. Gerade im Bereich von Annotation kann das sehr hilfreich sein, denn Annotationen sind oft neben dem persönlichen Gebrauch, auch für andere Personen interessant. Kommentare und Anmerkungen

helfen unterschiedliche Perspektiven aufzuzeigen und ermöglichen dadurch eine Diskussion zu den betreffenden Inhalten. Neben den Vorteilen des zeit- und ortsunabhängigen Arbeitens bringt kollaboratives Arbeiten auch Nachteile mit sich, die nicht vernachlässigt werden dürfen.

1.1 Erwartetes Resultat

Durch die intensive Beschäftigung mit den unterschiedlichen Interaktionsmöglichkeiten mit Tablet-Computer und Videos, sollen Interaktionskonzepte erarbeitet werden, die sich besonders gut für eine Applikation zur Videoannotation eignen. Ziel ist es, dass Personen ohne Vorkenntnisse im Annotationsbereich die Applikation nutzen können. Dabei soll kein bestimmtes Szenario vorgegeben werden, sondern die Applikation für ein breites Feld von Anwendungen geeignet sein.

Um dies zu erreichen, wird basierend auf den erarbeiteten theoretischen Grundlagen, sowie durch User Feedback, ein Konzept zur Weiterentwicklung des Prototyps erarbeitet.

1.2 Methodisches Vorgehen

Im Zuge der Diplomarbeit wird Licht auf relevante Problemfelder rund um die Themenstellung geworfen. Zum einen ist das die Gestaltung von Interaktion jenseits der üblichen Maus-Tastatur-Bildschirm-Systeme, zum anderen die Interaktion mit Videoinhalten, und schließlich die Randbedingungen der Umsetzung von IKT-Systemen zum kollaborativen Arbeiten. Nachdem die Anwendung in der Mischung aus Einfachheit und Funktionalität in dieser Form bisher noch nicht verfügbar war, liegt nahe, die weitere Ausgestaltung mittels eines explorativen Vorgehens zu finden. Dafür wurde die App einerseits als früher Prototyp in den iTunes App-Store gestellt und mit Hinweisen auf die Möglichkeit des Einbringens von Feedback versehen. Andererseits wurde eine Reihe von Gesprächen angesetzt, um Vektoren für die weitere Entwicklung zu Sammeln. Dabei wurde darauf Wert gelegt, die Gesprächspartner so zu wählen, dass sie einen unmittelbaren Bezug zur (kollaborativen) Annotation oder Klassifikation von Video haben. Aus dem so gesammelten Feedback wird ein Konzept zur iterativen Weiterentwicklung der Applikation gestaltet, mit der zuvor nicht sichtbare Potentiale und Einsatzmöglichkeiten der neuen Interaktionsformen von Tablets in diesem interaktiven Anwendungsfeld umgesetzt werden.

1.3 State-of-the-Art

Es gibt bereits einige Programme [2, 3, 4] für Desktop-Computer, die das Annotieren von Videos ermöglichen. Diese Programme werden meistens im wissenschaftlichen Bereich eingesetzt und besitzen komplexe User-Interfaces. Diese User-Interfaces eignen sich nur bedingt für mobile Geräte [5]. Generell können bisherige Regeln und Guidelines zur Entwicklung von

User-Interfaces nicht einfach 1:1 auf mobile Geräte übernommen werden. Interaktionen müssen neu überdacht und angepasst werden. Ramos G. & Balakrishnan R. [1] zeigen mögliche Interaktionstechniken zum Steuern und Annotieren von Videos mit Hilfe eines Stiftes auf einem Tablet-Computer. Inwieweit diese Techniken auf aktuellen Tablets anwendbar sind ist zu erforschen.

Welche Vorteile das kollaborative Annotieren mit sich bringen kann beschreiben Ellis S. & Groth D. [6] in ihrer Arbeit. Darüber hinaus gibt es offensichtlich beim Annotieren selbst auch Unterschiede darin, ob Annotationen für persönliche Zwecke oder für die Öffentlichkeit erstellt werden [7].

1.4 Bezug zum angeführten Studium

Das Masterstudium Medieninformatik beschäftigt sich intensiv mit dem Thema Interaktion Design, das in den LVAs User Interface Design, Beyond the Desktop oder Interface und Interaction Design auf verschiedenen Ebenen behandelt wird. Weitere Methoden zum Generieren und Evaluieren von Entwürfen, die für diese Diplomarbeit sehr hilfreich sind, wurden in den LVAs Exploratives Design und Projektorientierte Recherche und designgenerierende Methoden kennen gelernt. Auch die Wichtigkeit von kollaborativen Arbeiten wurde in der LVA Koopartives Arbeiten, sowie Grundlagen von CSCW-Systemen erläutert.

KAPITEL 2

Gestaltung von Interaktion Beyond the Desktop

Die Gestaltung von Interaktion jenseits der üblichen Desktop Systeme ist ein wichtiges Themengebiet, wenn man sich mit der Erstellung von Tablet-Apps beschäftigt. Durch die Verfügbarkeit und Verbreitung neuer Hardware ist eine Vielzahl neuer Möglichkeiten zur Interaktion entstanden. Dieses Kapitel beschäftigt sich mit der Entwicklung der Human-Computer Interaction und dem damit verbundenen Wandel der User-Interfaces. Außerdem wird die Interaktion mittels Touchscreen näher betrachtet. Welches Potential bieten Touchscreens, welche Eigenschaften besitzen sie und wo liegen ihre Schwächen? Ein Fallbeispiel soll demonstrieren, welche Schritte nötig sind, um eine Toucheingabe in eine Linie zu verwandeln und welche Schwierigkeiten dabei auftreten.

"Tap is the new Click" titelt Dan Saffer in seinem Buch *Designing Gestural Interfaces* [8, p. 3] und geht davon aus, dass wir uns durch auf Gesten basierende Interaktion mittlerweile in eine neue Ära des Interaktionsdesigns begeben haben. Wie genau Gesten aufgebaut sind und wie sie funktionieren, wird in diesem Abschnitt ebenso untersucht, wie deren Gestaltung. In weiterer Folge wird betrachtet, inwiefern Gesten dabei helfen, Interaktion zu vereinfachen und anschließend werden einige grundlegende Gesten vorgestellt, die bereits eine weite Verbreitung auf verschiedenen Systemen gefunden haben. Es wird außerdem darauf eingegangen, warum der Finger nicht immer die beste Eingabemethode ist und welche Vorteile die Verwendung eines Stylus bietet (v.a. in Hinblick auf Annotation). Abschließend werden noch einige Herausforderungen die beim Interaktionsdesign für Touchscreens auftreten angesprochen.

2.1 Human Computer Interaction

Um die Funktionen eines Computers nutzen zu können, ist es notwendig geeignete Interfaces zur Verfügung zu stellen. Die Interaktion zwischen Menschen und Computer hat in den letzten Jahren unzählige Veränderungen erfahren und entwickelt sich kontinuierlich weiter. Viele dieser Entwicklungen beruhen auf Erkenntnissen von Forschung im Bereich der Human-Computer Interaction (HCI). Der Begriff HCI tauchte bereits im Jahr 1975 das erste Mal auf [9] und wurde von der Special Interest Group on Computer-Human Interaction (SIGCHI) folgendermaßen definiert:

> *"Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them."* [10]

Die Disziplin der HCI ist breit gefächert und beschäftigt sich grundsätzlich mit der Verbindung zwischen der Funktionalität und der Bedienbarkeit eines Systems. Funktionalität bekommt nur dann einen Wert, wenn die Funktionen auch effektiv durch den Menschen genutzt werden können [11].

2.2 User-Interfaces im Wandel

Eine der ersten Möglichkeiten die zur Interaktion mit Computern entwickelt wurde, waren Command-Line Interfaces (CLIs). Bei CLIs werden Programme durch die Eingabe von Text mittels einer Tastatur gesteuert. Nach den CLIs wurden Graphical-User-Interfaces (GUIs) entwickelt, um die Interaktion zu vereinfachen. GUIs ermöglichen die Bedienung durch direkte Manipulation von Icons und anderen grafischen Elementen. Sie basieren auf der Desktop Metapher, bei der ein klassischer Schreibtisch nachgebildet wird, um auf bereits bekannte Interaktionen (xxx zB?) zurückgreifen zu können und dadurch die Nutzung zu vereinfachen. Bei der Gestaltung von Interaktion werden solche Metaphern häufig verwendet. Das Modell auf denen aktuelle GUIs basieren, wird als WIMP (Akronym für Windows, Icons, Menus, Pointer) bezeichnet. GUIs haben die auftretenden Hürden bei der Benutzung von Computern deutlich verringert. Wigdor D. & Wixon D. [12] gehen davon aus, dass Natural-User-Interfaces (NUIs) den nächsten Schritt in der Entwicklung der Interaktion mit Computer System darstellen. Mit Hilfe von NUIs sollen vorhandenen Hürden weiter verringert werden und gleichzeitig den Usern mehr Möglichkeiten geboten werden.

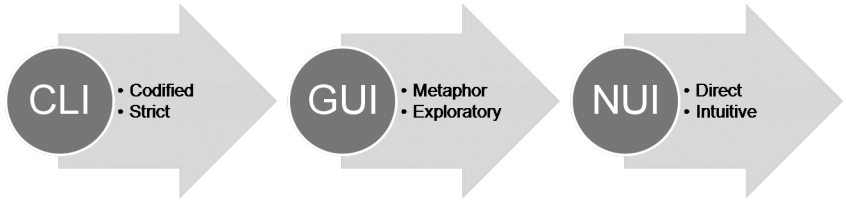

Abbildung 2.1: Die Entwicklung des User-Interface vom CLI über das GUI hin zum NUI [13]

2.3 Natural-User-Interface (NUI)

Ein NUI ist ein Interface dessen Ziel es ist, sich bei der Benutzung natürlich und intuitiv anzufühlen. Natürlich beschreibt hier nicht das Interface selbst, sondern bezieht sich hauptsächlich auf die Interaktion damit. In den letzten Jahren wurden viele Technologien entwickelt, die eine brauchbare Alternative zur Eingabe mittels Tastatur oder Maus darstellen. Touchscreens, Spracherkennung oder Bewegungserkennung sind ein paar der mittlerweile populäreren Beispiele. Diese Technologien erleichtern das Gestalten von NUIs, sind aber selbst noch keine Garantie für ein gelungenes NUI. Welche Punkte beim Erstellen eines NUI beachtet werden müssen, beschreiben Wigdor D. & Wixon D. in ihren Design-Richtlinien [12]:

- *"Create an experience that, for expert users, can feel like an extension of their body."*
- *"Create an experience that feels just as natural to a novice as it does to an expert user."*
- *"Create an experience that is authentic to the medium—do not start by trying to mimic the real world or anything else."*
- *"Build a user interface that considers context, including the right metaphors, visual indications, feedback, and input/output methods for the context."*
- *"Avoid falling into the trap of copying existing user interface paradigms."*

Ein wichtiges Kriterium für NUI ist also, dass das Interface für Anfänger zugänglich ist und mit wenig bis keiner Einarbeitungszeit benutzbar ist. Diese Zugänglichkeit soll durch die Natürlichkeit der Interaktion geschaffen werden. Um dies zu erreichen, kann beispielsweise auf Erfahrungen der User, die sie bei anderen Aktivitäten oder der Interaktion mit anderen User-Interfaces

gesammelt haben, aufgebaut werden [14]. Gleichzeitig sollen NUIs auch für Experten nützlich sein.

Direkte Manipulation ist ein Begriff, der im Zusammenhang mit Computersystemen bereits 1983 auftauchte [15]. Die direkte Manipulation von grafischen Objekten erfolgte aber bis vor wenigen Jahren noch mit Hilfe der Maus und anderen Eingabegeräten. Durch den Touchscreen wurden die Eingabegeräte durch den Finger ersetzt. Diese Entwicklung ermöglicht wirkliche direkte Manipulation und zusätzlich ergibt sich durch das Verschwinden der Eingabegeräte eine neue Nähe zum Interface. Diese natürliche Nähe ist beim Designen eines NUI sehr hilfreich und auch im Zusammenhang mit dem Annotieren ergeben sich dadurch Vorteile. Bei Texten werden zum Beispiel Notizen oft direkt neben oder über einer bestimmten Textpassage erstellt und ergeben nur in diesem Kontext einen Sinn. Erstellt man ein digitales Interface zum Annotieren, sollte man darauf achten, dass die Möglichkeit des Annotierens im dazugehörigen Kontext gegeben ist und ein Touchscreen bietet gute Voraussetzungen dafür.

Beim Designen von Interfaces für Touchscreens ist es wichtig dass man sich dieser Möglichkeiten bewusst wird und nicht einfach auf WIMP GUIs, die für den Einsatz mit der Maus entwickelt wurden, zurückgreift.

2.4 Touchscreens

Wie bereits erwähnt gibt es inzwischen eine Vielzahl von Alternativen zu den klassischen Eingabemöglichkeiten mittels Maus und Tastatur. Keine dieser Alternativen hat sich aber in den letzten Jahren so stark verbreitet wie der Touchscreen. Seit der Einführung des iPhones (2007) stieg die Anzahl an Smartphones, Tablets und Laptops mit integriertem Touchscreen kontinuierlich und die Interaktion mit Touchscreens gehört mittlerweile zum Alltag. Diese rasche Verbreitung fußt auf jahrelanger Forschung und Weiterentwicklung in diesem Bereich. Die ersten Versuche mit Touchsensoren wurden sogar schon vor der Entwicklung von Computern durchgeführt. Die Sensoren wurden zur Erzeugung von Elektroakustischer Musik mittels Synthesizer eingesetzt. Eines der ersten Touchscreen Interfaces wurde 1965 von E.A Johnson entwickelt. Er versuchte damit die komplexen Aufgaben, die bei der Luftraumkontrolle auftreten, zu vereinfachen [16]. Schon damals basierte der beschriebene Touchscreen auf kapazitiven Sensoren, die gleiche Technologie, die man heute in vielen Geräten findet.

Der nächste bedeutende Schritt in der Evolution von Touch-Geräten war das Ermöglichen von Multi-Touch Eingabe. Das gleichzeitige Erfassen von mehreren Touch-Punkten ermöglicht komplexere Interaktionen. Ein einfaches Beispiel dafür, welche Art von Interaktionen durch Multi-Touch möglich ist, zeigt bereits eine herkömmliche Tastatur. Durch das gleichzeitige Drücken der Shift-Taste und einem Buchstaben, wird der Buchstabe groß geschrieben und der Schreibfluss somit nicht unterbrochen. Schon 1982 beschrieb Nimish Mehta in seiner Arbeit “A Flexible Machine Interface” [17] ein System, das die Eingabe mittels Multi-Touch ermöglich-

te. In den nächsten Jahren wurde nicht nur die Hardware weiter entwickelt, sondern auch die Auswirkung von Touch und Multi-Touch Systemen auf die HCI weiter untersucht. Eine der einflussreichsten Arbeiten auf diesem Gebiet stammt von J.Y. Han. Dieser veröffentlichte 2005 ein Multi-Touch fähiges System basierend auf frustrierter Totalreflexion (FTIR) [18]. Das Besondere daran war, neben der einfachen, günstigen und skalierbaren Hardware, die Software und die damit verbundenen Interaktionstechniken, die er ein Jahr darauf im Zuge eines TED Talks [19] präsentierte. So wurde gezeigt wie Fotos skaliert, rotiert und gezoomt werden können, ähnlich wie es auf aktuellen Tablets und Smartphones möglich ist. Außerdem präsentierte er auch Interaktionsmöglichkeiten zum Skizzieren, Animieren und dem Formen von Gegenständen ähnlich der Bildhauerei.

B. Buxton hat sich intensiv mit der Geschichte der Entwicklung von Touch und Multi-Touch Hardware auseinandergesetzt und einen umfassenden Überblick erarbeitet [20]. Welche Herausforderungen und Möglichkeiten sich durch den Einsatz von Touchscreens ergeben beschreibt der nächste Abschnitt.

Herausforderungen und Eigenschaften von Touchscreens

Touchscreens ermöglichen neue Arten von Interfaces und neue Formen der Interaktionsgestaltung. Viele der traditionellen Paradigmen der HCI, die mittlerweile seit Jahren genutzt werden, müssen dazu neu überdacht werden. Cut & Paste, Fenster, die Desktop Metapher, Speichern und viele weitere Konventionen die in den 60er und 70er Jahre entwickelt wurden, sind für uns mittlerweile selbstverständlich. [8]. Ob und wie diese Paradigmen auf Touchscreen Interfaces zu transferieren sind, muss genau untersucht werden. Außerdem ist es wichtig, sich auch gänzlich von den Denkmustern dieser traditionellen Methoden lösen zu können, um neuen Ideen im Interaktionsdesign Raum zu geben. Um Touchscreen Interaktion zu verstehen, sollte man sich der wesentliche Herausforderungen und Eigenschaften von Touchscreens bewusst sein.

Eine bereits erwähnte Besonderheit von Touchscreens ist, dass die Eingabe direkt mit einem Finger getätigt werden kann. Die zwei meist verbreiteten Bauweisen sind resistive bzw. kapazitive Touchscreens. Der wesentliche Unterschied dabei ist, dass resistive Touchscreens auf Druck reagieren. Sie bestehen aus zwei dünnen, leitfähigen Folien, zwischen denen ein minimaler Abstand ist. Durch die Berührung des Touchscreens berühren sich diese Folien und es kann aufgrund von Spannungsunterschieden der genaue Berührungspunkt ermittelt werden. Im Gegensatz dazu basiert die Funktion von kapazitive Touchscreens auf der Veränderung eines elektrischen Feldes. Der menschliche Körper leitet Storm und kann so dieses Feld durch Berührung verändern und somit ist wiederum möglich, die Position der Berührung zu berechnen. kapazitive Touchscreens können deshalb beispielsweise nicht mit Handschuhen oder einem Eingabestift aus Plastik bedient werden. Allerdings gibt es spezielle Stifte, die auch mit kapazitiven Touchscreens funktionieren.

Durch die Möglichkeit der direkten Interaktion mittels Finger fällt eine Ebene der Interaktion weg und ermöglicht, wie bereits erwähnt, eine natürlichere Interaktion. Man sollte sich allerdings bewusst sein, dass der Finger nicht immer die beste Eingabemethode ist. Man denke nur daran, dass wir zum Schreiben prinzipiell Stifte und nicht unsere Finger verwenden. Ein Stift ermöglicht eine viel genauere Eingabe, als der Finger. Besonders im Zusammenhang mit dem Annotieren ist dies zu beachten. Wie genau die Eingabe für gewisse Interaktionen sein muss, ist auch ein wichtiges Kriterium beim Erstellen von Interfaces für Touchscreens. Die Berührung eines Button muss genau an einem definierten Bereich stattfinden. Eine Aktion, die eine noch genauere Eingaben benötigt ist zum Beispiel das Markieren eines einzelnen Buchstabens. Das Ausführen solcher Interaktionen mit dem Finger ist problematisch. Auf der einen Seite, weil der Finger selbst eine gewisse Größe hat und dadurch eine Ungenauigkeit entsteht, auf der anderen Seite, weil die meisten Touchscreens kein taktiles Feedback liefern.

Für viele der heute gängigen Interaktionen mit Touchscreens wird oft nur ein Finger benötigt. Allerdings besitzt der Mensch mehr als nur einen Finger und ist daran gewöhnt, diese im Alltag zu benutzen. Multi-Touch Touchscreens ermöglichen das gleichzeitige Erkennen von mehreren Berührungspunkten. Dadurch ist es möglich komplexere Interaktionen durchzuführen. Durch die Interaktion mit mehreren Fingern erhöht sich auch gleichzeitig die Anzahl der Freiheitsgrade. Die Maus ermöglicht die Bewegung des Mauszeigers auf einer 2D Ebene, das bedeutet eine Interaktion in 2 Freiheitsgraden. Können nun zwei Finger gleichzeitig verwendet werden, so wird auch die Anzahl der Freiheitsgrade verdoppelt. Neben mehreren Fingern ist grundsätzlich auch die Interaktion mit mehr als einer Hand möglich. Wir sind es bereits gewöhnt Aufgaben mit zwei Händen zu verrichten. Die Eingabe mit zwei Händen zu ermöglichen macht natürlich erst ab einer gewissen Größe des Touchscreens Sinn. Außerdem sollte man beachten, ob der Touchscreen auf irgendeine Weise fixiert ist, oder ob bereits eine Hand zum Halten des Geräts benötigt wird. Die Größe des Touchscreens ist generell ein wichtiger Faktor und muss beim Design der Interaktionsmöglichkeiten beachtet werden. Auch das Format spielt eine Rolle, da Touchscreens vorwiegend nicht quadratisch sind, gibt es meistens eine horizontale und vertikale Orientierung.

Wird ein Touchscreen mit einem Finger oder Stift berührt, so wird erkannt an welcher Position die Berührung stattgefunden hat. Neben der Position gibt es allerdings noch andere Faktoren, die je nach Hardware erkannt werden können und für mögliche Interaktionen von Bedeutung sind. Es gibt Touchscreens die druckempfindlich sind und zusätzlich zu den Koordinaten auch die stärke des Touches ermitteln. Dafür wird häufig auch einfach die Größe des Touches verwendet. Es wird angenommen, dass je größer die Berührungsfläche des Fingers ist, desto fester wurde der Touchscreen berührt. Diese zusätzlichen Parameter können beispielsweise verwendet werden um die Strichstärke bei einer Anwendung zum Zeichnen zu variieren. Durch die Bewegung auf einem Touchscreen können außerdem Bewegungsvektoren berechnet werden. Diese Vektoren können ähnlich wie der ausgeübte Druck dazu genutzt werden, um weitere

Parameter zu generieren. Dadurch kann unterschieden werden, ob die Eingabe langsam oder schnell stattfindet.

Bei der Eingabe kann generell zwischen diskreter und kontinuierlicher Eingabe unterschieden werden. Der Touch auf einen Button oder die Eingabe mittels eines On-Screen-Keyboards stellt eine diskrete Interaktion dar. Diskrete Interaktionen besitzen meistens einen grafischen Hinweis. Im Gegensatz dazu gibt es kontinuierliche Gesten. Diese besitzen oft keinen Hinweis am Bildschirm und der User muss bereits wissen, dass es die Möglichkeit gibt diese Geste auszuführen. Ein Beispiel dafür ist das Durchblättern von Fotos durch Wischgesten nach links oder rechts.

Dieser Abschnitt hat gezeigt, dass Touchscreen nicht gleich Touchscreen ist und es viele Faktoren gibt, dessen man sich bewusst sein muss, wenn man Interaktion für Touchscreen-Geräte gestaltet. Im Folgenden sollen einige dieser Herausforderungen anhand eines Beispiels greifbar gemacht werden.

Fallbeispiel: Vom Touch zur Linie auf dem iPad

Um das Schreiben oder Markieren auf Tablets zu ermöglichen, muss die analoge Bewegung des Fingers auf dem Touchscreen in eine digitale, sichtbare Linie umgewandelt werden. Auch wenn die Frameworks der aktuellen mobilen Betriebssysteme dem Entwickler schon vieles an Arbeit abnehmen, um dies zu ermöglichen, gibt es trotzdem noch einige Faktoren, die beachtet werden müssen. In diesem kurzen Fallbeispiel wird beschrieben, welche Schritte notwendig sind, um eine kontinuierliche Bewegung auf dem Touchscreen des iPads mit Hilfe des iOS-SDKs in eine Linie umzuwandeln.

Abbildung 2.2a zeigt die kontinuierliche Bewegung des Fingers, wie sie am Touchscreen stattgefunden hat. Um aus dieser Bewegung eine Linie zu erzeugen, wird die analoge Eingabe in digitale Daten umgewandelt. Dazu wird die jeweilige Position der Berührung zu gewissen Zeitpunkten gemessen, in Bildschirmkoordinaten umgewandelt und gespeichert. Allerdings ist die zeitliche Auflösung begrenzt und man erhält nur ein paar Punkte, die auf der Linie liegen (Abbildung 2.2b). Diese zu geringe Auflösung ist das Hauptproblem bei der Umwandlung in eine Linie und macht die exakte Darstellung der Eingabe unmöglich. Der einfachste Weg, um die

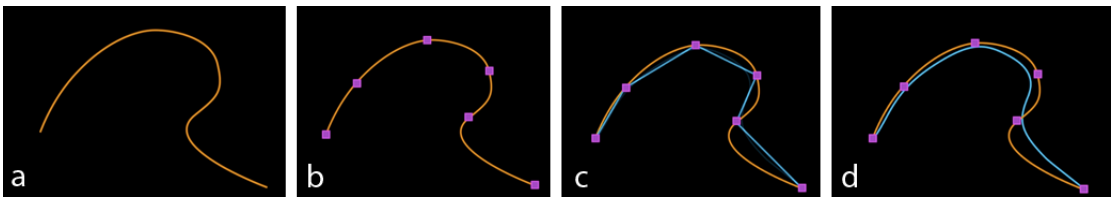

Abbildung 2.2: Vom Touch (orange) zur digitalen Linie (blau) [WWDC 2012 Session 233]

durch die geringe Auflösung verlorene Punkte wieder herzustellen, ist die Abtastpunkte einfach mit einer gerade Linien zu verbinden (Abbildung 2.2c). Das führt allerdings zu einer eckigen Linie und der Schwung der ursprünglichen Eingabe geht verloren. Das Ergebnis ist bei dieser Variante umso schlechter, desto schneller die Eingabe erfolgt ist. Eine Möglichkeit, um das Ergebnis zu verbessern, ist die Verwendung von Bézierkurven. iOS bietet mit Core Graphics eine API, welche das schnelle Zeichnen anhand von Pfaden ermöglicht. Mit Hilfe dieser API können außerdem quadratische Bézierkurven erstellt werden. Eine quadratische Bézierkurve besteht aus einem Start-, einem End-, sowie einem Kontrollpunkt. Wichtig ist die richtige Auswahl dieser Punkte. Es wird jeweils der Mittelpunkt zwischen zwei Abtastwerten berechnet. Diese Mittelpunkte werden als Start- und Endpunkte der quadratischen Bézierkurve verwendet und die eigentlichen Abtastwerte als Kontrollpunkte. Auch wenn dadurch die Linie nie direkt durch die, vom iOS-SDK als Touch Events gelieferten Abtastpunkte verläuft, liefert die Methode eine gute und geglättete Annäherung (Abbildung 2.2d).

Schematisch zusammengefasst enthält der Algorithmus zur Umwandlung von Touch-Events in eine Linie folgende Schritte:

- Es werden jeweils die letzten 3 Touches gespeichert **touch1, touch2, touch3**. Wobei **touch1** die aktuellste Position beinhaltet.

- *touchesMoved* - Event wird vom iOS-SDK ausgelöst. Das heißt es hat eine Berührung + Bewegung am Touchscreen stattgefunden. Das Event liefert die x,y Koordinaten der aktuellen Position (*locationInView*), sowie die vorherige Position (*previousLocationInView*) zurück.

- Die aktuellen Punkte werden gespeichert
  touch3 = touch2
  touch2 = previousLocationInView
  touch1 = locationInView

- Berechnung der Mittelpunkte zwischen den 3 Berührungspunkten
  mid2 = (touch2 + touch3) / 2
  mid1 = (touch1 + touch2) / 2

- Danach kann ein Pfad mittels einer quadratischen Bézierkurve vom Mittelpunkt mid2, zum Mittelpunkt mid1 mit Hilfe des Kontrollpunktes touch2 erstellt werden.
  CGPathMoveToPoint(path, NULL, **mid2**.x, **mid2**.y);
  CGPathAddQuadCurveToPoint(path, NULL, **touch2**.x, **touch2**.y, **mid1**.x, **mid1**.y);

- Nun muss der Pfad noch mit der *CGContextStrokePath* Methode gefüllt werden. Dabei können die Linienfarbe und Linienstärke festgelegt werden.

Der Touchscreen des iPads hat keinen Drucksensor eingebaut. Aus diesem Grund kann nicht festgestellt werden, mit welchem Druck der Strich gezeichnet wurde. Außerdem ist es mit dem aktuellen SDK auch nicht möglich, die Größe eines Touchevents abzufragen. Es ist allerdings wichtig, die Strichstärke der Linie zu variieren, um einen realistischen Eindruck zu erzeugen. Eine Möglichkeit eine variierende Strichstärke zu simulieren ist es, die Strichstärke von der Geschwindigkeit der Touch Eingabe abhängig zu machen. Desto schneller die Linie gezeichnet wurde, desto dünner ist die Strichstärke. Dieses Fallbeispiel demonstrierte die Simulation eines Stiftes auf dem iPad. Das nächste Kapitel beschreibt, welche weiteren Möglichkeiten die Touch-Eingabe mittels Gesten noch liefert.

2.5 Interaktion mittels Gesten

Beschäftigt man sich mit Interfaces abseits der Desktop Metapher und den WIMP Paradigma kommt man an auf Gesten basierender Interaktion nicht vorbei. Der Begriff Geste ist sehr breit gefächert und hat je nach Einsatzgebiet leicht unterschiedliche Bedeutungen. Menschen sind mit verschiedenen Formen von Gesten vertraut und verwenden Gesten als Teil ihrer Kommunikation. Durch die Vertrautheit mit Gesten, stellen diese eine sehr gute Option für NUIs dar. Die Interaktion mittels Gesten im Zusammenhang mit Computersystemen wurde im HCI-Bereich schon sehr früh erforscht. Eine weite Verbreitung erlangte sie allerdings erst mit der Einführung von auf Touchscreen basierenden Smartphones und Tablets. Grundsätzlich kann zwischen zwei Arten von Gesten unterschieden werden. Bei Motion-Gesten wird die Bewegung einer Person oder eines Gegenstandes im dreidimensionalen Raum getrackt und als Geste interpretiert. Eine mögliche Geste wäre zum Beispiel das Winken mit einer Hand. Die zweite Art von Gesten sind Surface-Gesten. Surface-Gesten werden auf einer Touchscreen-Oberfläche mit Hilfe der Finger oder eines Stiftes ausgeführt. Surface-Gesten sind somit rein auf den zweidimensionalen Raum beschränkt. Diese Arbeit beschäftigt sich weiterführend ausschließlich mit Surface-Gesten, welche in weiterer Folge nur noch als Gesten bezeichnet werden.

Vom Standpunkt der Informationstheorie ausgehend, kann eine Geste als Signal vom User zum Computersystem betrachtet werden [21] (Abbildung 2.2). Der Touchscreen ist dabei der Kanal über den die Geste übertragen wird. Die Übertragung kann durch Ungenauigkeiten bei der Eingabe durch den User, sowie durch die limitierten Abtastungsmöglichkeiten der Hardware gestört werden. Danach findet die Auswertung der Daten und die Gestenerkennung statt. Schlussendlich wird dadurch die mit der jeweiligen Geste verbundene Aktion ausgeführt.

Eigenschaften von Gesten

Gesten haben einige wichtige Parameter, deren man sich bei der Erstellung von auf Gesten basierender Interfaces bewusst sein sollte. Je nach der verwendeter Hardware und Plattform sind

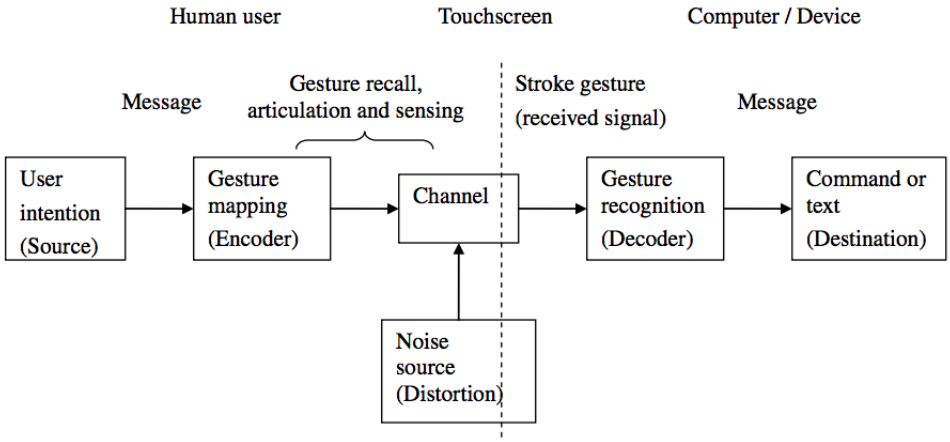

Abbildung 2.3: Modell zur Eingabe von Gesten aus der Sicht der Informationstheorie [21]

nicht immer alle Parameter vorhanden oder zugänglich. Das Interface ist umso anspruchsvoller und umfangreicher, desto mehr dieser Parameter in das Design einfließen [8].

Der grundlegendste Parameter, der für eine Geste benötigt wird, ist ob der Finger oder Stift den Touchscreen berührt. Eine Berührung ist der Beginn jeder Geste. Zusätzlich kann festgestellt werden, wie viele Berührungspunkte vorhanden sind. Berührt nur ein Finger den Touchscreen, oder sind es mehrere? Manche Systeme können außerdem unterscheiden, ob die Berührung von einer, oder mehreren Personen, oder einem Objekt stammt. Wurde ein Touch erkannt, wird außerdem die Information geliefert, wo genau am Touchscreen die Berührung stattgefunden hat. Neben der x/y-Position des Touches können noch weitere Parameter verfügbar sein - die Größe des Touches, der Druck mit dem der Touch ausgeführt wurde oder die Orientierung des Touches. Ein weiterer wichtiger Faktor, der beim Design von Gesten benötigt wird, ist die Zeitdauer. Damit kann zwischen schnellen/kurzen oder langen/langsamen Berührungen unterschieden werden. Neben diesen essentiellen Parametern, besitzen Gesten noch einige weitere Eigenschaften [21].

Es kann zwischen analogen und abstrakten Gesten unterschieden werden. Analoge Gesten basieren auf physischen Abläufen und auf aus der realen Welt bekannten Effekten, wohingegen abstrakte Gesten keine dieser Konventionen imitieren. Allerdings bestehen viele Gesten aus einer Mischung zwischen analogen und abstrakten Komponenten. Grundsätzlich sind analoge Gesten natürlicher und dadurch leichter zu erlernen, wohingegen die Ausführung von abstrakten Gesten oft einfacher und weniger komplex ist.

Dadurch ergibt sich eine weitere Einteilung von Gesten in ihre Level der Komplexität. Eine einfache Berührung des Touchscreens ohne Bewegung ist eine Geste der nullten-Ordnung. Ein Beispiel für eine solche Geste ist der einfache Touch auf einen Button oder auf ein Textfeld.

Gesten, die nur aus einem einzelnen Strich bestehen, werden als Gesten der ersten-Ordnung bezeichnet. In diese Kategorie fällt zum Beispiel die bekannte Wischgeste. Sobald eine Geste aus mehreren Bewegungen besteht, spricht man von einer Geste höherer Ordnung. Das kann entweder gleichzeitig mit Hilfe von Multitouch passieren, oder hintereinander.

Zusätzlich kann zwischen ortsunabhängigen und ortsabhängigen Gesten unterschieden werden. Eine Geste wird als ortsunabhängig bezeichnet, wenn sie, egal wo sie am Touchscreen ausgeführt wurde, die selbe Aktion auslöst. Dadurch ergibt sich zusätzlich eine visuelle Unabhängigkeit. Der User muss nicht zuerst ein gewisses Objekt am Bildschirm suchen sondern kann die Geste direkt ausführen. Auf der anderen Seite haben ortsabhängige Gesten den Vorteil, dass sie meisten mehr Information enthalten. Je nachdem wo die Geste ausgeführt wird, wird eine andere Aktion ausgelöst. Zusätzlich ist die Geste oftmals durch den Ort, an dem sie ausgeführt wird, direkt an ein Objekt gekoppelt. Einfache Gesten der nullten-Ordnung, sind meist ortsabhängig. Allerdings ist die Unterscheidung zwischen ortsabhängigen und ortsunabhängigen Gesten oft nicht eindeutig. Viele Gesten sind nur zu einem gewissen Grad ortsunabhängig.

Design von Gesten

Gesten sollen dabei helfen, die Interaktion zu vereinfachen und zu verbessern. Damit das möglich ist, muss die zur Interkation verwendete Geste sorgsam gewählt und designt werden. Das Designen einer Geste ist kein triviales Unterfangen und hängt von verschiedenen Faktoren ab. Die im letzten Abschnitt angeführten Eigenschaften und Parameter bilden einen guten Ausgangspunkt und sollen nun durch zusätzliche Überlegungen ergänzt werden.

Gesten sollten so einfach wie möglich gestaltet werden. User bevorzugen im Allgemeinen einfachere Gesten [22]. Einfach bedeutet in diesem Zusammenhang, dass die Geste einerseits physisch leicht ausführbar ist und andererseits einen geringen kognitiven Aufwand mit sich bringt. Gesten mit einem Finger sind einfacher auszuführen und beliebter, als Gesten bei denen man mehrere Finger, die ganze Hand oder sogar beide Hände benötigt. Außerdem werden im Gegensatz zu abstrakten Gesten oder Gesten die auf Metaphern basieren, Gesten bevorzugt, die auf physischen Konzepten oder bekannten Symbolen basieren. In einer der Designrichtlinien für NUIs heißt es, dass es ein Ziel ist, eine User-Experience zu erschaffen die sich für Anfänger und Experten natürlich anfühlt. Um dies zu gewährleisten, ist es wichtig, dass sich die Geste vorausahnen lässt und offensichtlich ist. Eine Möglichkeit diese Eigenschaften zu erreichen, ist die Verwendung von analogen Gesten. Um beispielsweise die Lautstärke zu erhöhen, kann eine Bewegung nach oben verwendet werden und um die Lautstärke zu verringern eine Bewegung nach unten. Solche Interaktionen funktionieren deshalb sehr gut, weil sich unsere Denkweise stark an physischen Modellen orientiert [23]. Eine andere Möglichkeit zum Erstellen von leicht erlernbaren Gesten ist die Verwendung eines userzentrierten Ansatzes. Wobbrock et al. [24] haben in ihrer Arbeit die Aktionen, die eine Geste auslösen sollen vorgegeben und mehrere User

unabhängig voneinander eine dazu passende Geste ausführen lassen. Dabei konnte festgestellt werden, dass die Befragten, bei manchen Aktionen, gleiche oder sehr ähnliche Gesten verwendeten. Durch diese Übereinstimmung, kann davon ausgegangen werden, dass speziell diese Gesten leicht erlernbar sind bzw. ohne besondere Vorkenntnisse verwendet werden können. Verwendet man einen solchen, auf User basierenden, Ansatz ist es notwendig die Testgruppe gezielt auszuwählen. Alter, technische Erfahrung und ähnliche Eigenschaften können zu sehr unterschiedlichen Ergebnissen führen. Die Anzahl der intuitiven und leicht verständlichen Gesten ist allerdings beschränkt und kann in gewissen Situationen nicht ausreichend sein. In diesen Fällen gibt es verschiedene Ansätze, um die Gesten möglichst gut zu kommunizieren und das Erlernen von Gesten zu erleichtern. Die einfachste Möglichkeit ist das Anzeigen einer Übersicht aller möglichen Gesten, deren Auswirkung und einer Beschreibung oder Animation, wie die Geste ausgeführt werden muss. In der Literatur wird diese Variante als Crib-Sheet bezeichnet [25].Eine solche Übersicht ist zwar eine gute Referenz, allerdings oft nicht ausreichend um die Charakteristik einer Geste und den Kontext in dem sie verwendet werden kann zu kommunizieren. GestureBar [26] und OctoPocus [27] sind zwei Lösungsansätze die sich mit dieser Problematik befassen.

Neben der Einfachheit und leichten Erlernbarkeit von Gesten, ist es ebenfalls wichtig, dass sich Gesten visuell nicht zu ähnlich sind. Zu ähnliche Gesten können zu einer höheren Fehlerrate bei der Gestenerkennung führen. Dahingegen haben Gesten, die sich eindeutig unterscheiden, eine bessere Erkennungsgenauigkeit und lassen sich außerdem leichter einprägen. Long et al. [28] hat Gesten bezüglich ihrer visuellen Ähnlichkeit untersucht und dadurch ein Modell zum Vorhersagen der durch den User wahrgenommen Ähnlichkeit erstellt. In diesem Zusammenhang wurde auch das Gesten Design Tool quill [29] entwickelt, welches dabei helfen soll, Gesten zu erstellen und zu optimieren. Es gibt auch noch weitere Toolkits, die beim Designen von Gesten behilflich sind. Einiger dieser Toolkits bieten neben Algorithmen zum Erkennen von Gesten und Erstellen von Trainingsdaten zusätzlich die Möglichkeit, Gesten auf motorische und visuelle Eigenschaften zu prüfen.

Verbreitete Gesten und Interaktionen

Ein wesentlicher Faktor bei der Entwicklung von gelungenen User-Interfaces ist Konsistenz. Im Zusammenhang mit Gesten bedeutet das, dass es nicht ausreicht die einzelnen Gesten selbst gut zu designen, sondern dass es auch notwendig ist, dass sie im Zusammenspiel und somit als Gesamtsystem gut funktionieren. Viele der traditionellen Interfacekonventionen wie das Auswählen, Drag & Drop oder Scrollen funktionieren auch auf Touchscreens sehr gut oder teilweise sogar besser als auf herkömmlichen Computersystemen [8]. Es gibt aber auch bekannte Interaktionen, die durch die Gestensteuerung nicht mehr funktionieren oder nicht mehr benötigt werden. Beispielsweise wird der von WIMP-Interfaces bekannte Cursor, der zur Visualisierung

der aktuellen Mausposition dient, nicht mehr benötigt, da das Interface direkt mit dem Finger berührt wird. Ein anderes Beispiel ist der Hover oder Mouseover-Effekt, bei dem eine Aktion rein durch die Positionierung des Cursors hervorgerufen wird. Diese Interaktion funktioniert bei Geräten mit Touchscreen deshalb nicht, weil die Position des Fingers nur dann erfasst werden kann, wenn der Finger den Touchscreen berührt.

Durch die Verbreitung von Smartphones und Tablets haben sich mittlerweile einige grundlegende Gesten und Interkationen systemübergreifend herauskristallisiert. Dabei wird darauf geachtet, dass diese Interaktionen konsistent verwendet werden. So enthalten die Dokumentationen für Entwickler für Android und iOS Konventionen dafür welche Geste welche Aktion auslösen soll [30, 31]. Die folgende Aufzählung bietet einen Überblick über die gängigsten Gesten, sowie die damit verbundenen Entwurfsmuster.

- **Tap / Touch** - Ein Tap ist das einfache, kurze Berühren des Touchscreens mit einem Finger. Die Berührung dauert weniger als 500 ms und kann als Äquivalent zum Mausklick gesehen werden. Die Tap-Geste wird zum Aktivieren oder Auswählen von Elementen verwendet. Im Gegensatz zu Mausklicks wird bei Taps die jeweilige Aktion erst dann ausgeführt, wenn der Finger den Touchscreen wieder verlassen hat. Damit besteht die Möglichkeit den Finger noch vom jeweiligen Element zu ziehen und damit die Aktion abzubrechen. Neben dem einfachen Tap gibt es noch die Gesten **Double-Tap** und **Long-Press / Hold-Geste**. Bei der Double-Tap-Geste werden zwei Taps schnell hintereinander ausgeführt. Diese Geste wird zum Zoomen oder zum Zentrieren von Inhalten verwendet. Die Long Press Geste ist ein Tap der länger als 500ms dauert und wird für verschiedene Aktionen eingesetzt. Zum Beispiel wird damit auf iOS eine Lupe aktiviert mit deren Hilfe sich der Textcursor genauer positionieren lässt. Bei Android kann durch eine Long Press Geste, je nach ausgewählten Element, ein vom Kontext abhängiges Menu aufgerufen werden.

- **Drag / Slide / Swipe** - Die Drag-Geste wird durch Berühren eines Objektes am Touchscreen und gleichzeitiges Bewegen in eine Richtung ausgeführt. Sie wird verwendet, um zu scrollen oder um einzelne Objekte zu bewegen. Objekte können dadurch mit dem Finger von einer Stelle zur anderen geschoben werden. Die Geste wird oft auch als Swipe-Geste bezeichnet, wenn sie nicht auf einem bestimmten Objekt und horizontal oder vertikal ausgeführt wird. Die Swipe-Geste wird verwendet, um eine Seite umzublättern oder zwischen verschiedenen Bildschirmen zu wechseln.

- **Flick / Fling** - Als Flick bezeichnet man eine schnell ausgeführte Drag-Geste. Die Geste wird durch ein schnelles Wischen in eine Richtung am Touchscreen ausgeführt und muss nicht zwingend auf einem bestimmten Objekt durchgeführt werden. Meistens wird ein

Flick verwendet um schnell durch mehrere Elemente zu navigieren oder um schneller durch eine lange Liste zu scrollen.

- **Pinch in or out / Pinch open or close** -Alle bisherigen Gesten sind in ihrer Grundform Single-Touch Gesten. Die Pinch-Geste ist eine Multitouch Geste, die durch das Berühren und gleichzeitige auseinanderziehen bzw. zusammenführen von zwei Fingern ausgeführt wird. Sie wird zum Zoomen von Inhalten oder zum Vergrößern/Verkleinern von Objekten eingesetzt. Im Gegensatz zur Double-Tap-Geste kann hier ein stufenloser Zoom realisiert werden.

2.6 Unterschiede zwischen der Eingabe mittels Finger und Stift

Bisher ist das Hauptaugenmerk dieser Arbeit auf der direkten Interaktion mit dem Touchscreen mittels Finger gelegen. Der Grund dafür liegt darin, dass die Entwicklung der letzten Jahre deutlich in die Richtung von Geräten geht, die auf kapazitiven Touchscreens basieren und meist ausschließlich mit Hilfe der Finger bedient werden. Viele der frühen mobilen Geräte verwendeten allerdings einen Stift (auch Stylus) als primäre Methode zur Eingabe. Ein bekanntes Beispiel dafür sind PDAs, die meistens mittels eines Stiftes bedient wurden. Durch das Weglassen des Stiftes als Eingabegerät entstehen allerdings gewisse Vorteile. Auf der einen Seite ist die Interaktion dadurch natürlicher und direkter als mit einem Stift. Vor allem unter den Kriterien eines NUI betrachtet, ist diese Reduktion ein logischer Schritt. Auf der anderen Seite ist der Stift ein zusätzlicher Gegenstand, der zur Benutzung des jeweiligen Geräts benötigt wird und somit immer mitgeführt werden muss. Das schränkt zum einen die Mobilität ein und zum anderen kann das Gerät nicht mehr benutzt werden wenn der Stift verloren oder vergessen wird.

Der Finger als Eingabegerät ist natürlicher, direkter und für viele Anwendungsfälle sehr gut geeignet, allerdings nicht für alle. Darum ist es wichtig, sich der Interaktiongestaltung für Tablets der Unterschiede zwischen der Eingabe mittels Finger und der Eingabe mittels Stiftes bewusst zu sein. Tu et al. [32] haben die Unterschiede anhand der Ausführung von unterschiedlichen Gesten verglichen. Dabei hat sich herausgestellt, dass es einige Eigenschaften gibt, bei denen große Unterschiede zu sehen sind und andere, bei denen sich Stift und Finger sehr ähnlich sind. Gesten die mit dem Finger ausgeführt werden, sind deutlich größer und werden im Allgemeinen schneller ausgeführt als Gesten mit einem Stift. Allerdings spielt bei der Ausführungsgeschwindigkeit die Komplexität der Geste eine entscheidende Rolle. Nur einfache Geste lassen sich deutlich schneller mit dem Finger ausführen. Der Stift zeigt seine Vorteile bei komplexeren Gesten und bei Aufgaben, die eine höhere Genauigkeit erfordern. Interessant ist auch, dass der Unterschied bei Kindern nicht in dieser Art vorhanden ist und es keine signifikanten Unterschiede zwischen der Eingabe mittels Stift oder Finger gibt [33]. Das zeigt, dass der Umgang mit einem Stift etwas ist, das erlernt und trainiert werden muss.

Betrachtet man diese Unterschiede im Kontext des Annotierens zeigt sich, dass der Stift für gewisse Aufgaben besser geeignet ist. Zum einen natürlich für das Erstellen von Notizen, zum anderen aber auch für das Markieren von bestimmten Stellen. Beides sind Aufgaben, bei denen die Genauigkeit wichtig ist. Meistens wird der Fokus beim Interaktionsdesign entweder auf die Eingabe mittels Stift oder Finger gelegt. Allerdings sollte auch der gleichzeitige Einsatz beider Eingabemöglichkeiten in Betracht gezogen werden, da sie sich gut ergänzen. Hinckley et al. [34] haben verschiedene Interaktionstechniken untersucht, bei denen Finger und Stift gleichzeitig verwendet werden.

> *"the pen writes, touch manipulates, and the combination of pen + touch yields new tools."* [34]

Viele dieser Techniken orientieren sich an unseren alltäglichen Gebrauch mit Stift und Papier. Dabei wird der Stift in der bevorzugten Hand gehalten und zum Schreiben oder Markieren verwendet. Die andere Hand kann für verschiedene manipulative Tätigkeiten, wie zum Beispiel das Verschieben oder Drehen von Objekten verwendet werden. Die Kombination von Finger und Stift bietet viele verschiedene Möglichkeiten zur Interaktion. Als problematisch kann gesehen werden, dass durch die Verwendung beider Hände die Möglichkeit wegfällt, den Tablet mit einer Hand zu halten. Vor allem in mobilen Einsatzszenarien kann dies zu Schwierigkeiten führen. Außerdem sind Interaktionen mit Stift und Finger meistens nicht so intuitiv und selbstverständlich wie einfache Gesten. Hinckley et.al. argumentieren allerdings, dass sich die Gesten, wenn sie erst mal gelernt wurden, durchaus natürlich anfühlen können. Ein anderer wichtiger Faktor dafür, ob die gleichzeitige Interaktion mit Finger und Stift Sinn macht, ist die Größe des Touchscreens.

2.7 Schwierigkeiten und Herausforderungen beim Designen für Touchscreens

Neben den vielen Potentialen die Natural-User-Interfaces und Touchscreens bieten, besitzen sie aber auch gewisse Schwächen. Dieser Abschnitt spricht einige dieser Schwierigkeiten und Herausforderungen an, denen man sich bei der Gestaltung von Interaktion abseits des Desktops bewusst sein sollte.

Einer der größten Kritikpunkte von Touchscreens ist das fehlende haptische Feedback. Touchscreen Interfaces sind im Prinzip nur flache Abbildungen mit denen interagiert werden kann. Feedback muss deshalb auf einem anderen Weg erfolgen und das geschieht zum Großteil auf visuelle Weise. Ein Touchscreen kann aus diesem Grund nur verwendet werden, wenn direkter Sichtkontakt besteht. Touchscreen Interfaces benötigen somit die komplette visuelle

Aufmerksamkeit. Nur so kann man beispielsweise wissen, wo genau sich ein Button am Touchscreen befindet oder ob eine Geste erfolgreich ausgeführt wurde. Neben der visuellen Aufmerksamkeit, werden auch meistens beide Hände benötigt, um einen Touchscreen zu bedienen. Wobei eine Hand das Gerät hält und somit nur noch eine Hand zum Interagieren übrig bleibt. Das ist vor allem bei Smartphones der Fall, aber kommt durchaus auch bei Tablets vor.

Durch die Eingabe mittels Finger wird oft ein Teil des Touchscreens verdeckt. Bei Tablets verdeckt nicht nur der Finger selbst Teile des Bildschirms, sondern auch die restliche Hand. Außerdem haben Finger von verschiedenen Personen, verschiedene Abmessungen. Kinder haben kleinere Finger als Erwachsene. Ältere Menschen sind unter Umständen nicht in der Lage gewisse Abläufe mit der selben Genauigkeit auszuführen, wie jüngere Personen. Zusätzlich gibt es Unterschiede zwischen Rechtshändern und Linkshändern.

Ein weiterer Kritikpunkt auf Gesten basierender Interaktion ist das Fehlen von Standards. So kann es vorkommen, dass die gleichen Gesten oder Interaktionen auf verschiedenen Geräten, oder verschiedenen Betriebssystemen unterschiedliche Aktionen hervorrufen. Auch, wenn sich die Gesten und die damit Verbundenen Aktionen teilweise ähnlich sind, sind das oft die Entscheidungen von individuellen Unternehmen, welche ihre eigenen Richtlinien festlegen.

Norman N. [35] kritisiert die existierenden, nicht allgemein geltenden Richtlinien und argumentiert, dass sie sich oft nicht an fundamentale Prinzipien des Interaktionsdesigns halten. Er führt dabei folgenden Prinzipien auf.

- *Visibility (also called perceived affordances or signifiers)*
- *Feedback*
- *Consistency (also known as standards)*
- *Non-destructive operations (hence the importance of undo)*
- *Discoverability: All operations can be discovered by systematic exploration of menus.*
- *Scalability: The operation should work on all screen sizes, small and large.*
- *Reliability: Operations should work. Period. And events should not happen randomly.*

Zusammenfassung

Durch die Evolution des User-Interfaces vom CLI über das GUI hin zum NUI sind spannende und neue Interaktionsmöglichkeiten entstanden. Ein NUI wir nur durch geschickte Designentscheidungen und Überlegungen auch wirklich zum *Natural* User-Interface, das für Anfänger sowie Experten von Nutzen ist. Der Touchscreen, als Hardwarekomponente gesehen, bietet durch die gewonnene Nähe zum User die ideale Voraussetzung für ein NUI und in weitere Folge für

ein App zum Annotieren von Videoinhalten. Die Möglichkeiten, die durch auf Gesten basierender Interaktion entstehen, sind erfrischend und oftmals intuitiv. Trotzdem ist es wichtig, auch die Limitierungen genau zu betrachten. Der Vergleich zwischen der Eingabe mittels Finger und der Eingabe mittels Stylus veranschaulicht wie sich unterschiedliche Varianten der Interaktion ergänzen können.

KAPITEL 3

Videos annotieren und mit Videos interagieren

Das Video ist eines der präsentesten Medien und hat sich in den letzten Jahren, nicht zuletzt durch die Digitalisierung, grundlegend weiter entwickelt. Die Hürden, um Videos zu produzieren, sind so gering wie noch nie. Nicht nur die Produktion ist vereinfacht worden, sondern, durch das Internet, auch die Möglichkeit Videos zu verbreiten. Eine Studie von CISCO [36] sagt voraus, dass im Jahr 2017 69% des weltweiten Internettraffics durch Videos verursacht wird (2012 waren es 57%). Auch die Geräte, auf denen Videos konsumiert werden, haben sich in den letzten Jahren verändert. Mit Hilfe von Smartphones und Tablets können Videos mittlerweile immer und überall angesehen werden. Vor allem Tablets sind zum Konsumieren von Videos sehr beliebt. comScore [37] hat dazu einen Studie veröffentlich und kommt zu dem Schluss, dass mehr als die Hälfte der Tablet User regelmäßig Videos konsumieren. Im Gegensatz dazu sind es bei den Smartphone-Usern nur 20%. Dieser Aufschwung bringt allerdings auch neue Herausforderungen mit sich. Dieses Kapitel beschäftigt sich mit dem Wandel von Videos und betrachtet ihn aus der Perspektive des Users. Wie hat sich sein Verhalten durch die neuen technischen Möglichkeiten verändert?

Ein weiteres Thema dieses Kapitels ist das Annotieren von Videos. Um sich mit den Inhalten von Videos auseinanderzusetzen, sollte es möglich sein, Videos, sowie Teile eines Videos, zu annotieren. Bei Texten ist das Annotieren kein Problem. Viele der gängigen Textverarbeitungs-Programme beinhalten geeignete Funktionen dafür. Für Videos hingegen existieren nur wenige, meist sehr komplexe Applikationen die das Annotieren erlauben. Es werden einige dieser Applikationen betrachtet, sowie die vielfältigen Einsatzmöglichkeiten und Potentiale der Videoannotation im Allgemeinen aufgezeigt. In weiterer Folge werden die Anforderungen an ein System zur Videoannotation besprochen. Danach werden automatisierte Ansätze aus dem Bereich der

Bildverarbeitung erörtert, welche die oft auch sehr zeitaufwändige Aufgabe der Videoannotation vereinfachen können. Außerdem werden Standards präsentiert, die es erlauben Annotationen auf eine kompatible Art abzuspeichern und so auszutauschen. Zum Abschluss werden zwei interessante Techniken vorgestellt, um mittels Tablet und einem Stylus mit Videos zu interagieren. Diese Möglichkeiten sollen zeigen, dass die im Moment gängigen User-Interfaces im Zusammenhang mit Videos durchaus verbessert und erweitert werden können.

3.1 Video im Wandel

Das Medium Video hat sich in den letzten Jahren sowohl durch technische Entwicklungen, als auch durch die Entstehung neuer Umgangsformen verändert. In der Vergangenheit gab es eine klare Abgrenzung zwischen Produzent und Konsument. Videos wurden von professionellen Produzenten hergestellt, die sich intensiv mit Themen wie Videodreh und Videoschnitt beschäftigten. Diese Videos wurden in weiterer Folge von anderen Personen konsumiert. Wobei der Konsum ein rein passives Verhalten darstellte. Diese klare Trennung ist in den letzten Jahren allerdings immer mehr verschwommen. Dafür gibt es verschiedene Gründe. Zum einen die Verbreitung günstiger und leistungsfähiger Hardware. Dazu zählen die zur Aufnahme von Videos nötigen Geräte wie Videokameras, Digitalkameras mit Videofunktion, Smartphones und Tablets, aber auch die Möglichkeit Videos digital auf Festplatten abzuspeichern. Zum anderen war die Weiterentwicklung des Internets für diese Entwicklung notwendig. Mittlerweile können Videoinhalte problemlos über das Internet ausgetauscht, publiziert und sogar in Echtzeit übertragen werden.

Juhlin et al. [38] beschreiben drei wichtige Trends, die für die Weiterentwicklung der Videointeraktion von Bedeutung sind. Der erste Trend ist die bereits angesprochene starke Bewegung vom reinen Konsum der User hin zur Eigenproduktion von Inhalten. Plattformen wie Youtube forcieren die Erstellung von eigenen Videos und bieten neben der einfachen Möglichkeit Videos zu veröffentlichen, auch ganz neue Wege, um Videoinhalte zu vermarkten. Außerdem sind die Produktionskosten zur Erstellung und Verbreitung von Videos mittlerweile sehr gering. Durch diese drastische Senkung der finanziellen, sowie technischen Barrieren ist eine große Gruppe an Amateurproduzenten entstanden. Der zweite Trend ist die hohe Verbreitung sowie Verwendung von mobiler Technologie. Die Möglichkeit jederzeit ein Video aufzunehmen zu können ändert auch das Verhalten im Umgang mit Videos und eröffnet neue Möglichkeiten. Nicht nur sind Smartphones in der Lage Videos in HD Qualität aufzunehmen, die Videos können auch sofort über das Internet verschickt und geteilt werden. Ein Bereich bei dem man die Auswirkungen der ständigen Allgegenwärtigkeit von mobilen Geräten bereits beobachten konnte, ist die Veränderung der Berichterstattung bei globalen Newsevents. So spielten beispielsweise Smartphones beim Arabischen Frühling 2011 eine große Rolle [39] und Menschen aus der ganzen Welt konnten die Revolution durch, mit Hilfe von Smartphones erstellten und

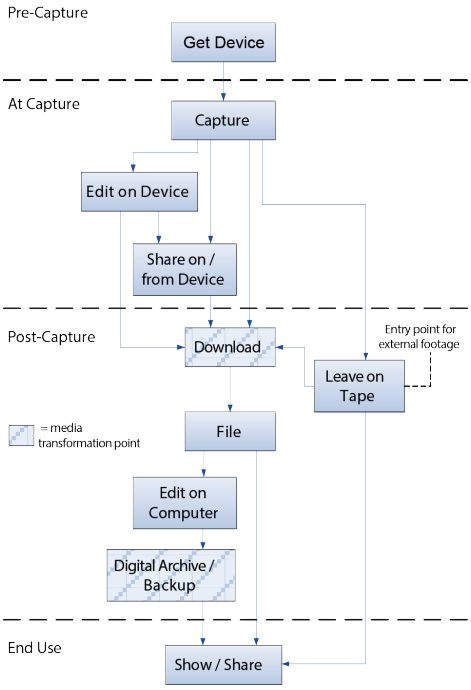

Abbildung 3.1: Videowork Lifecycle beschreibt mögliche Schritte bei der Arbeit mit Video [40]

im Internet veröffentlichten Videos, mitverfolgen. Der dritte Trend ist interaktives Fernsehen oder auch iTV.

Dabei geht es darum, dem User mehr Kontrolle darüber zu bieten, welche Inhalte er konsumiert und wie diese Inhalte konsumiert werden. Dazu zählt das bereits gängige Angebot von Video-on-Demand-Diensten genauso, sowie die Erforschung von Chancen, Social Media Aspekte mit dem Fernsehangebot zu verknüpfen [41]. Ein weiteres Beispiel, das in den Bereich von iTV fällt, sind sogenannte Second Screen Anwendungen [42]. Dabei handelt es sich um Tablet Apps, die das Fernsehangebot durch zusätzliche Informationen, alternative Kameraeinstellungen oder Anbindung an Soziale Netzwerke erweitern. Zusätzlich zu diesen drei Trends, ist es auch interessant, sich die Verwendung von Video aus Usersicht genauer anzusehen. Oft werden neue Interaktionstechniken entwickelt ohne Rücksicht darauf zu nehmen, wie und unter welchen Umständen das Medium Video genutzt wird. Kirk et al. [40] haben in diesem Zusammenhang versucht Muster und Abläufe bei der Verwendung von Video im privaten Bereich zu finden. Dafür haben sie Familien, sowie Teenager beobachtet und befragt. Aus ihren Ergebnissen haben sie den Videowork Lifecycle (Abbildung 3.1) erstellt , der aus den vier Phasen besteht: *Pre-Capture*, *At Capture*, *Post-Capture* und *End Use*Dabei hat sich gezeigt, dass es grundlegende Unterschiede zwischen der Verwendung von Videokameras und der Verwendung von Smartphones gibt. Das unterstreicht auch die Bedeutung des zuvor besprochenen Trends

der mobilen Geräte. Kirk et al. unterscheiden zwei Kategorien im Umgang mit Video: *Heavyweight* und *Lightweight*. Als *Heavyweight* wird das geplante Aufnehmen von Videos mit Hilfe einer Videokamera bezeichnet. Die Videokamera dient ausschließlich zur Aufnahme und das Videomaterial wird nach der Aufnahme auf den Desktop Computer übertragen, um es dort zu editieren und archivieren. Der User will das Material oft schneiden, auch wenn das eine zeitaufwändige Tätigkeit ist und der Fokus liegt generell mehr auf Kreativität. Im Gegensatz dazu bezeichnet *Lightweight* das spontane Aufnehmen von Videos mit Hilfe eines Smartphones. Das Video bleibt dabei meistens am Gerät selbst, weil das Smartphone in der Lage das Video aufzunehmen, abzuspeichern, wiederzugeben und mit anderen Personen zu teilen. Meistens wird das Video nicht editiert und der Fokus liegt am schnellen Festhalten und Teilen von Momenten.

3.2 Videoannotation

Annotieren bedeutet etwas mit einer Anmerkung zu versehen. Das Annotieren von Texten hat eine lange Tradition und wurde bereits in Manuskripten der Antike betrieben. Anmerkungen können unter anderem verwendet werden, um Wichtiges hervorzuheben, Bestehendes zu ergänzen oder auf Fehler hinzuweisen. Annotieren beschränkt sich allerdings nicht nur auf Text alleine. Auch andere Inhalte wie Bilder, Videos oder Audioaufnahmen können annotiert werden. Durch den rapiden Anstieg an verfügbaren Videomaterial, sowie des Videokonsums ist es ein logischer Schritt, Anwendungen zu entwickeln, die das Annotieren von Videos ermöglichen. Für das Annotieren von Texten gibt es bereits eine Vielzahl von Programmen, die über eine große Anzahl an einfachen Möglichkeiten und Optionen zur Annotation verfügen. Sieht man sich im Vergleich dazu die gängigen Videoplayer-Programme an, enthalten diese im Gegensatz dazu meist keine Möglichkeit um Annotationen zu erstellen.

Um die vielfältigen Anwendungsszenarien für das Annotieren von Videos zu demonstrieren, stellt dieser Abschnitt diverse Anwendungen und Einsatzgebiete vor. Ein Begriff, in dessen Zusammenhang die Videoannotation, sowie das Annotieren generell, immer wieder auftaucht, ist die Semantische Lücke (*semantic gap*) [44]. Dabei geht es um die offensichtlichen Unterschiede zwischen der maschinellen Beschreibung von visuellen Daten (z.B. ein Bild oder ein Video) im Gegensatz zu der Beschreibung und Interpretation dieser Daten durch einen Menschen im gegeben Kontext. Die semantische Lücke zu schließen, oder sie zumindest zu verkleinern, ist eines der Hauptprobleme bei der Entwicklung von Systemen zur Suche, Verwaltung und Analyse von Videos. Aus dieser Problematik heraus sind eine Vielzahl von Programmen und Tools entstanden, die dabei helfen sollen, semantische Information zu Videos durch Annotation zu erstellen. Dabei gibt es drei grundlegende Ansätze. Auf der einen Seite die automatische Analyse des Videos bzw. der einzelnen Frames eines Videos anhand von Methoden der Bildverarbeitung, auf der anderen Seite die manuelle Annotation durch einen User. Der dritte Ansatz ist die Kombination beider Varianten.

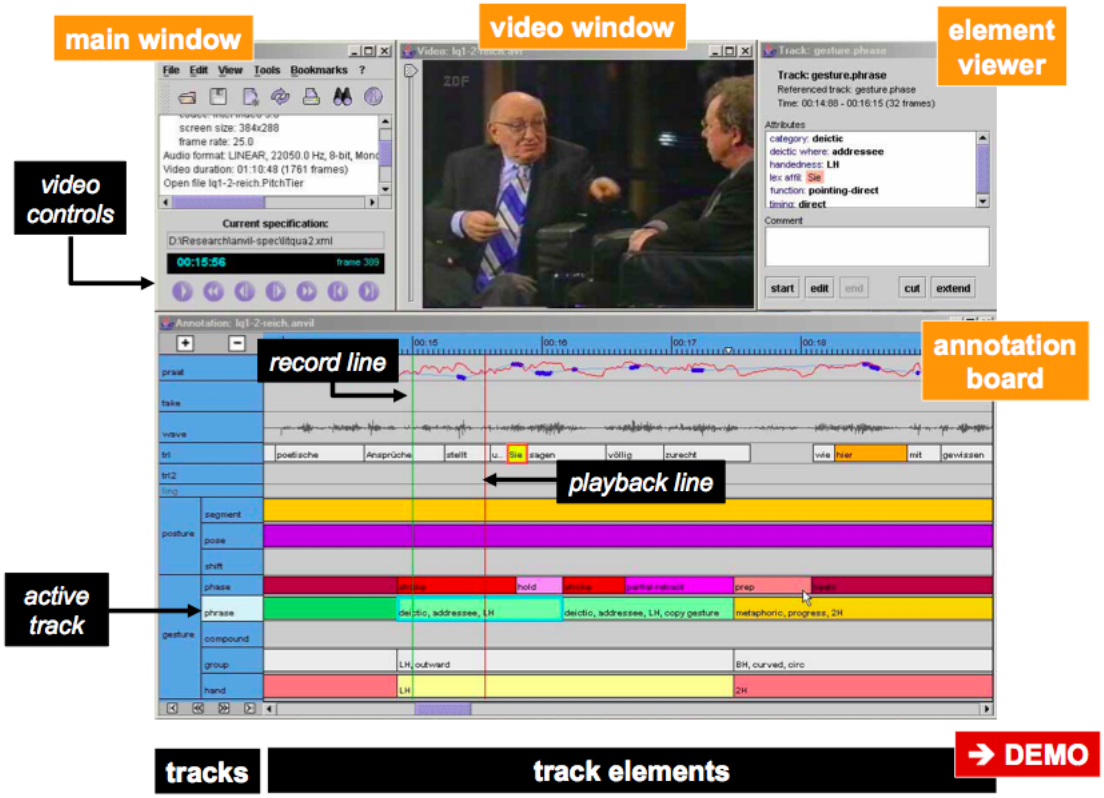

Abbildung 3.2: ANVIL User-Interface [2]

Dasiopoulou et al. [45] geben in ihrer Arbeit einen umfassenden Überblick über Desktop Anwendungen zur semantischen Videoannotation. Diese Programme wurden teilweise als allgemeine Tools zur Videoannotation und teilweise für spezifische Anwendungsfälle entwickelt. Die spezifischen Anwendungsfälle stehen meist in engem Zusammenhang mit wissenschaftlichen Arbeiten. Eines der von Dasiopoulou et al. vorgestellten Anwendungen ist ANVIL (*Video Annotation Research Tool*) [2]. ANVIL ist ein Videoannotation-Programm für Windows, Mac OS und Linux. Es wurde ursprünglich für die Annotation von menschlichen Gesten in Videos entwickelt. Als weitere Einsatzgebiete werden die Annotation von Videos in so unterschiedlichen Bereichen wie der HCI, Mutter-Kind Interaktion, Filmanalyse und Meereskunde genannt. Mit ANVIL können Annotationen in verschiedenen Formaten und auf mehreren Ebenen erstellt werden. Abbildung 3.2 zeigt das GUI von ANVIL. Die meisten anderen Desktop Anwendungen zur Videoannotation haben ähnlich komplexe User-Interfaces, benötigen eine nicht zu vernachlässigende Einarbeitungszeit und richten sich hauptsächlich an Experten.

Einen einfacheren Ansatz für ein anderes Anwendungsgebiet hat Hosack B. [43] mit seinem

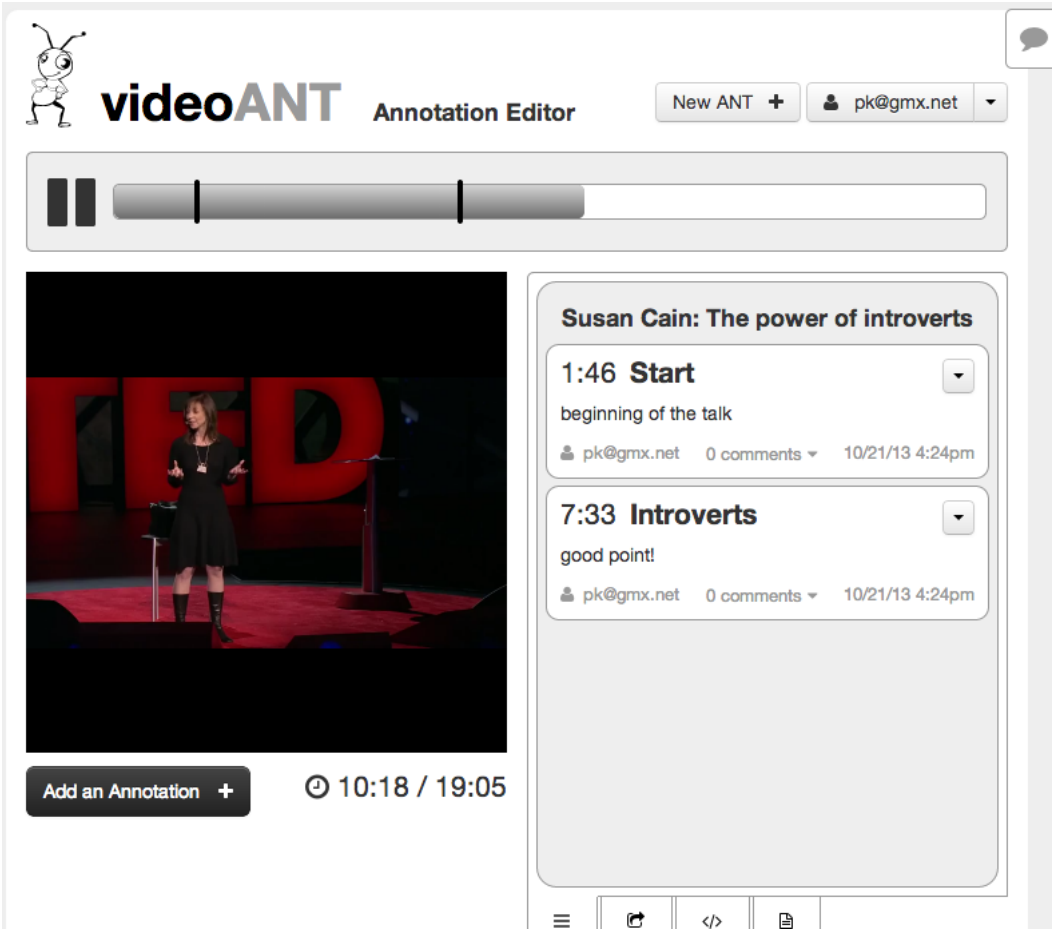

Abbildung 3.3: Video ANT User-Interface [43]

Programm VideoANT gewählt. Mit Hilfe von VideoANT [46] ist es möglich, Es wurde für die Verwendung im Bildungsbereich entwickelt und soll Lehrenden sowie Studenten dabei helfen, Feedback auf einfache Art mittels Videoannotationen zu verfassen und auszutauschen. Es können textuelle Annotationen zu beliebigen Zeitpunkten in einem Video verfasst werden. Anhand dieser Annotationen kann zusätzlich durch das Video navigiert werden. Außerdem ist es möglich, Kommentare zu Annotationen zu verfassen. Das annotierte Video kann mittels eines Links geteilt werden. Wobei es zwei verschiedene Links gibt. Einer führt zu einer view-only Variante des annotierten Videos und der andere erlaubt das Editieren des Videos, sowie das zusätzliche Hinzufügen von Annotationen

Die bisher vorgestellten Anwendungen sind hauptsächlich für den Desktop entwickelt worden. Für Tablets gibt es im Vergleich dazu noch relativ wenig Anwendungen, die das Annotieren von Videos ermöglichen. Lux M. & Riegler. M [47] haben eine Applikation für Android

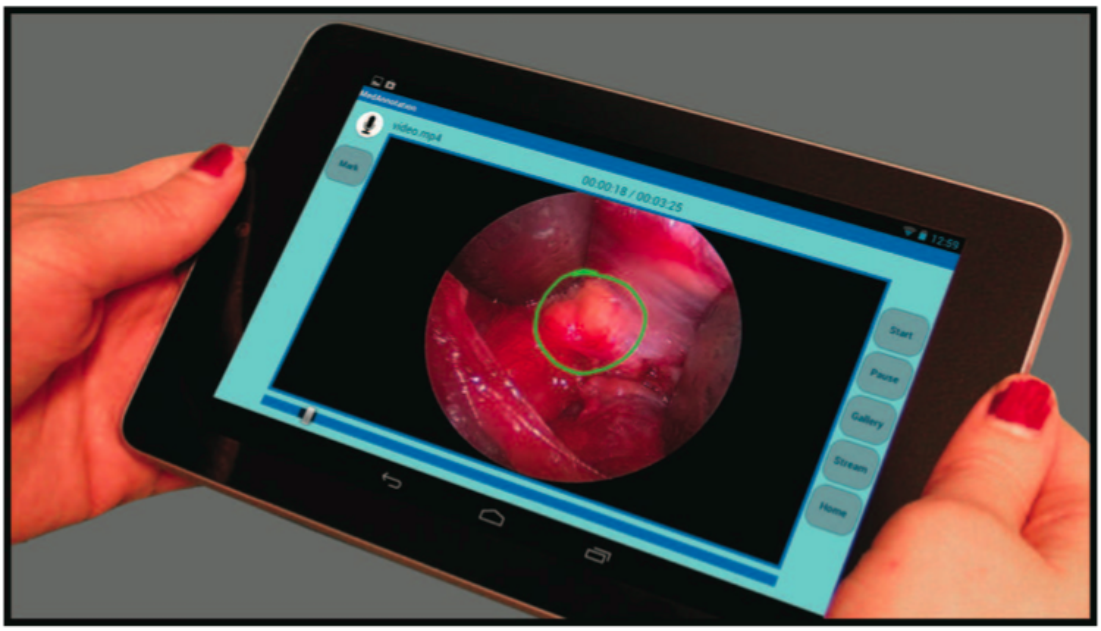

Abbildung 3.4: Endoskopie App User-Interface [47]

Tablets im medizinischen Bereich entwickelt (Abbildungen 3.4). Die Applikation soll Ärzten das Annotieren von Aufnahmen, die während einer Endoskopie entstehen, ermöglichen. Videos können sowohl durch Audioaufnahmen, als auch durch das direkte Zeichnen auf das Video annotiert werden. Audioannotationen können zusätzlich automatisch mit Hilfe einer Speech-To-Text Funktion in Text umgewandelt werden. Der Fokus bei der Entwicklung lag darauf, die den Ärzten bereits bekannten Interaktionsmethoden so gut wie möglich auf den Tablet zu transferieren.

Eine weitere Anwendung für Tablets haben Cabral D. et al. [48] entwickelt. Das Creation-Tool (Abbildung 3.5) wurde zum Annotieren von zeitgenössischen Tanzvideos entwickelt. Es unterstützt sowohl Choreographen, als auch Tänzer bei der Auseinandersetzung mit Tanzaufnahmen. Dabei werden unterschiedliche Methoden zur Annotation angeboten, wie die Eingabe von Text, Aufnahme von Audio oder das Skizzieren. Mit dem Creation-Tool können neben bereits zuvor aufgenommenen Videos, auch Videostreams in Echtzeit annotiert werden. Dafür werden zwei unterschiedliche Methoden angeboten. Beim *continuous* - Modus wird das Video während einer Annotation nicht pausiert, sondern läuft weiter. Im Gegensatz dazu wird beim *suspended* - Modus wird das Video während einer Annotation nicht pausiert sondern läuft weiter. Im Gegensatz dazu wird beim suspended - Modus das aktuelle Frame des Videos eingefroren und kann dann annotiert werden während der Stream weiter läuft. Die Anwendung kann

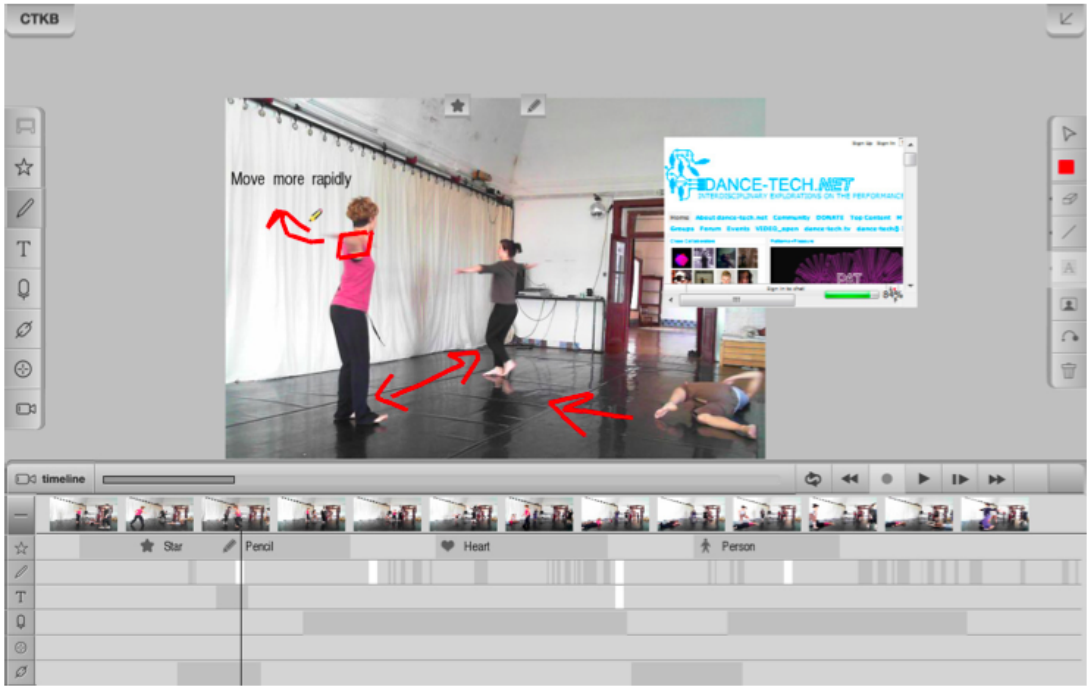

Abbildung 3.5: Creation-Tool User-Interface [48]

sowohl mit einem Stylus, als auch mit den Fingern bedient werden und es wurden in diesem Zusammenhang Interaktionen, die auf der gleichzeitigen Eingabe mittels Stift und Touch basieren, untersucht. Das Creation-Tool wurde sowohl von Choreographen, als auch von Tänzern gut angenommen. Choreographen verwendeten eher die Live-Funktion, wohingegen Tänzer ihre Aufnahmen im Nachhinein analysierten.

Diese Beispiele zeigen, dass die Videoannotation nicht auf einen speziellen Anwendungsbereich beschränkt ist. Von der Forschung, über die Anwendung im medizinischen sowie Bildungsbereich, bis hin zur Analyse im Sport reichen die möglichen Einsatzgebiete.

Anforderungen und Kriterien eines Videoannotationsystem

Anhand der im letzten Abschnitt besprochenen Anwendungen erkennt man, dass Systeme zur Videoannotation sehr unterschiedliche Funktionen besitzen können. In Folge werden allgemeine Eigenschaften und Anforderungen an Videoannotationsysteme besprochen. Bevor ein Video annotiert werden kann, muss zuerst ein Video vorhanden sein. Mögliche Quellen für Videos können Aufnahmen von Videokameras, Videos von Internetplattformen wie YouTube oder auch Livestreams sein. Viele Tablets verfügen über eine integrierte Videokamera, mit der Videos aufgenommen werden können. Das ermöglicht es, Videos direkt nach der Aufnahme am Gerät

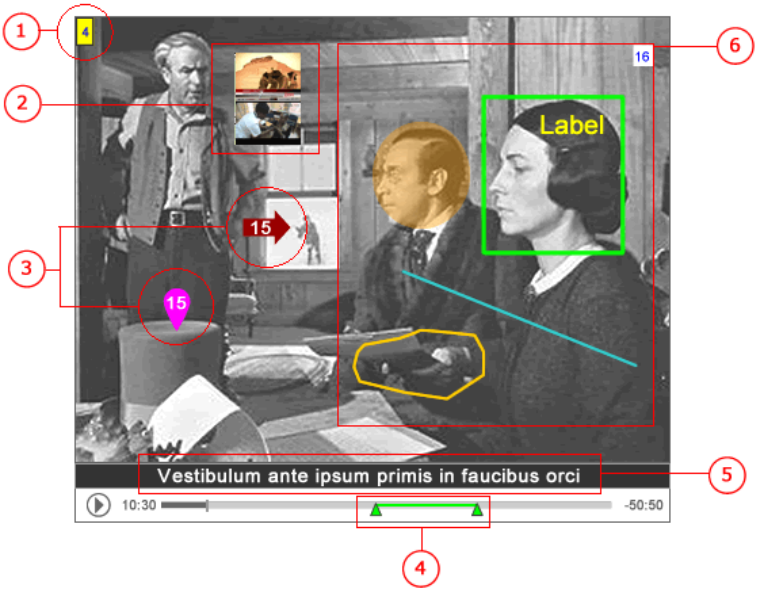

Abbildung 3.6: Anforderungen an ein Videoannotationssystem laut *Annotations at Havard* [49]

selbst zu annotieren und es fällt somit ein Zwischenschritt weg. Es existieren viele verschiedene Videoformate und ein Videoannotationsystem sollte möglichst viele Formate unterstützen. Neben dem Import, oder der Aufnahme von Videos, muss danach auch eine Möglichkeit vorhanden sein, das Video wiederzugeben. Dabei sollten grundlegende Playbackfunktionen wie Start, Stopp, Pause, Vorspulen, Zurückspulen und das Springen im Video möglich sein. Die *Annotations at Havard* Initiative [49] zeigt anhand eines Bildes (Abbildung 3.6) welche Funktionen ein System zum Annotieren von Videos benötigt:

1. Die Annotation des Videos als Ganzes. Das ist dann wichtig, wenn man mit mehreren Videos arbeitet. Durch das Hinzufügen von zusätzlicher Information zum gesamten Video können beispielsweise Videosammlungen besser sortiert und durchsucht werden.

2. Durch die Überlagerung von Bildern oder anderen Videos ist es möglich Verweise zu erstellen, oder bestimmte Details genauer hervorzuheben.

3. Mit Hilfe von Markern können bestimmte Objekte, Personen oder ähnliches markiert werden. Marker sollten in verschiedenen Varianten und Formen vorhanden sein.

4. Eine Annotation in einem Video hat meistens einen bestimmten Start- und einen bestimmten Endzeitpunkt. Deshalb ist es notwendig, ein passendes Zeitintervall für jede Annotation festlegen zu können.

5. Untertitel sind im Prinzip schon eine Art von Annotation. Manche Videos enthalten bereits Untertitel, aber es wäre grundsätzlich auch möglich, Untertitel für das Video automatisch aus den Audiodaten zu erstellen.

6. Die Möglichkeit, das Video mit vom User erstellten Grafiken, sowie kurzen Texten zu überlagern. Grafiken können einfache Formen wie Kreise, Rechtecke oder Linien sein. Aber auch Freihandzeichnungen sollten möglich sein. Um sicher zu gehen, dass sich die Grafiken vom Videobild abheben, sollte die Strichfarbe sowie Strichstärke frei wählbar sein.

Khurana K. & Chandak M. B. [50] haben einen Überblick über verschiedene Techniken von Videoannotation erstellt. Dabei definieren sie drei verschiedene Arten von Metadaten, die einem Video zugeordnet werden können.

- *Content independent metadata* - Das sind Metadaten, die zwar mit dem Video in Zusammenhang stehen, aber nicht den Inhalt direkt beschreiben. Das kann zum Beispiel der Autor des Videos, das Aufnahmedatum oder der Aufnahmeort sein. Ein wichtiges Merkmal dieser Daten ist, dass sie grundsätzlich nicht aus dem Videomaterial selbst hervorgehen oder ausgelesen werden können.

- *Content dependent metadata* - Darunter fallen diverse Merkmale des Videos die aus den einzelnen Frames oder einzelnen Videosequenzen extrahiert werden können. Das sind Dinge, wie Form, Farbe, Kontur, Kanten oder Bewegung. Diese Merkmale können mit Hilfe von diversen Algorithmen aus der Bildverarbeitung extrahiert werden. Außerdem lassen sich durch diese Merkmale auch sogenannte Feature Vektoren ableiten. Diese Vektoren bilden die Grundlage für Verfahren des maschinellen Lernens.

- *Content descriptive metadata* - Ist die dritte Form der Metadaten. Dabei handelt es sich um semantische Daten. Sie geben Aufschluss über Bedeutung und Zusammenhang von Inhalten, vorkommenden Ereignissen, Objekten oder Personen. Diese Daten können nur schwer bis gar nicht automatisch aus einem Video abgeleitet werden, sind aber oft die wertvollsten Daten. Besonders im Zusammenhang mit der Reduktion der Semantischen Lücke.

Ein weiteres Kriterium von Videoannotationssystmen ist die Granularität von Annotationen [45]. Darunter versteht man für welche Bereiche im Video die jeweiligen Annotationen gelten. Eine Annotation kann, wie schon besprochen, dem ganzen Video zugeordnet sein. Im Gegensatz dazu ist eine feinere Granularität dann gegeben, wenn eine Annotation nur für eine gewissen Zeitspanne im Video gilt. Wiederum eine Stufe darunter ist die Annotation von einzelnen Frames. Darüber hinaus kann eine noch feinere Granularität erzielt werden, wenn

man gewisse Regionen eines Frames annotieren kann, bis hin zu einzelnen Pixeln. Wobei das Annotieren von einzelnen Pixeln in den meisten Anwendungen nicht sinnvoll ist. Durch die zeitliche, sowie spatiale Zuordnung von Annotationen ergeben sich zusätzliche Möglichkeiten. Unter anderem sind das weitere Methoden zur Navigation durch das Video. So kann man dadurch beispielsweise eine Funktion umsetzen, die durch die Auswahl einer Annotation zur zugehörigen Stelle im Video springt. Das kann je nach vorhandener Granularität, ein genauer Punkt im Video sein, oder aber auch ein gewisser Bereich oder eine gewisse Szene.

Eine Annotation selbst kann verschiedene Formate besitzen. Oft wird davon ausgegangen, dass Annotationen rein in textueller Form verfasst werden. Das ist aber nicht die einzige Option. Genauso wie neben Text auch Bilder, Videos und vieles mehr annotiert werden kann, können Annotation selbst aus verschiedenen Formaten bestehen. Eine Annotation kann neben Text, auch eine Grafik, ein weiteres Video oder eine Audioaufnahme sein. Bei der Entwicklung eines Systems zur Annotation von Videos sollte auf diese verschiedenen Formen von Annotationen geachtet werden. Je nach Anwendungsbereich und der verwendete Plattform haben die jeweiligen Formate gewisse Vor- und Nachteile. Entwickelt man zum Beispiel ein System für ein Smartphone bietet sich die Annotation durch Spracheingabe an [51]. Dadurch können die Limitierungen, die sich durch die Größe des Bildschirms oder durch die Toucheingabe ergeben, umgangen werden. Neben den verschiedenen Formaten, können diese auch in verschiedenen Variationen implementiert sein. Der Text einer Annotation kann entweder völlig frei verfasst werden, oder von einer Liste aus Kategorien oder Tags ausgewählt werden. Das gleiche gilt für Grafiken. Manche Systeme bieten nur gewisse, vorgefertigte Formen wie Pfeile, Rechtecke und Linien. Andere Systeme hingegen ermöglichen auch das Annotieren durch Freihandzeichnungen. Auch in diesem Fall ist je nach Anwendungsfall abzuwägen, welche möglichen Formen der Annotation Sinn machen.

Automatisierte Ansätze

Die Forschung im Bereich der automatisierten Videoanalyse sowie Videoannotation besitzt mittlerweile eine sehr umfangreiche Geschichte. Es gibt unzählige Ansätze, um die Arbeit mit Videos, sowie den Konsum von Videos, durch automatisierte Verfahren zu vereinfachen. Aus diesem Grund kann dieser Abschnitt nur einen kleinen, nicht kompletten Ausschnitt dieser Methoden bieten. Viele dieser Verfahren basieren auf Erkenntnissen der Bildverarbeitung, die mit gewissen Einschränkungen und Modifikationen auch bei Videos angewendet werden können.

Eines der Probleme in diesem Kontext ist die automatische Videosegmentierung. Das bedeutet, die sinnvolle Unterteilung eines Videos in kleinere Teilstücke. Durch die Unterteilung eines Videos wird das Durchsuchen eines Videos und das Navigieren in einem Video vereinfacht. Die gängigste Variante zur Videosegmentierung ist die Schnitterkennung. Bei der Schnitterkennung wird versucht die Schnitte, oder auch Übergänge, in einem Video automatisch zu finden.

Es gibt zwei unterschiedliche Arten von Schnitten: harte Schnitte und weiche Schnitte [52]. Harte Schnitte sind Schnitte, die keinen Übergang besitzen und abrupt von einem Frame zum anderen auftreten. Weiche Schnitte besitzen dagegen einen Übergang. Wie genau dieser Übergang aussieht, ist nicht definiert. Einer der am häufigsten eingesetzten Effekte in Videos ist die Überblendung, bei der eine Szene allmählich in eine andere Szene übergeht. Harte Schnitte sind relativ einfach automatisch zu erkennen, weil sich die aufeinanderfolgenden Frames meistens deutlich unterscheiden. Weiche Schnitte stellen eine Herausforderung dar, da die Veränderung fließend passiert und deshalb mit einer Bewegung der Kamera oder der Bewegung eines Objekts verwechselt werden kann. Für die automatische Schnitterkennung gibt es unterschiedliche Ansätze, die auf der Basis von einem Vergleich auf der Pixel Ebene, dem Template-Matching oder einem Histogrammvergleich funktionieren [53]. Die Schnitterkennung durch einen Vergleich von Histogrammen besteht aus zwei Stufen. Zuerst wird das Farbhistogramm des Videoframes berechnet. Diese Histogramme werden dann verglichen, falls sie sich stark unterscheiden, ist es wahrscheinlich, dass ein Schnitt stattgefunden hat. Der Vergleich wird anhand eines Schwellenwertes durchgeführt. Die richtige Auswahl dieses Schwellenwertes ist das entscheidende Kriterium. Ein Problem bei diesem Ansatz ist, dass zwei unterschiedliche Szenen durchaus ein sehr ähnliches Farbhistogramm besitzen können.

Neben der Schnitterkennung, ist das Extrahieren von Key-Frames ein weiteres Thema in der Videoanalyse. Ein Key-Frame is jenes Frame einer Videosequenz, das den typischen Inhalt dieser Sequenz am besten vermittelt [54]. JJe nach Länge und Komplexität einer Sequenz, kann diese auch mehrere Key-Frames enthalten. Ist man in der Lage diese Frames automatisch zu extrahieren, ist es möglich eine visuelle Übersicht des Videos zu erstellen und das Video somit zu indizieren. Dadurch können wichtige Szenen in einem Video schneller gefunden und in weiterer Folge annotiert werden. Genau wie bei der Schnitterkennung wurden auch für die Extraktion von Key-Frames in den letzten Jahren eine Vielzahl von Verfahren entwickelt. Sujatha C. & Mudenagudi U. [55] haben in ihrer Arbeit einige dieser Methoden zusammengefasst und verglichen.

Der nächste logische Schritt in der automatischen Verarbeitung von Videos, ist die Erkennung von Objekten im Video. Die Objekterkennung ist eines der komplexesten Themen in der Bildverarbeitung. Die Vorteile, die eine automatische Objekterkennung für Videoannotationsystem mit sich bringt, liegen auf der Hand. Es könnten beispielsweise Annotationen erstellt werden, die einem Objekt folgen. Bei der grafischen Videoannotation tritt nämlich oft das Problem auf, dass durch die Bewegung des Videobildes, die Annotation nur für ein einziges Frame passend ist. Objekterkennung würde dieses Problem lösen. Goldman et al. [56] beschreiben ein System, das durch die Erkennung der Bewegung von Objekten zur Annotation und Navigation von Videos verwendet werden kann. Damit das möglich ist, muss das Video zuvor in diversen Schritten analysiert werden. Dazu verwenden sie eine Methode die Bewegung im Video als Partikel darstellt. Dieser Ansatz wurde gewählt, weil die berechneten Partikel ein dichtes

Feld darstellen und sogar Punkte in Bereichen des Videos generieren, die nur wenig markante Merkmale besitzen. Die Partikel werden danach im nächsten Schritt in verschiedene Regionen gruppiert welche die Objekte repräsentieren. Durch diese Vorbearbeitung und Unterteilung in Objekte, können verschiedene Applikationen implementiert werden. Neben unterschiedlichen Methoden zur Annotation (Abbildung 3.7), zeigen Goldman et al. auch, wie man durch das Video navigieren kann, indem man direkt auf ein Objekt im Video klickt und es entlang seines Bewegungspfades hin und her zieht. Diese neuartige Interaktionstechnik würde vermutlich auch sehr gut auf Tablets mit Toucheingabe funktionieren.

Neben der automatischen Schnitt, Key-Frame und Objekterkennung, findet man in der Literatur viele Ansätze zur automatischen Annotation von Videos basierend auf Ontologien. Eine Ontologie ist in diesem Zusammenhang die Repräsentation von semantischen Begriffen und deren Verhältnisse zueinander. Eine Ontologie kann für verschiedene Themenbereiche erstellt werden und bildet eine Art Klassifikationssystem. Der Unterschied zu Tags oder Keywords besteht darin, dass die Kategorien hierarchisch miteinander verknüpft sind. Es existieren verschiedene formale Sprachen in denen Ontologien definiert werden können [50]. Bagdanov D. A. et al. [57] präsentieren in ihrer Arbeit ein System zur Analyse von Fußballvideos, das durch Ontolgien eine Verbindung zwischen den Rohdaten und semantischen Konzepten erstellt. Dadurch ist es unter anderem möglich, Videos automatisch nach ähnlichen Szenen wie zum Beispiel ein Schuss aufs Tor zu durchsuchen.

Standards

Durch die Entwicklung und Verwendung von unterschiedlichen Applikationen zur Videoannotation ist eine Vielzahl an unterschiedlichen Formaten entstanden, mit denen die erstellten Annotationen abgespeichert werden. Viele Applikationen verwenden ein individuelles XML Format, das nicht mit anderen Applikationen kompatibel ist. Dadurch wird nicht nur der Aus-

Abbildung 3.7: Unterschiedliche Arten der Annotation [56]

tausch von Annotationen erschwert, sondern auch das Einbinden und die Verwendung von vorhandenen Annotationen in neue Applikationen. Allerdings gibt es durchaus Bestrebungen standardisierte Formate für Annotationen zu entwickeln, die solche Kompatibilitätsprobleme lösen sollen.

Eine dieser Bestrebungen ist der MPEG-7 Standard [58] . Er wurde im Jahr 2002 von der ISO unter dem vollen Namen *Multimedia Content Description Interface* eingeführt und bietet eine standardisierte Form zur Beschreibung von Multimediainhalten mit Hilfe von zusätzlichen Metadaten. Die Metadaten werden durch die MPEG-7 *Description Definition Language* (DDL) definiert. Die DDL basiert wiederum auf dem XML Schema. Mittels der DDL werden verschiedene *Multimedia Description Schemes* (MDS) definiert. Diese MDS enthalten unterschiedliche Elemente. Das können auf der einen Seite allgemeine Beschreibungen der Videodatei wie die Autoren, Erstellungsdatum oder eine Zusammenfassung des Inhaltes sein. Auf der anderen Seite aber auch genauere Details zu den visuellen Elementen des Videos, wie zum Beispiel die Beschreibung von Texturen, Farben oder die Bewegung von Objekten. Die Flexibilität des MPEG-7 ermöglicht es auch, Annotationen auf der Frame-Ebene zu speichern. Ein Vergleich von Programmen die den MPEG-7 Standard zur Annotation verwenden haben Döller M. & Lefin N. [59] erstellt.

Ein anderer Ansatz zur Standardisierung von Annotationen ist die Initiative der *Open Annotation Collaboration* [60]. Der Fokus der Initiative liegt auf der Erschaffung eines Standards, der den freien Austausch von Annotationen zwischen Usern, Servern und Applikationen im World Wide Web unterstützt. Dazu wurde das *Open Annotation Data Model* erschaffen, das im Moment als W3C Community Draft in Version 1.0 existiert [61]. Dabei ist eine Annotation als eine Verbindung zwischen Ressourcen definiert. Wobei diese meistens aus einem *body* und einem *target* bestehen, die in einer Relation zueinander stehen und verschiedene Metadaten beinhalten können (Abbildung 3.8). Diese Verbindungen werden mit Hilfe von RDF-Graphen [62]

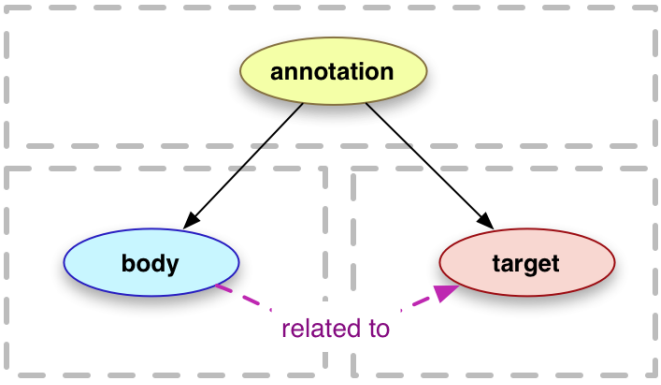

Abbildung 3.8: Open Annotation Mode-Body und Target [60]

modelliert. Durch die Modellation als Graphen ist der Standard sehr flexibel und erlaubt eine Vielzahl von verschiedenen Abbildungen, wie zum Beispiel das Annotieren von Videos oder Videofragmenten mit Bildern, Texten oder Videos. Da das *Open Annotation Data Model* erst vor kurzen veröffentlicht wurde, gibt es noch keine Applikationen zur Videoannotation die das Modell verwenden.

3.3 Videointeraktionstechniken

Ein Großteil der User-Interfaces zum Abspielen von Videos ist an das Interface eines Videorekorders angelehnt. Sie bieten eine Reihe von Buttons zum Abspielen, Stoppen, Pausieren, Vor- und Zurückspielen des Videos. Diese Buttons werden separat vom eigentlichen Video angezeigt und das Video kann nur indirekt durch die Verwendung dieser Funktionen gesteuert werden. Auch viele Applikationen, die zur Videobearbeitung verwendet werden haben oft analoge Anlagen zum Videoschnitt als Vorbild. Durch die Digitalisierung von Videos, ist diese Nachahmung von anlogen Geräten eigentlich nicht nötig und es können neue Interaktionstechniken entstehen.

Ramos G. & Balakrishnan R. [1] haben einige sehr interessante Interaktionstechniken für Videos auf Tablets unter Verwendung eines druckempfindlichen Stylus entwickelt. Einer dieser Techniken ist der *PVslider*, mit dessen Hilfe man in einem Video vor und zurück navigieren kann. Der Bereich in dem man navigieren kann, ist davon abhängig, wie weit weg man sich mit dem Stylus vom Video befindet. Desto näher die Interaktion ausgeführt wird, desto kleiner ist der Bereich, und umso genauer ist die Navigation. Das Intervall, in dem die Navigation stattfindet wird also durch vertikale Entfernung bestimmt. Wenn man den Stylus nach links bewegt wird im Video zurück gesprungen, bei einer Bewegung nach rechts, nach vorne. Mit dem *PVslider* kann neben der Position im Video auch gleichzeitig die Geschwindigkeit mit der die Navigation stattfindet bestimmt werden. Das passiert auf der horizontalen Ebene. Desto weiter der Stylus nach rechts oder links bewegt wird, desto schneller wird die Navigation ausgeführt. Apple verwendet in ihrem Video- und Musikplayer auf iOS eine Interaktion, die der *PVslider* Komponente sehr ähnlich ist. Die normale Slider-Komponente wird so erweitert, dass zusätzlich zur Position im Video, die Navigationsgeschwindigkeit (Scrubbing-Geschwindigkeit) verändert werden kann. Im Gegensatz zum *PVslider* wird hier die Navigation umso feiner, desto weiter der Finger vom Slider entfernt ist. Die Geschwindigkeit kann mittels vier verschiedener Stufen eingestellt werden: 100%, 50%, 25% und fein. Durch die Änderung der Navigationsgeschwindigkeit wird auch indirekt der Bereich in dem die Navigation stattfindet festgelegt. Bei 100% kann durch das ganze Video gesprungen werden. Navigiert man hingegen auf der feinsten Stufe, ist der Bereich durch die Breite des Touchscreens begrenzt. Sobald der Finger nämlich den Touchscreen verlässt, ist die Interaktion zu Ende.

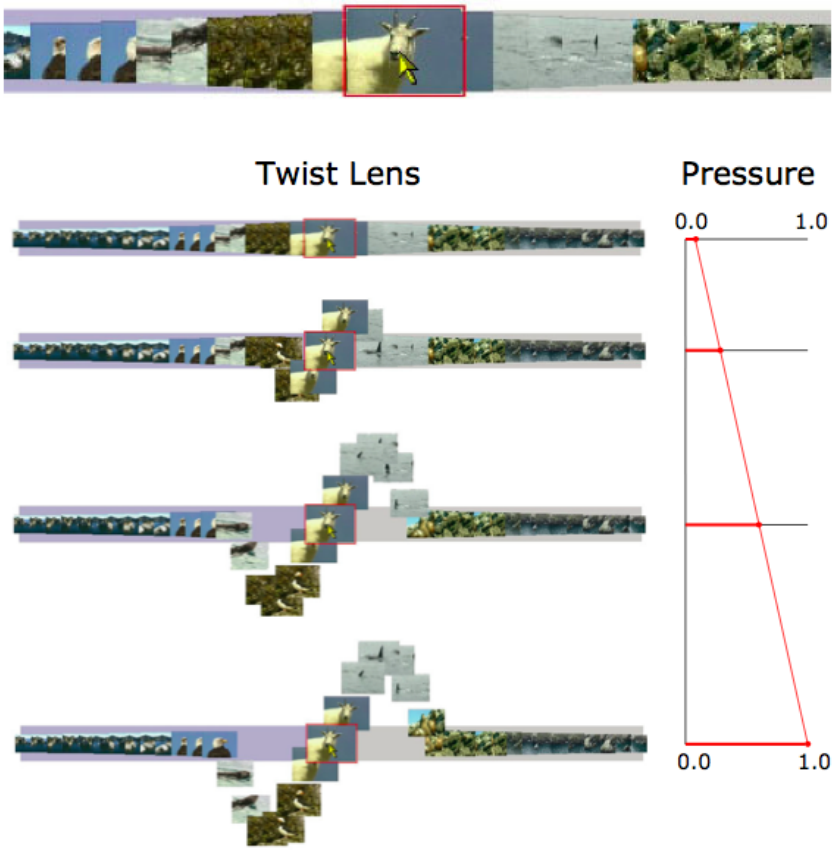

Abbildung 3.9: Twist Lense Interaktionstechnik zum genauen Navigieren in einem Video [1]

Neben dem *PVslider* stellen Ramos G. & Balakrishnan R. in ihrer Arbeit noch eine weitere Möglichkeit zur Videonavigation vor. Mit dem *Twist Lens slider* kann zu einer beliebigen Position in einem Video gesprungen werden (Abbildung 3.9). Dazu wird ein Streifen, mit Bildern von einzelnen Frames aus dem Video erzeugt und mit dem Effekt einer Fish-Eye Linse kombiniert. Je nachdem, wo auf diesem Streifen man sich mit dem Stylus befindet, wird das aktuelle Frame vergrößert dargestellt. Ein Problem, dass bei dieser Technik auftritt, ist dass die nachfolgenden sowie vorherigen Bilder teilweise verdeckt werden. Aus diesem Grund werden die Bilder der Frames in der Umgebung der aktuellen Stylusposition in einer S-Form angezeigt. Zusätzlich ist die Ausprägung dieser S-Form auch noch vom Druck der mit dem Stylus ausgeübt wird, abhängig. Je mehr Druck ausgeübt wird, desto weiter voneinander werden die einzelnen Frames dargestellt. Einen Nachteil, den diese Technik mit sich bringt, ist, dass durch die Anordnung in S-Form deutlich mehr Platz am Bildschirm benötigt wird.

Zusammenfassung

Die Verschiebung, von der professionellen Videoproduktion, hin zu der Erstellung von Videos durch Amateure, ist eine der wichtigsten Entwicklungen des Mediums der letzten Jahre. Durch die Verbreitung von mobilen Geräten und die einfachen Möglichkeiten Videos über das Internet auszutauschen, ergeben sich viele neue Chancen. Neben den technischen Neuerungen, ist es aber allerdings genauso wichtig, die veränderten Userverhalten im Zusammenhang mit Videos zu analysieren. Der in diesem Kapitel beschriebene Unterschied zwischen *light-* und *heavyweight* Verwendung von Video ist eine dieser Veränderungen.

Die Annotation von Videos ermöglicht es, sich mit den Inhalten intensiver auseinanderzusetzen und sie zu reflektieren. Mögliche Anwendungen dafür gibt es viele. Der Lehrbereich, das wissenschaftliche Arbeiten, der medizinische oder Sport-Bereich sind nur einige wenige Beispiele, die von Programmen zur Videoannotation profitieren würden. Viele der bisher verfügbaren Applikationen sind sehr komplex und nicht intuitiv bedienbar. Ein System zur Videoannotation muss einige grundlegende Funktion mit sich bringen. Dazu gehören verschiedene Methoden durch ein Video zu navigieren und die Auswahl von Videosegmenten, genauso wie das Erstellen von Annotation in unterschiedlichen Formen. Annotationen können neben Text, auch Bilder, Grafiken, Videos oder auch Audioaufnahmen sein. Durch die Fortschritte im Bereich der Bildverarbeitung können Annotationen teilweise automatisch erstellt werden oder das Annotieren zumindest unterstützt werden. Es wurden Ansätze zur automatischen Schnitterkennung, Erkennung von Key-Frames und Objekterkennung vorgestellt. Durch die Verwendung von Ontologien, kann die Annotation sogar teilweise auf semantischer Ebene automatisiert werden. Annotationen dienen auch zum Austausch von Wissen. Um diesen Austausch auch zwischen verschiedenen Software-Plattformen zu ermöglichen wurde das MPEG-7, sowie das *Open Annotation Data Model* vorgestellt.

Zum Abschluss des Kapitels wurde noch auf die Interaktion mit Videos eingegangen. Die Analogien zu Videorekorden scheinen vor allem in Anbetracht der direkten Interaktion auf Geräten mit Touchscreens überholt. Der *PVslider*, sowie der *Twist Lens slider* zeigen, dass es durchaus andere Ansätze zur Interaktion mit Videos gibt

KAPITEL 4

Kollaboratives Arbeiten

Die Zusammenarbeit in Gruppen oder Teams ist ein wichtiger Faktor bei der erfolgreichen Umsetzung von verschiedensten Projekten. Durch die Aufteilung oder das gemeinsame Bearbeiten können Aufgaben oft effektiver und umfassender gelöst werden. Dass Software das kollaborative Arbeiten unterstützen kann wurde früh erkannt. Nicht zuletzt durch die Vernetzung von Computern haben sich in diesem Bereich eine Vielzahl von neuen Möglichkeiten ergeben. Dieses Kapitel beschäftigt sich damit, wie und in welcher Form kollaboratives Arbeiten durch den Einsatz von Software unterstützt werden kann.

Das Kapitel ist in zwei Teile gegliedert. Im ersten Teil werden die theoretischen Grundlagen erarbeitet. Dafür werden zu Beginn die wichtigsten Begriffe im Zusammenhang mit Kollaboration definiert und ihre Bedeutung erklärt, um danach auf die Vorteile sowie Potentiale von kollaborativer Software einzugehen. Außerdem werden die wichtigsten Eigenschaften und verschiedene Einteilungsmöglichkeiten solcher Systeme vorgestellt. Generell ist diese Thematik sehr umfangreich und deshalb kann diese theoretische Einführung nur einen sehr groben Überblick über kollaborative Software geben.

Im zweiten Teil werden dann konkrete Einsatzgebiete und Anwendungsszenarien von kollaborativen Lösungen im für diese Arbeit essentiellen Bereich der Videoannotation vorgestellt. In diesem Zusammenhang werden vier verschiedene Arbeiten besprochen, die unterschiedliche kollaborative Funktionen beinhalten. Anhand dieser Beispiele soll gezeigt werden welche Möglichkeiten sich dadurch ergeben. Speziell im Zusammenhang mit Videos gibt es noch sehr wenig Software die das gemeinsame Bearbeiten und Annotieren ermöglicht, obwohl dies auf jeden Fall sinnvoll wäre.

4.1 Grundlagen & Definitionen

Zu Beginn dieses Kapitels soll geklärt werden, wie der Begriff des kollaborativen Arbeitens definiert ist und was in dieser Arbeit darunter verstanden wird. Das Wort Kollaboration besitzt lateinische Wurzeln und bedeutet ganz allgemein Zusammenarbeit. Die Vorsilbe "Ko"kommt auch in einigen ähnlichen Begriffen, sowie Synonymen vor und kann unterschiedliche Bedeutungen wie: "mit", ßusammenöder "gemeinsam"haben [63]. Der Duden [64] führt als Synonyme für Kollaboration folgende Wörter auf: Kooperation, Teamwork, Zusammenarbeit. Die im Duden angeführte Bedeutung der Zusammenarbeit gegen das eigene Interesse mit einem Kriegsgegner hat historische Gründe. Der Begriff Kollaboration wird in der heutigen Literatur wertfrei verwendet. Kollaboration bedeutet somit Zusammenarbeit. Allerdings gibt es noch weitere und genauere Definitionen, sowie Erklärungen des Begriffs.

> *"Collaboration is a process through which parties who see different aspects of a problem can constructively explore their differences and search for solutions that go beyond their own limited vision of what is possible."* [65]

> *"... collaboration is the process of shared creation: two or more individuals with complementary skills interacting to create a shared understanding that none had previously possessed or could have come to on their own. Collaboration creates a shared meaning about a process, a product, or an event. In this sense, there is nothing routine about it. Something is there that wasn't there before."* [66]

Durch kollaboratives Arbeiten wird also das gemeinsame Lösen von Problemen durch die verschiedenen Herangehensweisen und unterschiedlichen Fähigkeiten von mehreren Personen ermöglicht. Dadurch können Lösungen entstehen, die ohne diese Zusammenarbeit nicht möglich gewesen wären.

Neben dem Begriff der Kollaboration stößt man in diesem Zusammenhang auch oft auf die Begriffe Kooperation oder kooperatives Arbeiten. Wie bereits erwähnt führt der Duden Kooperation als Synonym für Kollaboration. In der Literatur ist dies nur bedingt der Fall. Teilweise wird zwischen den beiden Begriffen durchaus unterschieden. In beiden Fällen handelt es sich um eine Form der Zusammenarbeit. Allerdings kann kooperatives Arbeiten vom kollaborativen Arbeiten durch die Art der Arbeitsteilung unterschieden werden [67]. Beim kooperativen Arbeiten wird die Aufgabe in mehrerer Teilaufgaben gegliedert. Diese Teilaufgaben werden auf unterschiedliche Personen aufgeteilt und in weiterer Folge großteils individuell gelöst. Im Gegensatz dazu steht beim kollaborativen Arbeiten die gemeinsame Bearbeitung der Aufgabe oder des Problems im Vordergrund. Im Weiteren wird in dieser Arbeit zwischen den beiden Begriffen nicht mehr streng unterschieden, die unterschiedlichen Bedeutungen sollen allerdings demonstrieren, dass es unterschiedliche Arten der Zusammenarbeit gibt.

Die Zusammenarbeit von mehreren Personen ist in vielen Bereichen wichtig. Durch die Erfindung des Computers und des Internets entstanden neue Möglichkeiten, um das kollaborative Arbeiten mit Hilfe dieser Technologien zu unterstützen.

4.2 CSCW & Groupware

Im Jahr 1984 veranstalteten Greif I. und Cahsman P. einen Workshop unter dem Titel "Computer-Supported Cooperative Work-[68]. Dieser Begriff bzw. dessen Abkürzung CSCW fand regen Anklang und bezeichnet inzwischen ein sehr umfangreiches, interdisziplinäres Forschungsgebiet. Dieses beschäftigt sich mit der Verbesserung der Effizienz und Flexibilität von Gruppenarbeit durch den gezielten Einsatz von Computersystemen und den damit verbundenen Technologien. Dabei liegt der Fokus nicht ausschließlich auf der Entwicklung von geeigneter Hard- sowie Software, sondern es wird genauso darauf geachtet, die Dynamiken des Arbeitens in Gruppen zu verstehen und zu erforschen. Aus diesem Grund umfasst das Forschungsgebiet CSCW auch Bereiche wie Soziologie und Psychologie.

Die Ergebnisse und Erkenntnisse, die aus der Forschung auf dem Gebiet der CSCW gewonnen werden, werden verwendet, um Groupware zu entwickeln. Gerlicher A. [69] definiert Groupware folgendermaßen:

> *"Der Begriff Groupware bezeichnet ein aus Software und eventuell spezifischer Hardware bestehendes System, das die Zusammenarbeit im Team durch die Schaffung von Kommunikations- und/oder Koordinationslösungen unterstützt oder ermöglicht."*

Andere, gebräuchliche Bezeichnungen für Groupware sind CSCW-Systeme, kollaborative Software oder auch kooperative Software. Groupware bietet im Gegensatz zu herkömmlicher Single-User Software unterschiedliche Vorteile und Potentiale. Gerlicher A. [69] zählt dabei auf folgende Punkte auf:

- *Verbesserung der Kommunikation innerhalb einer Gruppe. Durch Groupware soll schneller, klarer und überzeugender kommuniziert werden können*
- *Kommunikation auch dort zu ermöglichen, wo es sonst nicht möglich wäre*
- *Telearbeit zu ermöglichen (bei geographisch entfernten Usern)*
- *Einsparung von Reisekosten*
- *Bündelung von Fachwissen und dem Gedankenaustausch in der Gruppe*

- *Gruppen mit gemeinschaftlichem Interesse zu bilden*
- *Zeit- und Kostenreduktion bei der Koordination von Gruppenarbeit*
- *Problemlösung in der Gruppe*
- *Erlangung neuer Kommunikationsmöglichkeiten, wie zum Beispiel dem anonymen oder strukturierten Informationsaustausch*

Durch diese Potentiale haben Groupware Systeme ein umfangreiches Anwendungsgebiet. Dazu zählen unter anderem die Softwareentwicklung, E-Learning, Projektmanagement, Produktdesign und Marketing. Anhand dieser Beispiele wird deutlich, dass die möglichen Anwendungen von Groupware sehr breit gefächert sind. Daraus ergibt sich auch eine der größten Herausforderungen bei der Entwicklung von erfolgreichen Groupware Systemen. Ein entscheidender Faktor ist nämlich die Akzeptanz durch die Zielgruppe [69]. Da es sich je nach Groupware um sehr verschiedene Gruppen und in weiterer Folge unterschiedlichen Gruppendynamiken handeln kann, ist es wichtig besonderen Wert darauf zu legen, diese genau zu analysieren und zu verstehen.

Groupware kann nach unterschiedlichen Kriterien eingeteilt werden. Diese Kriterien werden in weitere Folge vorgestellt, um gleichzeitig einen Überblick über die wichtigsten Eigenschaften sowie Herausforderungen von Groupware zu geben.

Groupware-Matrix

Eine der ersten und am häufigsten verwendeten Einteilungen von Groupware stammt von Johansen R. [71]. Er unterteilt Groupware in die zwei Dimensionen Raum und Zeit. Dadurch entstand die Groupware oder auch CSCW-Matrix oder Time/Space Matrix (Abbildung 4.1). Die Matrix ist in vier Bereiche aufgeteilt, welche die unterschiedlichen Varianten der Zusammenarbeit darstellen. Dabei wird beim räumlichen Bereich zwischen am selben Ort (colocated) und verschiedenen Orten (remote) unterschieden. Beim zeitlichen Bereich wird zwischen zur gleichen Zeit (synchronous) und zu verschiedenen Zeiten (asynchronous) unterschieden. Durch diese Aufteilung können verschiedene Systeme in einen der vier Bereiche eingeordnet werden. Software für Videokonferenzen ist ein Beispiel für eine Interaktion die zur gleichen Zeit, also synchron, allerdings an verschiedenen Orten stattfindet. Im Gegensatz dazu findet beispielsweise die Kommunikation bei E-Mail zwar auch an verschiedenen Orten statt, allerdings nicht zwangsweise zur gleichen Zeit.

Die Einteilung durch die Groupware-Matrix ist allerdings nur bedingt ideal. Viele Groupware Systeme lassen sich nicht eindeutig in einen der vier vorgegebenen Bereiche einordnen und auch die Unterscheidung zwischen synchron und asynchron ist nicht immer eindeutig [72]. Aus diesem Grund gibt es Ansätze die Groupware-Matrix zu erweitern. Grudin J. [73] führt

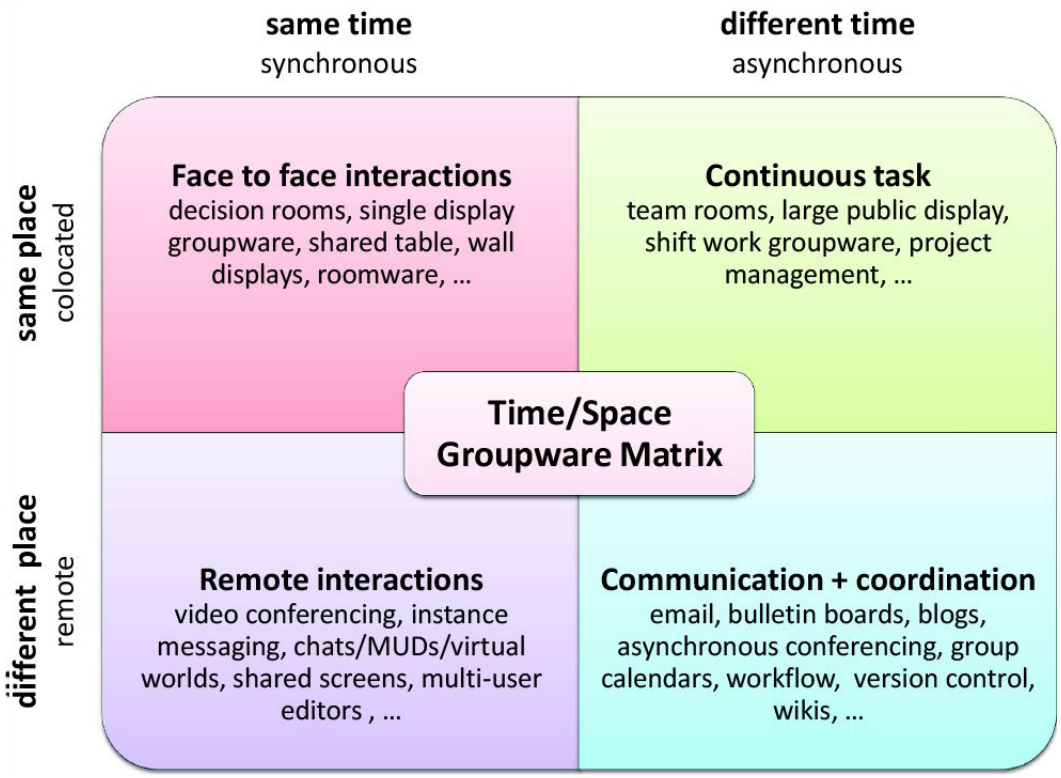

Abbildung 4.1: Die Groupware-Matrix [70]

den Begriff der Vorhersehbarkeit ein und teilt dadurch die Kategorien unterschiedlicher Ort und Zeit, zusätzlich in jeweils zwei Kategorien, so dass insgesamt neun Bereiche entstehen. Vorhersehbar bzw. nicht vorhersehbar bedeutet in diesem Fall, ob es für die jeweilige Aktivität einen gewissen zeitlichen, oder räumlichen Rahmen gibt. Als Beispiel für zeitlich vorhersehbar nennt Grudin E-Mail, da man üblicherweise erwarten kann, dass ein E-Mail innerhalb von wenigen Tagen gelesen wird. Auf der anderen Seite fällt dass kollaborative Schreiben eines Dokuments in die Kategorie zeitlich unvorhersehbar, solange keine Frist zur Fertigstellung vorgegeben ist. Aber auch diese Einteilung löst das Problem der Überschneidung der verschiedenen Bereiche nicht.

Die Einteilung von Groupware in genau einen Bereich der Groupware-Matrix ist allerdings nicht unbedingt nötig. Teufel S. [74] ist davon überzeugt, dass es durchaus zu Überschneidungen kommen kann und soll. Aus diesem Grund sollte die Matrix mehr als eine dynamische Form der Einteilung betrachtet werden, die durchaus auch durch zusätzliche Kriterien erweitert werden kann. Ein Beispiel dafür ist die genauere Unterscheidung der Gruppengröße, in der die Zusammenarbeit stattfindet [75]. Es kann auch durchaus ein Ziel von Groupware Systemen sein, möglichst viele Bereiche der Groupware-Matrix abzudecken, um damit eine besonders

flexible Arbeitsweise zu erlauben.

3K-Modell

Eine weitere in der Literatur beschriebene Einteilung ist das 3K-Modell [74]. Dabei wird Groupware durch die von ihr unterstützen Funktionen eingeteilt. Die drei Kategorien, die dem 3K-Modell ihren Namen geben, sind Kommunikation, Koordination und Kooperation, wobei Kooperation in diesem Fall gleichbedeutend mit Kollaboration ist. Groupware Systeme werden je nachdem, wie umfangreich sie eine dieser Kategorien unterstützen, in eine Systemklasse eingeteilt. Die Systemklassen sind Kommunikation, gemeinsame Informationsräume, Workflow-Mangement und Workgroup-Computing. Wobei ein Groupware System durchaus in mehre Systemklassen fallen kann. Die Klasse Kommunikation umfasst Systeme, die zum Austausch zwischen den einzelnen Personen einer Gruppe dienen. In die Klasse der gemeinsamen Informationsräume fallen Groupware Anwendungen, die den längerfristigen und persistenten Austausch von Informationen innerhalb einer Gruppe von Personen ermöglichen und den Zugriff auf diese Informationen erlauben. Die Workflow-Mangement-Systemklasse umfasst Systeme, die dabei helfen, dass Gruppenaktivitäten unter gewissen festgelegten Kriterien ablaufen. Diese können beispielsweise von einem Unternehmen intern festgelegte Prozesse oder Abläufe sein. Die vierte Systemklasse, Workgroup-Computing sind Systeme, bei denen die kollaborative Zusammen-

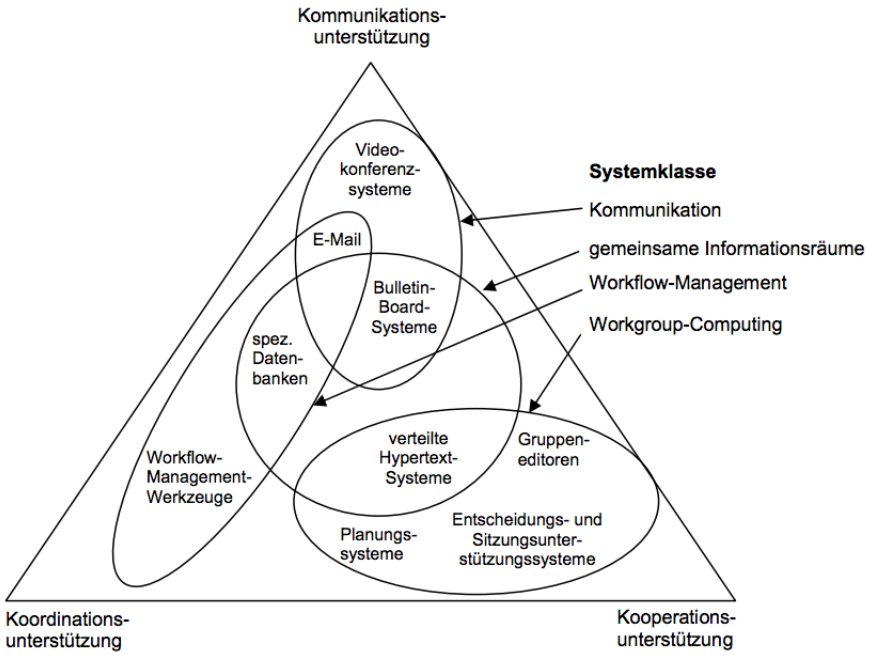

Abbildung 4.2: Das 3K-Modell [74]

arbeit zwischen einzelnen Gruppenmitgliedern im Vordergrund steht. Abbildung 4.2 zeigt eine grafische Repräsentation des 3K-Modells mit einigen Beispielen für die unterschiedlichen Systemklassen.

Eine Abwandlung des 3K-Modells beschreibt Cook N. [76]. Der 4Cs-Ansatz teilt Groupware anhand der folgenden vier ähnlichen Kategorien ein: Kommunikation, Kooperation, Kollaboration und Connection. Kommunikation und Kollaboration entsprechen dabei den gleichen Funktionen, die auch beim 3K-Modell zu finden sind. Cook unterscheidet bei seinem Ansatz, wie bereits besprochen, strenger zwischen Kooperation und Kollaboration. Unter Kooperation wird der Austausch von Daten zwischen Personen verstanden, allerdings nicht das gemeinsame Arbeiten oder Erstellen von Dokumenten. Der hauptsächliche Unterschied beim 4Cs-Ansatz liegt allerdings in der Connection-Kategorie. Diese bringt mögliche soziale Komponenten von Groupware ins Spiel. Darunter fallen Technologien, die es Personen ermöglichen eine Verbindung mit Inhalten oder anderen Personen aufzubauen. Ein populäres Beispiel dafür sind soziale Netzwerke. Diese, vor allem in den letzten Jahren sehr relevant gewordene, Kategorie ist im Gegensatz dazu im 3K-Modell nicht abgebildet.

Asynchron, synchron und multisynchron

Durch die Groupware-Matrix wurden die beiden Begriffe asynchron und synchron bereits eingeführt. In diesem Abschnitt wird allerdings noch einmal kurz darauf eingegangen, weil sie im Zusammenhang mit Groupware oft auftauchen und wichtige, unterschiedliche Anforderungen darstellen. Asynchron bedeutet, dass der Datenaustausch, sowie die Kommunikation zeitversetzt stattfindet. Das heißt, es besteht kein Anspruch auf einen Austausch in Echtzeit. Für viele Systeme ist dies aber auch nicht zwingend notwendig. In welchem Zeitrahmen der Datenaustausch stattfindet ist nicht festgelegt und kann von verschiedenen Faktoren abhängig sein. Bekannte Beispiele für asynchrone Systeme sind E-Mail, Internetforen oder Gruppenkalender. Für die Umsetzung von asynchronen Systemen wird oft eine Client-Server Architektur verwendet wird. Durch einen vorhanden Server können die Daten und Informationen von verschiedenen Personen zu beliebigen Zeiten abgerufen werden.

Bei synchronen Systemen ist es im Gegenteil dazu essentiell, dass der Datenaustausch zur gleichen Zeit stattfindet und dadurch gleichzeitiger, also synchroner, Zugriff auf Daten und Informationen durch mehrere Personen gewährleistet ist. Beispiele dafür sind Anwendungen zur Videotelefonie, Chats oder Remote-Desktop Systeme. Die Gewährleistung der Echtzeitanforderung ist für die Funktion solcher Systeme essentiell.

Neben asynchron und synchron gibt es noch zusätzlich die Möglichkeit von multisynchronem Austausch [69]. Darunter versteht man Systeme, bei denen mehrere Personen gleichzeitig auf ein Dokument oder Daten zugreifen, allerdings jeder User eine eigene, lokale Kopie erhält. Änderungen werden zuerst lokal vorgenommen und erst später mit den Änderungen der ande-

ren User zusammengeführt. Dieses Prinzip kommt unter anderem bei Versionsverwaltungssystemen zum Einsatz. Der Vorteil dabei liegt für den User darin, dass das System die Konsistenz zwischen den unterschiedlichen Versionen gewährleistet. Außerdem wird oft eine Funktion angeboten um auf ältere Versionen einer Datei zuzugreifen und diese unter Umständen auch wieder herzustellen. Multisynchron kann als Kombination von asynchron und synchron gesehen werden, bei der das Umschalten zwischen den beiden Modi automatisch passiert.

Awareness

Um erfolgreiches Arbeiten in eine Gruppe gewährleisten zu können, ist es wichtig, dass sich alle Gruppenmitglieder der Aktionen und Tätigkeiten der anderen bewusst sind. Beim Arbeiten am gleichen Ort, beispielsweise innerhalb eines Büros, ist dies einfacher zu gewährleisten oder sogar automatisch und intuitiv gegeben. Bei Groupware ist dieses Bewusstsein allerdings schwieriger umzusetzen und es muss bei der Umsetzung eines solchen Systems speziell darauf geachtet werden. Im Zusammenhang mit Groupware hat sich als Bezeichnung dafür der englische Begriff Awareness durchgesetzt. Greenberg et al. [77] unterscheiden dabei zwischen vier verschiedenen, teilweise überlappenden, Kategorien der Awareness.

- *Informal Awareness* - bezeichnet das Bewusstsein über die anwesenden Personen in einer Gruppe und deren aktuelle Tätigkeit. Diese Information ist beim Arbeiten am gleichen Ort zu einem gewissen Maß zwangsläufig gegeben. Bei Verwendung von Groupware ist das aber nicht der Fall.

- *Social Awareness* - beschreibt das Wissen über das emotionale und soziale Befinden der anderen Gruppenmitglieder. Dazu gehört zum Beispiel, ob man die Aufmerksamkeit einer Person hat oder deren aktuellen Gefühlszustand. Solche Informationen werden üblicherweise durch Körpersprache oder Blickkontakt kommuniziert. Diese Möglichkeiten fallen allerdings weg, sobald die Gruppenarbeit im virtuellen Raum stattfindet.

- *Group-structural Awareness* - dabei geht es darum, sich über die Strukturen und Hierarchien innerhalb eine Gruppe im Klaren zu sein. Wie sind verschiedene Aufgabe verteilt und was fällt in wessen Verantwortungsbereich sind Fragen, die in diese Kategorie fallen. Auch zum Kommunizieren dieser Information sollten Groupware-Systeme geeignete Mittel anbieten.

- *Workspace Awareness* - durch diesen Begriff wird das Wissen über die Interaktionen der Personen in einer Gruppe mit dem gemeinsamen Workspace bezeichnet [78]. Ein Workspace in der realen Welt kann eine Magnettafel oder ein Tisch sein, und beim gemeinsamen Arbeiten werden die Interaktionen in Echtzeit wahrgenommen. Bei Groupware Systemen gestaltet sich das eher schwierig. Jeder User hat immer nur einen begrenzten

Einblick auf den Workspace und beim Umgang damit gibt es wesentliche Unterschiede. So müssen natürliche Dinge, wie beispielsweise kurze Blicke, die üblicherweise ausreichen, um Workspace Awareness zu gewährleisten, durch langsame Interaktionen, wie das Scrollen am Bildschirm, ersetzt werden. Gutwin C. & Greenberg S. [78] haben sich ausgiebig mit der Workspace Awareness auseinandergesetzt und wichtige Elemente, sowie dazugehörige Fragestellungen erarbeitet, die dabei helfen sollen, dieses Bewusstsein zu gewährleisten (Tabelle 4.1). Dabei sind die wesentlichen Fragen, was während der Zusammenarbeit passiert und wo es passiert. Diese Elemente und Fragen bilden ein Framework, auf das bei der Erstellung von Groupware zurückgegriffen werden kann. Dabei muss überlegt werden, welche Informationen über die Interaktion der einzelnen User erfasst werden sollen und wie diese Information an die anderen User weitergeleitet und präsentiert werden soll.

Element	**Wichtige Fragen**
Präsenz	Welche Personen sind in die Interaktion involviert?
Ort	An welchem Ort findet die Interaktion statt?
Aktivitätslevel	Wie aktiv sind die Gruppenmitglieder innerhalb des Workspaces?
Aktionen	Was sind die derzeitigen Aktivitäten und Aufgaben der einzelnen Gruppenmitglieder?
Absichten	Was werden sie als nächstes tun?
Änderungen	Welche Änderungen werden wo gemacht?
Objekte	Welche Objekte werden zum Interagieren verwendet?
Reichweite	Was können die anderen Gruppenmitglieder sehen?
Fähigkeiten	Was sind die Fähigkeiten der einzelnen Gruppenmitglieder?
Einflussbereich	In welchem Ausmaß können sie Änderungen vornehmen? Welche Berechtigungen besitzen sie?
Erwartungen	Was wird von von den anderen Gruppenmitgliedern erwartet? Was sind die nächsten Interaktionen?

Tabelle 4.1: Workspace Awareness nach Gutwin C. & Greenberg S. [78]

4.3 Mögliche Anwendungsgebiete in Bezug auf Videoannotation

Kollaboration kann auch im Bereich der Videoannotation sinnvoll sein. Die möglichen Einsatzgebiete und Anwendungsszenarien solcher Systeme sind umfangreich. So sind unter anderem Lösungen in Bereichen wie E-Learning, Sportanalyse, Forschung und Medizin denkbar. Aber auch das Potential in der alltäglichen Verwendung von Video und Fernsehen ist zu erwähnen. So könnten kollaborative Systeme es maßgeblich erleichtern über Videos zu diskutieren und

einen allgemeinen Austausch fördern. Um die verschiedenen Möglichkeiten zu zeigen, stellt dieser Abschnitt verschiedene konkrete Arbeiten vor, deren Schwerpunkt auf den kollaborativen Eigenschaften und Funktionen liegt.

Kollaborative Videannotation zum Erstellen von Ground-Truth-Daten

Kavasidis et al. [79] haben eine Webanwendung entwickelt, die durch einen kollaborativen Ansatz dabei hilft, Ground-Truth-Datensätze aus größeren Videoarchiven zu generieren. Ground-Truth-Daten spielen bei der automatischen Videoanalyse eine wichtige Rolle. Die Bereiche, in denen bereits automatische Videoanalyse eingesetzt wird sind vielfältig. Beispiele dafür sind unter anderen die Videoüberwachung aus Sicherheitsgründen, Sportveranstaltungen oder die Beobachtung der Tierwelt. Das Problem dabei ist, dass in vielen Fällen wichtige Referenzdaten fehlen, um eine qualitativ hochwertige Analyse zu gewährleisten. Diese benötigten Referenzdaten werden auch als Ground-Truth-Daten bezeichnet und sind in diesem Fall annotierte Videos, welche den Algorithmen zur automatischen Videoanalyse als Trainingsdaten dienen. Für die Erzeugung von Ground-Truth Daten muss umfangreiches Videomaterial annotiert werden. Dafür wird viel Zeit und ein gewisser Grad an Genauigkeit benötigt. Die Algorithmen funktionieren nämlich umso besser, je besser die erzeugten Ground-Truth Daten sind.

Die Webanwendung von Kavasidis et al. wurde im Zusammenhang mit dem durch die EU geförderten Projekt Fish4Knowledge [80] entwickelt. Fish4Knowledge beschäftigt sich mit der Erforschung des Ökosystems Meer, anhand der automatischen Auswertung von Videomaterial. Um dies zu unterstützen, wurde eine Anwendung entwickelt, deren Ziel es ist, das Annotieren schnell, effektiv und den Useranforderungen entsprechend zu ermöglichen (Abbildung 4.3). Die Anwendung zerlegt die vorhandenen Unterwasseraufnahmen in einzelne Frames. Diese Frames können mit Hilfe von unterschiedlichen Werkzeugen annotiert werden. Es stehen Funktionen zum Auswählen, Zeichnen, Erstellen von Rechtecken und Polygonen, sowie zum Löschen zur Verfügung. Außerdem ist es durch das Kombinieren und Markieren auf mehreren Frames möglich, den Bewegungsverlauf eines Objekts zu definieren. Da auch das Festlegen der Kontur von einzelnen Fischen wichtig ist, wurden dafür automatisierte Ansätze implementiert. Es stehen die aus der Computer-Vision stammenden Algorithmen Grabcut [81], Snakes [82] und Canny [83] für die automatische Extrahierung der Konturen zur Auswahl.

Für die Umsetzung der kollaborativen Funktionen wurde ein asynchroner Ansatz gewählt. Annotationen können gespeichert und danach mit anderen Usern geteilt werden. In diesem Zusammenhang ist es möglich, Ground-Truth-Daten durch die Kollaboration mit anderen Usern zu erstellen, sobald mehrere Annotationen von verschiedenen User zum selben Videomaterial existieren. Dabei wird davon ausgegangen, dass durch eine Kombination von mehreren Annotationen bessere und genauere Daten erstellt werden können. Durch die Kombination der annotierten Konturen von Fischen, lässt sich eine sogenannte beste Ground-Truth berechnen

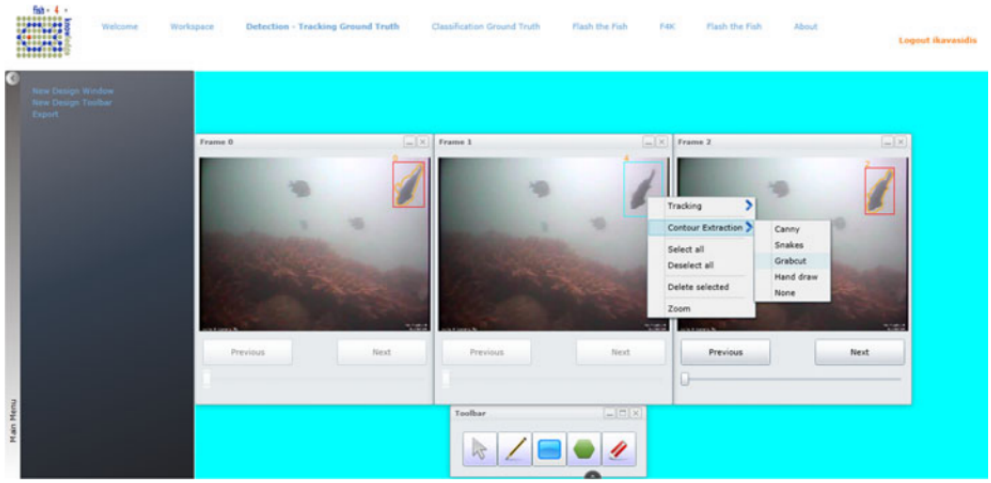

Abbildung 4.3: User-Interface Fish4Knowledge (Webanwendung) [79]

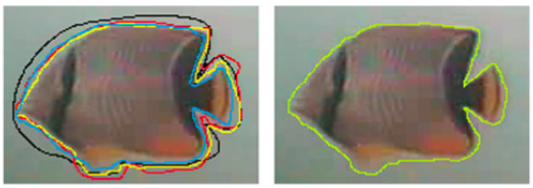

Abbildung 4.4: Beste Ground-Truth für die Konturen von Fischen [79]

(Abbildung 4.4). Der kollaborative Ansatz in der von Kavasidis et al. entwickelten Webanwendung besteht einerseits also daraus, dass das zeitaufwändige und umfangreiche Annotieren der Videos auf mehrere User aufgeteilt werden kann, andererseits wird gleichzeitig die Qualität der Annotationen verbessert. Mit Hilfe der Anwendung konnten im Zuge einer Testphase bereits mehr als 55.000 Annotationen erstellt werden und es sind weitere Verbesserungen der kollaborativen Bestandteile angedacht, welche die Effizienz steigern würden. Dazu zählt unter anderem das gleichzeitige, synchrone Annotieren der Videos von verschiedenen Nutzern oder die Verteilung der zur Ground-Truth-Berechnung benötigten Rechenleistung.

Kollaborative Videoannotation durch das Watch-and-Comment-Paradigma

Davon ausgehend, dass das gemeinsame Ansehen und gleichzeitige Kommentieren, sowie Diskutieren von Video- und Fernsehinhalten eine beliebte Tätigkeit ist, haben Cattelan et al. [84] das Watch-and-Comment (WaC) Paradigma definiert. Sie haben versucht diese Interaktion mit Hilfe eines Programms abzubilden, welches es Personen ohne speziellen Vorwissen ermöglicht, annotierte und interaktive Videos zu erstellen. So können beispielsweise Audiokommentare,

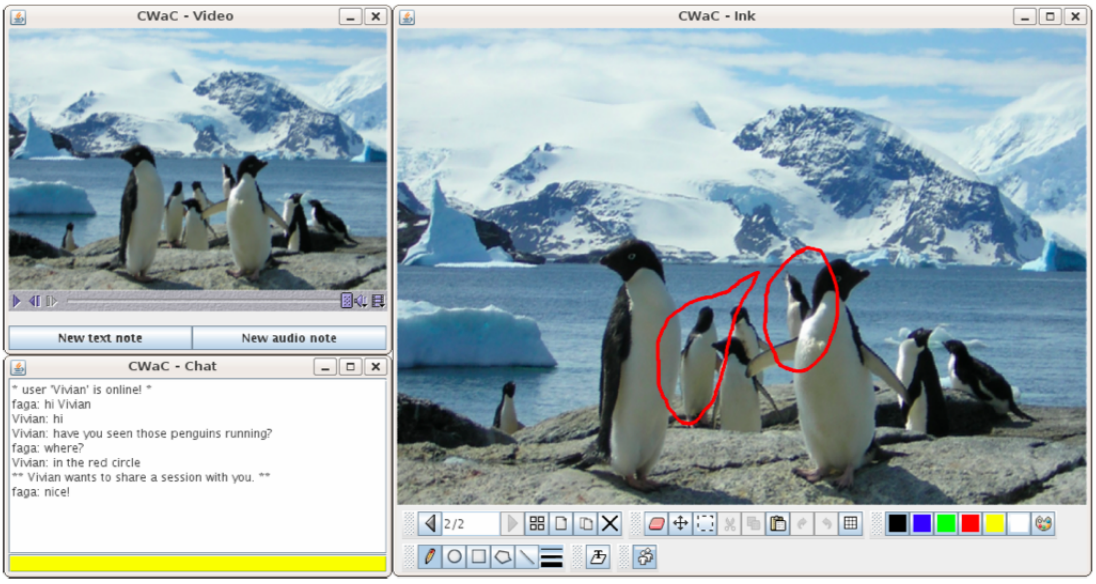

Abbildung 4.5: User-Interface des CWaCTool [84]

die während des Ansehens eines Videos abgegeben werden, automatisch aufgenommen und der dazugehörigen Stelle im Video zugeordnet werden. Das WaCTool [84] erlaubt neben Audioannotation, auch das Annotieren durch Texte und Zeichnungen.

Da das Kommentieren, sowie Diskutieren eines Videos ein Vorgang ist, der meist in einer Gruppe von Personen stattfindet, war es naheliegend das WaCTool mit kollaborativen Funktionen zu erweitern. Aus dieser Überlegung heraus wurde in weiterer Folge ein Prototyp mit dem Namen CWaCTool entwickelt [85]. Das Ziel dabei war es, den Usern das synchrone und gemeinsame Erstellen von Annotation während des Betrachtens von Videos an getrennten geographischen Orten, zu ermöglichen. Ein Anwendungsfall für das CWaCTool ist interaktives Fernsehen, auch iTV oder Social TV genannt. Die Ziele von Social TV sind im Allgemeinen [86]:

- Das Unterstützen der Kommunikation zwischen voneinander entfernten Zusehern
- Die Möglichkeit des direkten und indirekten sozialen Austausches
- Der Aufbau einer sozialen, emotionalen Verbindung
- Die Zuseher dazu ermuntern, eigene Inhalte zu erstellen und diese mit andern zu teilen
- Die Unterstützung von virtuellen Communites

Durch das CWaCTool soll es möglich sein, annotierte Videos auf Fernsehgeräten wiederzugeben, die interaktives Fernsehen unterstützen. Hinsichtlich der Kollaboration unterstützt das Programm zwei verschiedene Varianten:

- *Austausch von annotierten Videos* - Bei dieser Variante kann der User ein beliebiges Video auswählen und es mit Annotationen und Kommentaren versehen. Dieser Vorgang geschieht rein individuell. Nachdem das Video fertig annotiert wurde, besteht die Möglichkeit das Video als Ganzes oder nur die Annotationen, die in einem XML-Format gespeichert werden, auszutauschen. Dieser Austausch kann entweder über einen Server oder auf P2P-Basis abgewickelt werden. In diesem Fall handelt es sich also eindeutig um eine asynchrone Kollaboration. Motti et al. [85] bezeichnen diese Variante auch als sequentielle Kollaboration, da andere User nach dem Austausch in der Lage sind, die Annotationen zu erweitern und zu bearbeiten und das fertige Video als Produkt einer Gruppenarbeit gesehen werden kann. Der Vorteil bei der sequentiellen Arbeit besteht darin, dass es nicht so leicht zu Überschneidungen kommt. Den Usern ist immer ersichtlich, welche Teile des Videos bereits annotiert wurden, dies ist beim gleichzeitigen Arbeiten am Video nicht der Fall. Darunter leidet allerdings der Gedankenaustausch und die Interakation zwischen den Usern.

- *Teilen einer laufenden Annotation Session* - Um die gleichzeitige Interaktion zwischen den Usern des CWaCTools zu forcieren, wurde die zweite Variante entwickelt. Hier ist es möglich eine laufende Annotation Session für andere User freizugeben. Dadurch wird eine synchrone Kollaboration ermöglicht, bei der andere User die erstellten Annotationen einsehen, kommentieren und ergänzen können. Zur Kommunikation unter den Usern steht ein Text-Chat zur Verfügung. Sobald eine Session gestartet wurde, können alle Aktionen gleichzeitig ausgeführt werden und die User sind untereinander über ein P2P-Netzwerk verbunden. Ein Problem bei dieser Variante ist, dass jeder User die Annotationen der anderen User bearbeiten kann. Eine Lösung dafür wäre, einen Moderator für jede Session festzulegen, der in der Lage ist die Rechte der einzelnen User zu bestimmen.

Nach der Entwicklung des Prototypen wurde auch noch eine Evaluierung durchgeführt, bei der 22 User das Programm testet. Vor allem die umfangreiche Funktionalität und die unterschiedlichen Methoden zur Annotation wurden sehr positive bewertet. Auch der Austausch zwischen den Usern über den Chat wurde gut angenommen. Interessant war, dass Tester im Zusammenhang mit der Chat-Funktion vorgeschlagen haben, dass es eine Übersicht der anwesenden User im Chat geben sollte. Dieser Vorschlag steht im engen Zusammenhang mit der, in diesem Kapitel bereits besprochenen, Awareness-Komponente von Groupware-Systemen.

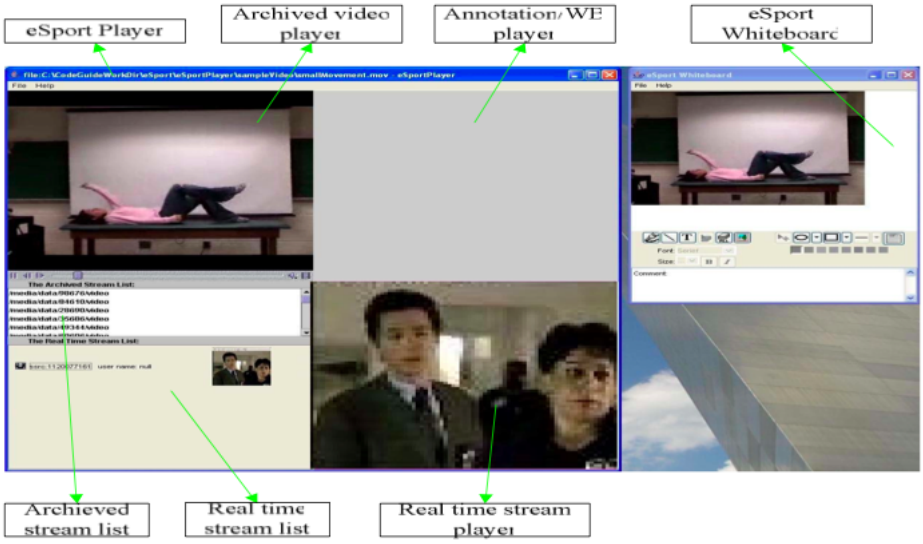

Abbildung 4.6: User-Interface eSprots Player (links) und Whiteboard (rechts) [87]

Kollaborative Videoannotation fürs Sport-Coaching

Zhai et al. [87] haben die Plattform eSports entwickelt, welche das kollaborative und synchrone Annotieren von Videos für Coaching-Zwecke im Sport ermöglicht. Eines der von ihnen beschriebenen Szenarien für eSports ist folgendes. Ein Basketballspiel soll analysiert werden. Ein ausgebildeter Basketballtrainer befindet sich auf einer Universität in den USA, die Studenten auf verschiedenen Universitäten in China. Mit Hilfe von eSports kann der Trainer ein Video des Spiels öffnen und gleichzeitig wird das Video per Internetstream für die Studenten verfügbar. Es ist nun möglich, das Video trotz der geographischen Trennung gemeinsam anzusehen und zu diskutieren. Sobald der Trainer eine interessante Situation bemerkt, die er mit seinen Studenten besprechen will, kann ein Screenshot davon angefertigt werden. Dieser Screenshot kann über ein virtuelles Whiteboard annotiert werden. Annotierte Screenshots werden gespeichert und können zu einem späteren Zeitpunkt sowohl vom Trainer, als auch von den Studenten, noch einmal angesehen werden. Über das Whiteboard können auch Studenten den Screenshot annotieren und ihre Meinung zu der aktuellen Szene abgeben. Für die Handhabung eines solchen Szenarios umfasst die eSports Plattform drei wichtige Komponenten.

- *eSports Player* - Der Player (Abbildung 4.6) dient zum Abspielen und Auswählen von Videos. Mit eSports können nicht nur bereits vorhandene Videos annotiert werden, sondern auch Livestreams. Im Player werden neben dem aktuellen Video auch noch bereits vorhandene Screenshots mit Annotationen angezeigt. Alle User die sich in der gleichen Session befinden sehen den Player im gleichen Zustand. Die Kontrolle über das Video-

Playback hat nur der Trainer. Bei eSports gibt es zwei verschiedene Berichtigungen: Trainer und Student.

- *eSports Whiteboard* - So wie der Player wird auch das Whiteboard (Abbildung 4.6) zwischen den Usern einer Session geteilt. Der Trainer kann von der Player Komponente aus einen Screenshot anfertigen, dieser wird dann am Whiteboard angezeigt. Das Whiteboard verfügt über verschiedene Funktionen zum Annotieren, die sowohl vom Trainer, als auch den Studenten benutzt werden können. Es können Text, Zeichnungen, Formen und Bilder mit Hilfe des Whiteboards hinzugefügt werden. Nur der Trainer ist in weiterer Folge berechtigt den aktuellen Inhalt des Whiteboards zu speichern oder zu löschen. Beim Abspeichern des Inhalts, wird ein JPG-Bild des Whiteboards generiert und dieses zusammen mit dem aktuellen Zeitstempel des Videos abgespeichert.

- *Instant Messenger* - Um die Kommunikation zwischen Studenten und Trainer zu gewährleisten, besitzt eSports auch eine Komponente zum Nachrichtenaustausch. Dafür wurde ein Chat eingebunden, in dem sich alle Nutzer gegenseitig schreiben können und jeder diese Nachrichten lesen kann.

Besonderer Wert wurde bei der Umsetzung von eSports auf die Möglichkeit der Annotation von Livestreams gelegt. Die Architektur der Plattform wurde so ausgelegt, dass sie auch für eine größere Anzahl von Usern skalierbar ist. Generell werden von eSports zwei verschiedene Szenarien unterstützt. Zum einen das bereits beschriebene gemeinsame Ansehen und Annotieren von Videos. Dabei wird gleichzeitig ein annotierter Videostream erzeugt und archiviert. Das zweite Szenario besteht aus dem nachträglichen Ansehen dieses archivierten Streams, inklusive der Annotationen und Kommentare. Dadurch können Studenten aus den aufgezeichneten Annotationen zu einer beliebigen Zeit lernen.

Kollaborative Videoannotation zur Unterstützung von E-Learning

YouTube und andere Videoplattformen haben in den letzten Jahren die Integration von sozialen Netzwerken stark forciert. Allerdings sind die Möglichkeiten zur Annotation und Interaktion noch immer sehr stark eingeschränkt. Genau an dieser Problematik haben Wilk et al. [88] angesetzt und haben ein System entwickelt, das es ermöglicht, Videos mit zusätzlichen Inhalten, durch kollaborative Zusammenarbeit, zu ergänzen. Das Ziel dabei ist es interaktive, soziale Videos zu erschaffen, die vor allem im Bereich des E-Learnings ihre Anwendung finden. Um das Konzept zu überprüfen, wurde ein Prototyp entwickelt (Abbildung 4.7).

Ein Hauptbestandteil des Systems sind sogenannte Videoobjekte. Videoobjekte sind Container und beziehen sich auf ein bestimmtes Objekt im Video. Das kann zum Beispiel eine Person, ein Tier oder ein Gegenstand sein. Videoobjekte werden im Prototyp der Client-Software

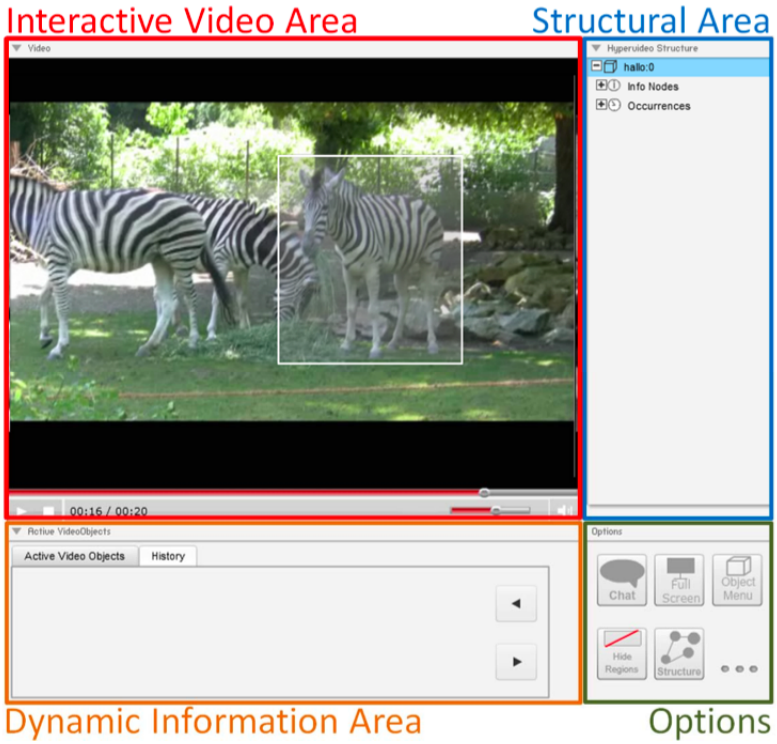

Abbildung 4.7: User-Interface des Prototyps von Wilk et al. [88]

auf verschiedene Arten dargestellt. Zum einen sind Videoobjekte, direkt im Video sichtbar und durch ein Rechteck hervorgehoben. Dieses Rechteck bildet einen Hotspot, welcher angeklickt werden kann und daraufhin weitere Informationen zum verknüpften Videoobjekt liefert. Damit die Markierungen auch dem jeweiligen Objekt im Video folgen, wurden verschiedene Algorithmen zum Object-Tracking aus dem Bereich der Computer-Vision implementiert. Neben dieser direkten Interaktion mit den Videoobjekten, beim Auftreten im Video, werden alle vorhandenen Videoobjekte zusätzlich auch noch in einer Baumstruktur im Client angezeigt. Durch diese Ansicht erhält man eine Übersicht über alle zusätzlich im Video vorhandenen Informationen. Videoobjekte können von den Usern des System erstellt und ergänzt werden. Um ein Videoobjekt zu definieren, muss das Objekt einfach durch ein Rechteck abgegrenzt werden. Danach kann das Videoobjekt in Form von verschiedenen Knoten annotiert werden.

- *Bild-Text-Knoten* - Zusätzliche Information kann als Text und Video durch HTML eingefügt werden.
- *Hyperlink-Knoten* - Einfügen von Links zu weiterführenden externen Webseiten
- *Video-Knoten* - Verknüpfung mit einem anderen sozialen Video innerhalb des Systems

- *Kommunikations-Knoten* - Durch das Hinzufügen eines Kommunikationsknoten wird ein Chat zum jeweiligen Videoobjekt eröffnet. Dort können sich die User austauschen und diskutieren. Interessant an diesem Ansatz ist, dass es keinen allgemeinen Chat gibt, sondern sozusagen einzelne Chaträume, zu den jeweiligen Videoobjekten. Der Inhalt der Chats ist persistent und wird auf einem Server gespeichert.

Administratoren können Videos hochladen, die dann von anderen User bearbeitet und ergänzt werden können. Aber auch die User selbst haben das Recht, Videos hochzuladen und diese zu annotieren. Die Kommunikation und Userverwaltung wird mit Hilfe des sozialen Netzwerkes Facebook abgewickelt.

Abschließend wurde der Prototyp evaluiert. Einerseits wurde getestet wie zufrieden die User mit dem entwickelten System hinsichtlich der Usability und der unterstützen Funktionen sind. Andererseits wurde eine Studie durchgeführt, ob sich die Lernerfolge durch die Benutzung des Systems verbessern lassen. Generell wurde der Prototyp sehr gut angenommen. Interessant war, dass von den verschiedenen Knoten der Kommunikations-Knoten am besten bewertet wurde. Vor allem die Möglichkeit des Hinzufügens von eigenen Annotationen und bei Unklarheiten das Nachfragen über die integrierten Chats bei anderen Usern wurde hervorgehoben. Aus diesem Grund kommen Wilk et al. zu dem Schluss, dass für erfolgreiches Lernen mittels Video die kollaborative Interaktion essentiell ist.

Zusammenfassung

Der Einsatz von kollaborativen Funktionen bietet im Zusammenhang mit Software viele Chancen, erfordert allerdings gleichzeitig ein umfassendes Verständnis der Abläufe innerhalb von Gruppen. Durch die Forschung auf dem Gebiet der CSCW konnten bereits viele Erkenntnisse darüber gewonnen werden, welche Faktoren wichtig sind und beachtet werden müssen. Die verschiedenen Bereiche, wie beispielsweise Soziologie und Psychologie, die dabei eine wichtige Rolle spielen, machen deutlich wie komplex und dynamisch die Zusammenarbeit von mehreren Personen sein kann. Durch die Schaffung von geeigneten Koordinations-, sowie Kooperationsmöglichkeiten in Softwareprojekten können Groupware-Systeme entstehen, die beim gemeinsamen Lösen von Problemen helfen. Im Idealfall entstehen durch gelungene Kollaboration Lösungen, die ohne Zusammenarbeit nicht möglich gewesen wären.

Die Einteilung von Groupware in Zeit/Raum-Abhängigkeit/Unabhängigkeit verschafft einen guten Überblick über die Möglichkeiten und Vorteile, die Groupware-Systeme bieten. Eine strenge Kategorisierung anhand diese Kriterien ist allerdings, ebenso wie durch die Begriffe asychron/synchron nicht möglich und auch nicht notwendig. Es sollte je nach Anwendungsfall abgewogen werden, welche Funktionen Sinn machen. Speziell bei Groupware ist es besonders wichtig, sich mit den potentiellen Anwendern und Anwendungsszenarien auseinander

zu setzen, da ein entscheidender Faktor des Erfolgs von Groupware die Akzeptanz innerhalb der Gruppe ist. Auch das 3K-Modell geht noch einmal auf die verschiedenen Funktionen von Groupware ein und auch hier ist anzumerken, dass die Ausprägung dieser Funktionen von Fall zu Fall unterschiedlich ist. Ein anderer wichtiger Faktor von Groupware-Systemen ist die Awareness. Die diversen, oft als selbstverständlich angesehenen Eigenschaften, die bei der Übertragung der Gruppenarbeit in den virtuellen Raum wegfallen, müssen durch geschicktes Design der Groupware erhalten bleiben.

Dass kollaborative Ansätze auch bei der Videoannotation eine Rolle spielen, zeigen die präsentierten Beispiele. Außerdem zeigen sie, wie unterschiedlich die möglichen Einsatzgebiete sein können. Vom asynchronen Teilen von Annotationen bis hin zum gleichzeitigen Annotieren und Diskutieren von Videos, können kollaborative Funktionen in Software zur Videoannotation umgesetzt werden. Die durch das Angebot solcher Funktionen gewonnen Vorteile sollten bei der Entwicklung auf jeden Fall berücksichtigt werden.

KAPITEL 5

Prototyp

Die Anfertigung von Prototypen spielt in der Softwareentwicklung eine wichtige Rolle. Prototypen helfen dabei, in einer frühen Phase wichtiges Feedback zu sammeln und somit mögliche Lösungsansätze zu validieren. Durch Prototyping ist es möglich verschiedene Interaktionen zu testen und dadurch Probleme frühzeitig zu erkennen. In diesem Kapitel wird die Entwicklung eines Prototyps zum Annotieren von Videos auf einem Tablet beschrieben. Tablets stellen durch die Kombination von Mobilität, Touchscreen und integrierter Kamera eine ideale Plattform für die Umsetzung einer solchen Applikation dar.

Zu Beginn des Kapitels werden die Ergebnisse einer Recherche über ähnliche Projekte präsentiert. Danach wird das Konzept, das als Grundlage für die Entwicklung der ersten Version des Prototyps diente, anhand einer Skizze besprochen und erklärt. In diesem Zusammenhang werden auch die technischen Rahmenbedingungen zur Umsetzung des Konzepts erläutert. Die Umsetzung des Prototyps gliedert sich in zwei Teile. In der ersten Version wird hauptsächlich eine spezielle Funktion umgesetzt. Aus diesem Grund kann die erste Version als eine Art vertikaler Prototyp angesehen werden [89]. Die umgesetzte Funktionalität ist das Annotieren einer bestimmten Stelle in einem Video durch eine Swipe-Geste, direkt auf dem Video. In weiterer Folge werden die beim Test dieses Prototyps aufgetretenen Probleme beleuchtet. Der zweite Teil besteht aus der Weiterentwicklung des Prototyps. Zum Einen wird dabei auf die Probleme der ersten Version eingegangen, zum Anderen werden zusätzliche Features umgesetzt und der Prototyp somit in die horizontale Richtung weiter entwickelt. Durch die Weiterentwicklung soll der Prototyp in ein Stadium gebracht werden das es erlaubt von einem größeren Personenkreis getestet zu werden. Des weiteren werden anhand des User-Interfaces die Funktionen und Interaktionen des zweiten Prototyps veranschaulicht und auch einige technische Details, sowie Herausforderungen, die bei der Entwicklung des Prototyps aufgetreten sind, näher betrachtet. Dies geschieht allerdings ohne explizit auf den Source Code oder bestimmte Algorithmen einzuge-

hen, sondern beschreibt ganz allgemein aufgetretene Probleme, sowie mögliche Lösungsansätze. Zum Abschluss des Kapitels werden noch kurz die Erfahrungen, die beim Veröffentlichen des Prototyps im Apple App Store gesammelt werden konnten, beschrieben.

5.1 Ähnliche Applikationen

Bevor mit der Entwicklung des Prototyps begonnen wurde, wurde zuerst eine Recherche über ähnliche Apps durchgeführt. Zu diesem Zweck wurde sowohl der Apple App Store, als auch der Google Play Store durchsucht. Dabei hat sich gezeigt, dass es sehr wenige Apps gibt, die das Annotieren von Videos ermöglichen. Im Vergleich dazu gibt es unzählige Apps zum Annotieren von Textdokumenten, PDF-Dateien und Bildern.

Eine Kategorie von Apps, bei denen das Annotieren von Videos allerdings eine wichtige Rolle spielt und wo es auch bereits einige Applikationen gibt, ist der Sportbereich. In diesem Bereich existieren Apps die Sportler und Trainer dabei unterstützen genaue Analysen durchzuführen. Einige populäre Beispiele sind: Coach's Eye [90], Ubersense Video Analysis & Coaching [91] und CoachMyVideo Mobile [92]. Diese Apps sind ähnlich aufgebaut und besitzen zum Großteil auch den gleichen Funktionsumfang. Abbildung 5.1 zeigt das User-Interface von CoachMyVideo. Man kann entweder ein oder zwei Videos öffnen. Wenn zwei Videos geladen werden, werden diese nebeneinander angezeigt und synchron wiedergegeben. Diese Funktion ermöglicht den direkten Vergleich mit anderen Sportlern, oder den Vergleich mit einer anderen Aufnahme, des gleichen Athleten. Videos können direkt mit dem iPad aufgenommen, aus der Foto-Bibliothek importiert oder durch einen Link aus dem Internet heruntergeladen werden. Neben der normalen Wiedergabe der Videos kann auch Frame für Frame zurück oder nach vorne gesprungen werden. Diese Funktion ist im Sportbereich besonders wichtig, da viele Bewegungen sehr schnell ausgeführt werden. Zum Annotieren des Videos stehen einfache Zeichenwerkzeuge zur Verfügung. Damit können Rechtecke, Kreise und Linien direkt auf das Videobild gezeichnet werden. Auch das Zeichnen von Freihandlinien ist damit möglich. Die Annotation kann in Kombination mit einem Videoframe als Bild exportiert werden. Dabei ist sie aber nicht einem bestimmten Frame, oder einer bestimmten Zeitspanne zugeordnet, sondern dem Video nur überlagert. Es kann auch immer nur eine Annotation erstellt und angezeigt werden. Das Video selbst kann allerdings nur ohne Annotation exportiert werden. Im Unterschied dazu bieten Coach's Eye und Ubersense die Möglichkeit die Annotationen aufzunehmen und als komplettes Video zu exportieren. Dabei kann neben den gezeichneten Annotationen auch Audio aufgenommen werden. Diese Funktion ermöglicht es Trainern, ihre Analyse samt gesprochenen Kommentaren komplett aufzuzeichnen und diese mit Sportlern zu teilen. Auch bei Coachs' Eye und Ubersense ist es nicht möglich, mehrere Annotationen zu unterschiedlichen Stellen im Video zu erstellen. Coach's Eye ist sowohl für iOS als auch Android verfügbar, Ubersense und CoachMyVideo nur für iOS.

Abbildung 5.1: Screenshot der Tablet-App CoachMyVideo [92]

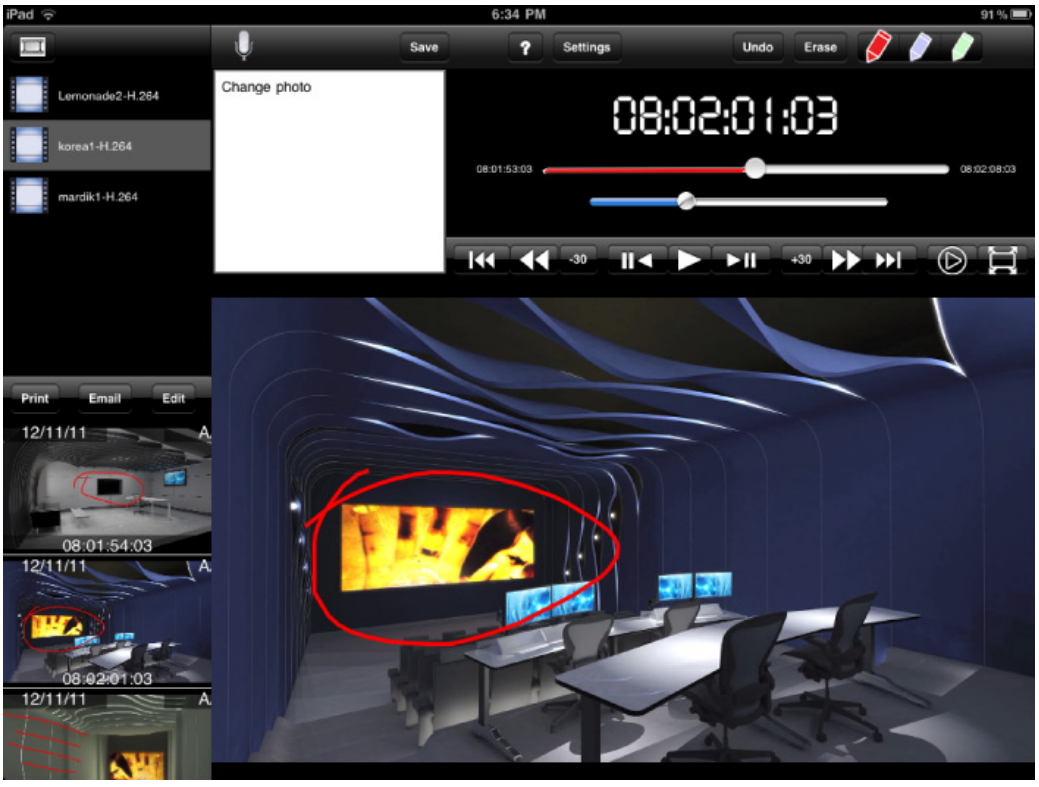

Abbildung 5.2: Screenshot der Tablet-App Scribbeo [93]

Neben den Apps im Sportbereich wurde im Zuge der Recherche eine weitere App entdeckt, welche für ein breiteres Spektrum an Anwendungen konzipiert wurde. Scribbeo [93] (Abbildung 5.2) ermöglicht das Annotieren von Videos mit Hilfe von Text, Audio, sowie Zeichnugen und ist für Videoproduzenten entwickelt worden. Videos können auf drei verschiedene Arten importiert werden. Über die Fotobibliothek des iPads, über die iTunes FileSharing Funktion und von einem Dropbox Account. Sobald ein Video in Scribbeo geladen wurde, stehen diverse Playback Funktionen zur Verfügung um das Video wiederzugeben. Neben dem Vor- und Zurückspulen, gibt es die Möglichkeit das Video Frame für Frame in Zeitlupe abzuspielen und zusätzlich ist es möglich, in 30s Schritten durch das Video springen. Eine Annotation kann nur dann erstellt werden, wenn das Video pausiert wurde. Daraus ergibt sich auch die Einschränkung, dass eine Annotation immer genau einem Frame zugeordnet ist. Mit Scribbeo ist es nicht möglich eine Annotation einer Zeitspanne im Video zuzuordnen. Es gibt drei verschiedene Möglichkeiten, um ein Videoframe zu annotieren: Durch eine Audioaufnahme mit Hilfe des im iPad integrierten Mikrofons, durch Eingabe von Text in einem Textfeld oder durch das direkte Zeichnen auf das aktuelle Videoframe. Zum Zeichnen stehen die Farben Rot, Blau und Grün zur Verfügung. Im Gegensatz zu den Sport-Apps, gibt es bei Scribbeo keine zusätzlichen Zeichenwerkzeuge. Nach dem Erstellen kann die Annotation gespeichert werden. Zu jeder Annotation wird zusätzlich das aktuelle Datum, der aktuelle Zeitpunkt im Video und der Autor abgespeichert. Die Annotationen werden danach im linken unteren Eck der App, in chronologischer Reihenfolge, aufgelistet. Sobald man eine der Annotationen auswählt, wird diese wieder angezeigt und es wird automatisch zu der Position im Video gesprungen, bei der die Annotation erstellt wurde. Es ist nicht möglich, die gespeicherten Annotationen bei der Wiedergabe eines Videos automatisch anzuzeigen. Die Annotationen können in einer tabellarischen Übersicht ausgedruckt oder per E-Mail verschickt werden. Dabei wird jeweils ein Bild des Frames mit der darauf gezeichneten Annotation neben der textuellen Annotation und dem Timestamp dargestellt. Außerdem ist es über einen Netzwerk Modus möglich Annotationen mit anderen Personen über einen eigenen Server auszutauschen und somit gemeinsam an einem Video zu arbeiten. Scribbeo wird allerdings nicht mehr weiter entwickelt. Die aktuelle Version der App wurde im Mai 2012 veröffentlicht und das letzte Update der Homepage fand im März 2012 statt. Scribbeo ist nur für iOS verfügbar und es gibt neben der Version für das iPad auch eine iPhone Version.

Andere Anwendungen konnten im Zuge der Recherche nicht gefunden werden. Diese Erkenntnis deckt sich auch mit der Aussage von Lux A. & Riegler M. [47], die in ihrer Arbeit über das Annotieren von Endoskopievideos anmerken, dass im Moment keine Apps zum Annotieren von Videos im Google Play Store existieren.

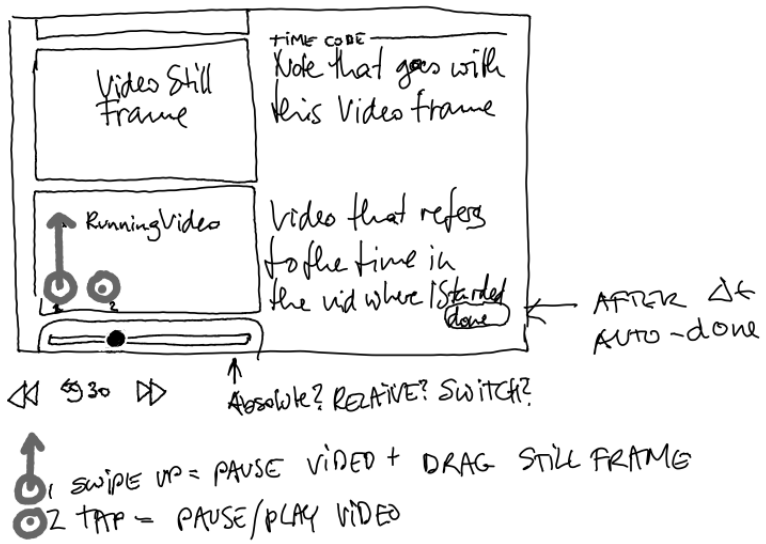

Abbildung 5.3: Skizze des Konzepts für den Prototyp

5.2 Das Konzept

Abbildung 5.3 zeigt eine Skizze des ursprünglichen Entwurfs für das Interaktionsdesign einer Tablet-App zum Annotieren von Videos. Diese Skizze war der Ausgangspunkt zur Entwicklung der ersten Version des Prototyps. Der Entwurf ist in zwei Bereiche unterteilt. Das zu annotierende Video befindet sich links unten und nimmt in etwa ein Viertel des Bildschirms ein. Dort befinden sich auch diverse Playback-Elemente zum Steuern des Videos. Sobald direkt auf dem Video eine Swipe-Geste ausgeführt wird, wird das Video pausiert und das aktuelle Videoframe erscheint links oben. Auf der rechten Seite kann eine Notiz zur aktuellen Stelle im Video verfasst werden. Sobald die Annotation fertig ist, kann sie gespeichert werden. Zu jeder Annotation wird zusätzlich auch die zugehörige Zeit im Video gespeichert. Dadurch kann jede Annotation genau einer Stelle im Video zugeordnet werden. Dieser Zusammenhang ist insofern wichtig, als dadurch die einzelnen Annotationen auch als eine Art Lesezeichen im Video dienen. Sobald eine Annotation gespeichert wurde, läuft das Video weiter. Links oben werden weiterhin die Videoframes angezeigt, zu denen eine Annotation erstellt wurde. Durch Scrollen gelangt man zu älteren Notizen und kann diese wieder anzeigen.

Die Entwicklung des Prototyps dient dazu das Konzept zu evaluieren. Beim Entwurf wurde darauf wert gelegt die Interaktion einfach zu gestalten und den Funktionsumfang ausschließlich auf die Kernaufgabe der Annotation zu reduzieren.

5.3 Technische Umsetzung

Die App wurde von Beginn an als Applikation für Tablets konzipiert. Als Plattform zur Umsetzung des Projektes wurde das Apple iPad ausgewählt. Grund dafür war, neben der teilweise bereits vorhandenen Erfahrung mit der Programmiersprache Objective-C, das umfangreiche SDK für iOS [94]. Dieses SDK enthält zahlreiche Frameworks und Bibliotheken, die die Entwicklung beschleunigen. Das ist speziell beim Erstellen eines Prototyps ein großer Vorteil. Der Prototyp wurde für iOS Version 6 und aufwärts entwickelt, als Entwicklungsumgebung kam XCode 4.6 zum Einsatz. XCode beinhaltet einen Simulator, mit dem ein iPad am Computer simuliert werden kann und man dadurch nicht immer direkt am Gerät testen muss. Auch das Abspielen von Videos funktioniert mit diesem Simulator problemlos. Im Vergleich dazu bietet das Android SDK [95] ebenso einen Emulator, allerdings ist dort das Abspielen von Videos nicht ohne Schwierigkeiten möglich. Da die Wiedergabe von Videos einen Hauptbestandteil des Prototyps darstellt, war diese Funktion des XCode Simulators auch ein Entscheidungskriterium für die Auswahl der Plattform. Der Prototyp wurde hauptsächlich auf einem iPad 3 getestet, sollte aber auch auf allen anderen bisher erschienen iPad Varianten funktionieren, mit Ausnahme des ersten, da dieses von iOS 6 nicht mehr unterstützt wird. Die erste Version des Prototyps orientiert sich direkt an dem zuvor vorgestellten Entwurf. Die App besteht nur aus einem einzigen Bildschirm. Beim Start wird ein Video geladen, das direkt im Sourcecode angegeben ist und somit nicht durch den User geändert werden kann. Das ist allerdings in diesem Stadium nicht wichtig, da ausschließlich die grundsätzliche Idee, sowie das Interaktionsdesign erprobt werden sollen. Abbildung 5.4 zeigt das User-Interface des ersten Prototyps. Zum Abspielen des Videos wird die *MPMoviePlayerController*-Klasse aus dem *MediaPlayer* Framework verwendet. Sobald eine Drag-Geste vom Video in Richtung des linken, oberen Bereichs ausgeführt wird, wird das Video pausiert und gleichzeitig das aktuelle Frame oberhalb in einem *ImageView* angezeigt. Der *MPMoviePlayerController* bietet eine vorgefertigte Methode, um ein Videobild zu einem bestimmten Zeitpunkt zu extrahieren.

Außerdem wird auf der rechten Bildhälfte ein *TextView* aktiviert, in dem die Annotation verfasst werden kann. Eine Annotation kann einen Titel besitzen und zusätzlich wird die aktuelle Zeit im Video rechts oben im Eck angezeigt. Sobald der Save Note Button gedrückt wird, wird die Annotation gespeichert und das Video läuft weiter. Die Annotation wird nicht persistent gespeichert und geht nach einem Neustart der App verloren. Um auf ältere Annotationen zugreifen zu können, sind die *ImageViews*, welche die einzelnen Frames enthalten, in einem *ScrollView* platziert. Dadurch kann man zu vorhergehenden Annotationen scrollen. Wird ein Touch auf einem dieser *ImageViews* ausgeführt, wird die dazugehörige Annotation geladen und auf der rechten Seite angezeigt. Zusätzlich wird im Video zu der Position gesprungen, bei der die Annotation erstellt wurde. Das Video wird in diesem Fall pausiert.

Der Prototyp wurde in dieser einfachen Version getestet und diente als erste Evaluierung

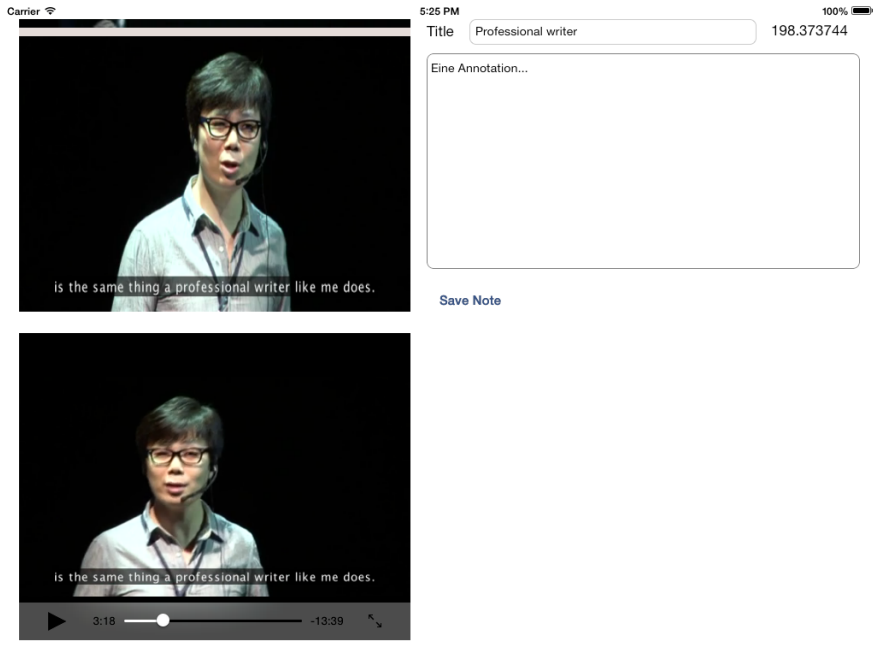

Abbildung 5.4: Erste Version des Prototyps

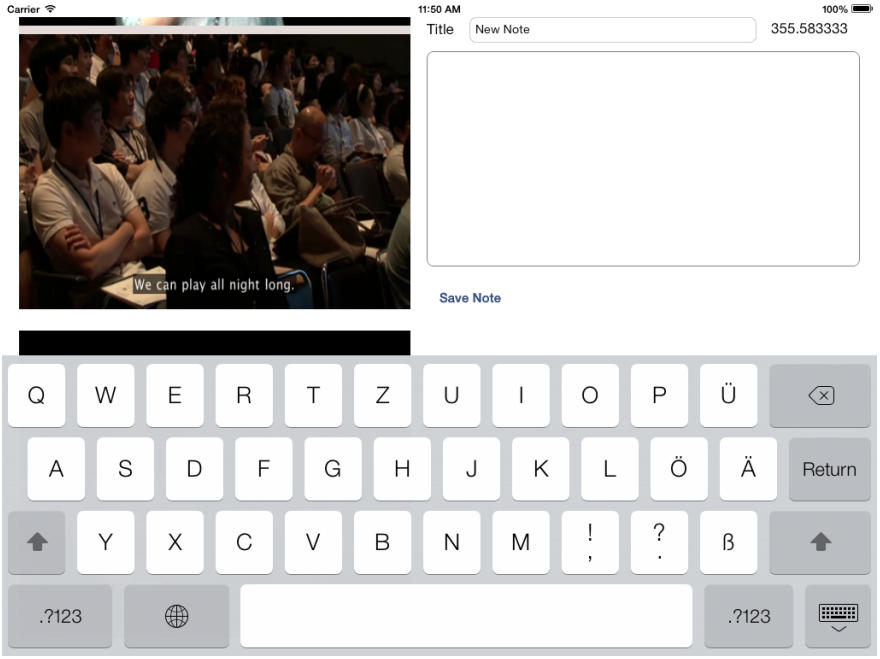

Abbildung 5.5: On-Sreen-Tastatur

des Konzepts. Im Allgemeinen machte der Prototyp einen sehr guten ersten Eindruck und war in der Lage das Potential einer solchen App zu demonstrieren. Allerdings wurden gleichzeitig auch die ersten Probleme sichtbar:

- Die Drag-Geste nach oben zum Erstellen einer neuen Annotation fühlt sich nicht besonders intuitiv an. Generell ist es gewöhnungsbedürftig, dass das Video links unten platziert ist.

- Die Anzeige der bisherigen Annotationen durch die dazugehörigen Frames ist nicht ideal, da man durch das Scrollen schnell den Überblick verliert. Außerdem ist auch bei der Übersicht selbst, kein Zusammenhang gegeben, erst wenn man ein Frame berührt wird die dazugehörige Annotation, sowie die Position im Video sichtbar.

- Die Annotation durch Text ist nicht besonders zielführend. Zum einen verdeckt die On-Screen Tastatur die Hälfte des verfügbaren Bildschirms (Abbildung 5.5). Zum anderen ist man, obwohl die Tastatur in dieser Größe angezeigt wird, mit einer On-Screen-Tastatur deutlich langsamer, als mit einer herkömmlichen Tastatur [96].

5.4 Weiterentwicklung und Funktionen des Prototyps

Nachdem durch die erste Version bereits ein guter Eindruck gewonnen werden konnte, wurde mit der Weiterentwicklung des Prototyps begonnen. Der Fokus lag dabei einerseits darauf, die offensichtlichen Probleme, die bei der ersten Version auftraten, zu lösen. Andererseits sollte der Prototyp soweit fortschreiten, dass er von einem breiteren Publikum getestet werden kann. Dadurch soll zusätzliches Feedback eingeholt werden, um die App weiter zu entwickeln. Dieser Abschnitt des Kapitels gibt einen Überblick, über den aktuellen Stand des Prototyps. Die implementierten Funktionen werden anhand des User-Interfaces besprochen.

Startansicht

Sobald der Prototyp gestartet wird, wird eine Übersicht über bisher erstellte Projekte angezeigt (Abbildung 5.6). Alle Videos die bisher annotiert wurden, werden in einem Gitter angezeigt. Dazu wird automatisch ein Bild aus dem Video als Titelbild für das Projekt ausgewählt. Es wird immer ein Videoframe aus der Mitte des jeweiligen Videos extrahiert. Würde man immer das erste Frame nehmen, hätte man oft das Problem, dass das Frame schwarz ist und deshalb keinen Aussagewert hat. Neben einem Bild aus dem Video, wird für jedes Projekt die Anzahl der vorhandenen Annotationen angezeigt.

Wenn man ein Video annotieren will, hat man zwei Möglichkeiten. Die erste ist, ein vorhandenes Projekt auszuwählen. Das geschieht durch einen Touch auf das jeweilige Titelbild in

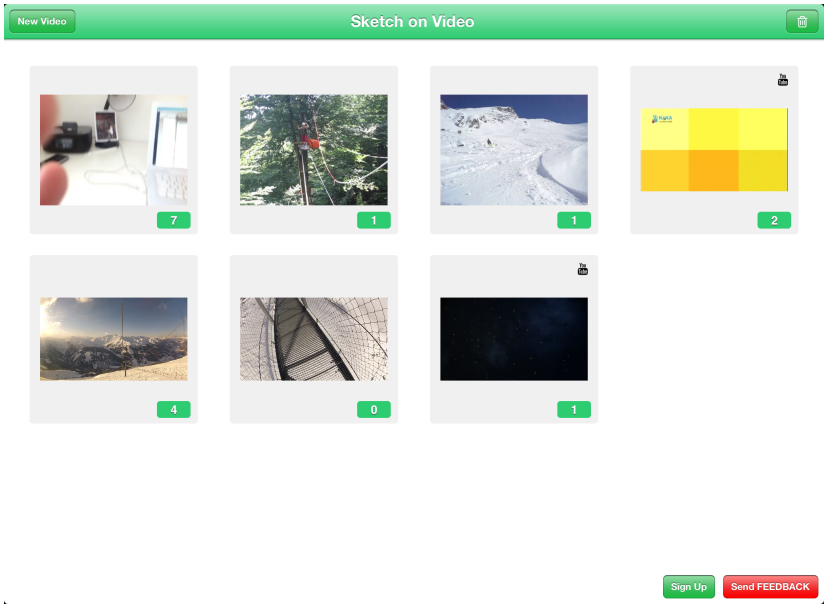

Abbildung 5.6: Startansicht des Prototyps

der Gitterübersicht. Dadurch gelangt man direkt zur Annotationsansicht. Die zweite Möglichkeit, ist das Erstellen eines neuen Projektes mit Hilfe des New-Video-Buttons. Hier kann man aus drei unterschiedlichen Optionen wählen. Es ist möglich ein bereits vorhandenes Video aus der Photo-Library zu wählen, ein Video aufzunehmen oder ein YouTube-Video zu verwenden. Will man ein YouTube Video annotieren, erscheint ein Eingabefeld, bei dem man die URL zum jeweiligen Video angeben kann. Sobald man eine dieser drei Optionen gewählt hat, gelangt man ebenfalls zur Annotationsansicht.

Annotationsansicht

Die Annotationsansicht ist der zentrale Teil des Prototyps und ermöglicht das Erstellen von Annotationen (Abbildung 5.7). Sie ist außerdem die Weiterentwicklung der ersten Version des Prototyps. Es wurde versucht die dort aufgetretenen Probleme zu lösen. Aus diesem Grund wurde das Layout geändert. Das Video wird nun nicht mehr links unten, sondern links oben angezeigt. Der Bildschirm ist in zwei Hauptbereiche aufgeteilt. Der obere Bereich, ist der Playbackbereich und der untere, der Annotationsbereich. Diese Trennung ist auch optisch anhand der unterschiedlichen Farben zu erkennen. Der Playbackbereich hat einen schwarzen Hintergrund, der Annotationsbereich einen weißen. Sobald man eine Drag-Geste vom Video in den Annotationsbereich ausführt, wird das aktuelle Videoframe auch unten angezeigt. Das funktioniert sehr ähnlich wie bereits in der ersten Version des Prototyps. Allerdings wirkt diese Interaktion durch die bloße Änderung der Richtung nun um einiges intuitiver. Ein Grund dafür

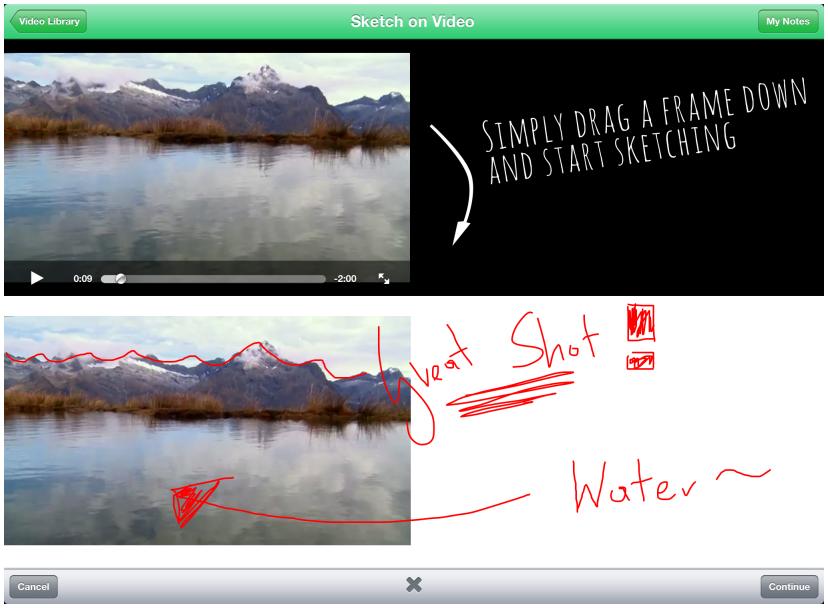

Abbildung 5.7: Annotationsansicht des Prototyps

könnte sein, dass man bei der Drag-Geste von unten nach oben, das Videobild mit der Hand verdeckt und somit etwas an Kontext verloren geht.

Sobald ein Videoframe in den Annotationsbereich gezogen wurde, wird das Video pausiert und gleichzeitig der Playback-Bereich deaktiviert. Nachdem bei der ersten Version des Prototyps festgestellt wurde, dass Texteingabe nicht ideal ist, wurde dies geändert. Die Annotation erfolgt nun durch freihändiges Zeichnen. Dazu kann entweder der Finger oder ein Stylus verwendet werden. Sobald der Touchscreen irgendwo im Annotationsbereich berührt wird, wird an dieser Position eine rote Linie gezeichnet. Rot wurde als Farbe deshalb gewählt, weil es sich bei den meisten Videos sehr gut vom Videobild abhebt. Man kann entweder direkt über das Videoframe zeichnen oder den zusätzlichen Platz auf der rechten Seite nutzen. Dadurch können beispielsweise Markierungen direkt im Video vorgenommen werden, aber auch zusätzlich kurze, handschriftliche Notizen im rechten Bereich verfasst werden. Der Annotationsbereich ist somit eine durchgängige Zeichenfläche. Am unteren Bildschirmrand wird, während eine Annotation erstellt wird, eine Toolbar angezeigt. Dort kann die Annotation abgebrochen oder die Zeichenfläche gelöscht werden. Durch die Berührung des Continue-Buttons wird die Annotation gespeichert.

Nachdem eine Annotation erstellt wurde, wird das Video im Playback-Bereich automatisch wieder fortgesetzt. Zusätzlich wird die soeben erstellte Annotation dem Videobild überlagert und innerhalb von 3 Sekunden kontinuierlich ausgeblendet. Aus diesem Grund hat der Playback-Bereich genau die gleiche Größe, wie der Annotationsbereich, obwohl das Video nur

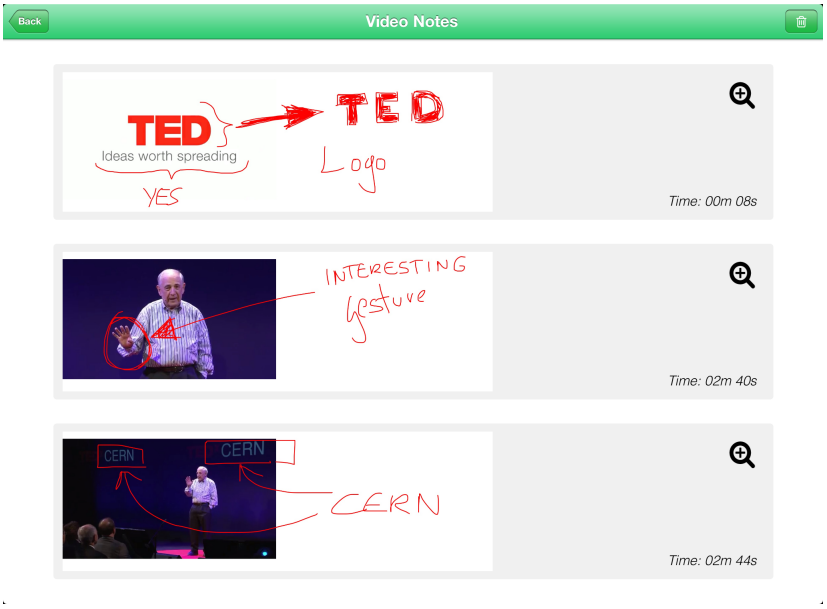

Abbildung 5.8: Video-Notes-Ansicht des Prototyps

die Hälfte des Platzes einnimmt. Generell werden im Playbackbereich bereits erstellte Annotationen automatisch angezeigt. Sobald zur aktuellen Position im Video eine Annotation gefunden wurde, wird diese überlagert und ebenfalls für 3 Sekunden eingeblendet. Wenn es mehrere Annotationen zum gleichen Zeitpunkt, oder sehr nahe beieinander gibt, werden diese übereinander angezeigt .

Video-Notes-Ansicht

Von der Annotationsansicht gelangt man über den *My-Notes*-Button zu der Video-Notes-Ansicht (Abbildung 5.8). Diese bietet einen Überblick über alle bisher erstellten Annotationen. Im Gegensatz zum anfänglichen Konzept und der ersten Version des Prototyps, wurde diese Funktion in eine eigene Ansicht ausgelagert. Die Annotationen werden in einer Liste angezeigt und sind nach dem zugehörigen Zeitpunkt im Video sortiert. Jeden Annotation wird als Bild des dazugehörigen Videoframes dargestellt. Zusätzlich wird die Zeit der Annotation angezeigt. Man kann durch die Liste der Annotationen scrollen und durch einen Touch auf eine der Annotationen gelangt man zurück zur Annotationsansicht. Dort wird dann genau zu jener Stelle im Video gesprungen, bei der die Annotation erstellt wurde. Die Annotation wird dabei im Playbackbereich dem Video überlagert. Die Video-Notes-Ansicht ist ein Inhaltsverzeichnis der erstellten Annotationen. Durch die Verbindung der Annotation mit dem jeweiligen Zeitpunkt im Video können die Annotationen für längere Videos zusätzlich auch als Bookmarks dienen. Dadurch wird das Wiederfinden von bestimmten Stellen in einem Video erleichtert. In der Video-Notes-Ansicht

können außerdem einzelne Annotationen gelöscht werden. Zusätzlich ist es durch das Lupen Icon möglich, jede Annotation einzeln als Bild im Großformat zu betrachten.

5.5 Technische Details und Herausforderungen

In diesem Abschnitt sind einige wichtige Punkte im Zusammenhang mit der technischen Umsetzung des Prototyps zusammengefasst. Das passiert allerdings, ohne genauer auf den Source-Code, oder einzelne Algorithmen einzugehen. Stattdessen soll dieser Abschnitt dazu dienen, die Herausforderungen, die bei der Entwicklung einer App zum Annotieren von Videos, auf der iOS Plattform auftreten, aufzuzeigen und mögliche Lösungen zu besprechen.

Um ein Video annotieren zu können, muss ein Video am iPad vorhanden sein. Das Laden eines Videos in einer iOS App kann auf unterschiedliche Arten geschehen. Grundsätzlich hat ein App keinen Zugriff auf das Dateisystem, sondern nur auf dessen eigene Sandbox [97]. Deshalb kann nicht einfach ein Video geladen werden, das irgendwo am iPad vorhanden ist. Einer der Wege, um Videos zu laden, ist über die Photo-Library. Die Photo-Library ist eine zentrale Bibliothek für Fotos und Videos, auf die Apps zugreifen dürfen, wenn es der User erlaubt. Videos können in die Photo-Library gelangen, wenn der User ein Video über die eingebaute Kamera aufnimmt oder beispielsweise ein Video von einem E-Mail abspeichert. Ein anderer Weg um Videos in einer App verfügbar zu machen, ist über File-Sharing [98]. Durch diese Methode können Dateien über iTunes mit der App ausgetauscht werden. Diese Methode ist im Prototyp allerdings im Moment nicht implementiert. Die dritte Variante, die neben dem Zugriff über die Photo-Library und dem direkten Aufnehmen eines Videos im Prototyp verfügbar ist, ist die Angabe einer YouTube URL. Dazu wird die *LBYouTubeView* [99] Klasse verwendet. Diese parst die YouTube HTML Seite und spielt das Video dann mit Hilfe der *MPMoviePlayerController* Klasse ab. Diese Funktion des Prototyps ist als experimentell einzustufen, weil YouTube ihre HTML Seite öfters ändert und außerdem nicht geklärt ist, ob diese Methode mit den Terms of Service (TOS) von YouTube vereinbar ist. Generell hat sich herausgestellt, dass der Zugriff auf Videos von verschiedenen Quellen am iPad nicht ganz einfach ist. Eine Alternative, die von vielen Apps in diesem Zusammenhang genutzt wird, ist die Einbindung eines Cloud-Datenspeicherdienstes wie Dropbox oder Google Drive. Diese bieten APIs, mit denen der Zugriff auf den Cloud Speicher über die eigene App möglich ist.

Sobald ein Video geladen wurde, kann es abgespielt werden. Dazu wird, wie bereits in der ersten Version des Prototyps, die *MPMoviePlayerController* Klasse verwendet. iOS unterstützt die Wiedergabe der Videoformate *.mp4*, *.mov* und *.m4v* [100]. Der *MPMoviePlayerController* kann Videos in einem *View* mit beliebiger Größe oder im Vollbildmodus anzeigen. Im Prototyp nimmt der MoviePlayer ungefähr ein Viertel des Bildschirms ein. Der Vollbildmodus kann zwar aktiviert werden, allerdings kann das Video in diesem Modus nicht annotiert werden und bereits vorhandene Annotationen werden auch nicht angezeigt. Neben grundlegenden Playback

Funktionen, bietet die *MPMoviePlayerController* Klasse auch eine Funktion zum Extrahieren von einzelnen Frames eines Videos. Dabei kann gewählt werden, ob das näheste Keyframe, oder das Frame exakt zum angegeben Zeitpunkt verwendet wird. Außerdem ist es mit dem *MPMoviePlayerController* auch möglich Videos zu streamen. Dieses Feature kommt bei der YouTube-Funktion zum Einsatz. Die *MPMoviePlayerController*-Klasse ist eine umfangreiche Klasse, die bereits viele Probleme, die beim Abspielen von Videos auftreten, löst. Für den Prototyp waren die Funktionen der Klasse ausreichend. Falls mehr Kontrolle über die Wiedergabe von Videos benötigt wird, bietet das iOS-SDK zusätzlich das *AV Foundation* Framework [101].

Um das Annotieren zu ermöglichen, muss ein Touch in eine Linie umgewandelt werden. Abschnitt 1.4.2 beschreibt bereits, welche Schritte dafür unter iOS nötig sind. Bei der Umsetzung war die WWDC 2012 Session 233 hilfreich. Dort wird gezeigt, wie man mit durch die Verwendung von Bézierkurven, Linien glätten kann. Viele der im Zuge der Recherche getesteten Apps, glätten bei der Zeichenfunktion die Linien nicht. Das führt zu eckigen Strichen und schadet der Usability, weil die Linien anders aussehen, als sie eigentlich vom User gezeichnet wurden. Aus diesem Grund wurde beim Prototyp Wert darauf gelegt, die Zeichenfunktion gut umzusetzen. Ein weiteres Problem, dass dabei aufgetreten ist, war schlechte Performance. Zum Zeichnen wurde das *Core Graphics* Framework verwendet, welches eine auf C basierende API für diverse Zeichenfunktionen bietet. Das Problem trat hauptsächlich beim iPad 3 auf und ist auf das Retina-Display und dessen höhere Auflösung zurückzuführen. Das Problem konnte dadurch gelöst werden, dass der Code zum Zeichnen überarbeitet wurde. Mit Hilfe der *setNeedsDisplayInRect* Methode kann angegeben werden, welcher Bereich eines Views neu gezeichnet werden soll. Durch diese Einschränkung konnte genug an Performance gewonnen werden, um das Zeichnen ausreichend schnell zu ermöglichen. Um die Performance noch weiter zu steigern, wäre die Verwendung von *OpenGL* eine mögliche Alternative. Der Prototyp kann am iPad nur im Querformat verwendet werden. Einer der Gründe dafür war, dass der *View*, der die Zeichenfunktion implementiert, dadurch eine fixe Größe hat. Ansonsten würde beim Rotieren der Zeichenbereich teilweise abgeschnitten und man müsste eine Zoomfunktion oder eine Möglichkeit zum Scrollen einbauen.

Eine weitere Funktion, die bei der ersten Version des Prototyps noch nicht vorhanden war, ist das persistente Speichern von Annotationen, sowie Projekten. Dafür wurde das Core Data Framework [102] verwendet, das ein Teil des iOS-SDKs ist. *Core Data* ist ein Framework zum Verwalten von Objekten. Damit können Objekte direkt gespeichert und geladen werden, ohne diese Objekte vorher in ein spezielles Dateiformat umwandeln zu müssen. Der Prototyp verwendet als Backend für *Core Data* eine SQLite Datenbank. Das Datenmodell kann grafisch mit Hilfe von XCode definiert werden und besteht beim Prototyp aus zwei Entities. Das sind *VideoProject* und *VideoNote*. Die V*ideoProject* Entity steht mit der *VideoNote* Entity in einer one-to-many Beziehung. Das heißt, einem Video können beliebig viele Annotation zugeordnet werden. Mit Hilfe von XCode können auch solche Beziehungen grafisch definiert werden. Die

Annotationen selbst werden im Moment einfach als Bild des kompletten Zeichenviews im *jpg* Format gespeichert. Dazu wird die *UIImage* Klasse verwendet.

Eine weitere Klasse, die beim Prototyp verwendet wurde, ist der *UICollectionView*. Mit einem *UICollectionView* können Daten in einem flexiblen und frei definierbaren Layout angezeigt werden. Ein *UICollectionView* wird bei der Startansicht verwendet, um die einzelnen Projekte anzuzeigen. Jedes Projekt ist eine *UICollectionViewCell* und die einzelnen Zellen werden in einem Gitter dargestellt. Der *UICollectionView* kümmert sich um die Anordnung der einzelnen Zellen und zusätzlich um die Darstellung, wenn mehr Zellen vorhanden sind, als auf einen Bildschirm passen. Auch bei der Video Notes Ansicht wird ein *UICollectionView* verwendet, um die einzelnen Annotationen in einer Liste anzuzeigen. Die *UICollectionView* Klasse bietet eine sehr einfache Möglichkeit, Daten darzustellen, ist aber gleichzeitig sehr flexibel und kann dadurch die Daten auf einer Vielzahl von verschiedenen Arten präsentieren.

Um den Prototyp auf verschiedenen Geräten zu testen, wurde TestFlight [103] verwendet. TestFlight ist ein gratis Service, das es erlaubt, Testversionen einer App an eine Gruppe von ausgewählten Testern zu verteilen. Zusätzlich stehen Funktionen zur Auswertung von Abstürzen, Userverhalten und Feedback zur Verfügung. TestFlight bietet ein SDK für iOS an, welches in die App eingebunden werden kann, damit diese Daten gesammelt werden.

5.6 Veröffentlichung im Apple App Store

Um zusätzliches Feedback einzuholen, wurde der Prototyp auch im App Store veröffentlicht. Zur Veröffentlichung werden diverse Informationen und Artefakte bezüglich der App benötigt. Als Name wurde *Sketch on Video* gewählt. Neben einer Beschreibung der App, werden auch Icons und Screenshots benötigt, die später im App Store aufscheinen. Der Prototyp wurde als gratis Version in den App Store gestellt. Zusätzlich zum App-Namen und der Beschreibung können auch noch Keywords angegeben werden, die für die Suche im App Store verwendet werden. Um durch die App Feedback zu ermöglichen, wurde ein Button eingebaut, durch den sich ein E-Mail Dialog öffnet, in dem man seine Meinung zu *Sketch on Video* verfassen kann. Die Veröffentlichung einer App wird über iTunes Connect abgewickelt. Sobald alle Daten eingetragen wurden, wird die App zum Review freigegeben. Jede App wird durch Apple einem manuellen Review unterzogen, das einige Tage dauern kann. *Sketch on Video* wurde bei der ersten Einreichung abgelehnt. Der Grund dafür war, dass keine Apps im App Store veröffentlich werden, die Beta, Demo, Test oder ähnliche Versionen sind. Nachdem der Prototyp aber grundsätzlich schon eine funktionsfähige App darstellt und die Hauptfunktionen gegeben sind, wurde beschlossen, alle Anmerkungen, die darauf hinweisen, dass es sich um einen Prototypen handelt, zu entfernen und die App noch einmal zum Review einzureichen. Das zweite Mal klappte es und die App wurde im App Store freigeschaltet. Beim Einreichen einer App muss auch eine Homepage angegeben werden. Zu diesem Zweck wurde eine Homepage erstellt, die alle nöti-

gen Informationen zu Skech on Video enthält [104]. Zur Hilfe bei der Erstellung kam HTML Kickstart [105] zum Einsatz. HTML Kickstart bietet einige bereits fertige HTM5, CSS und Javascript Bausteine, die das Erstellen einer Homepage beschleunigen. Besonders, dass diese Bausteine responsive sind, das heißt, sich auf mobilen Geräten mit verschiedenen Displaygrößen automatisch anpassen, war in diesem Fall hilfreich.

Zusammenfassung

Im Zuge der Recherche wurde deutlich, dass kaum Apps zur Annotation von Videos für Tablets existieren. Nur im Bereich der Sportanalyse gibt es einige Apps, die bereits eine solche Funktionalität bieten. Aber auch dort ist es meistens so, dass man lediglich über das Videobild zeichnen kann und die einzelnen Annotationen keinen zeitlichen Zusammenhang mit dem Video besitzen. In diesem Kapitel wurde die Entwicklung eines Prototyps zur Videoannotation für das iPad zusammengefasst. Durch die Umsetzung der ersten Version des Prototyps, war es möglich, die Kernfunktionalität zu testen. Dabei wurden allerdings auch die ersten Probleme sichtbar. Beispielsweise, dass das textuelle Annotieren über die On-Screen Tastatur mühsam ist, oder die Platzierung des Videos und die Richtung, in der die Geste zum Annotieren ausgeführt wird, wichtig für die User-Experience sind. Auf der anderen Seite zeigte die erste Version auch das Potential einer solchen Anwendung.

In weiterer Folge wurden Lösungen für die bei der ersten Version aufgetretenen Probleme, erarbeitet und im Zuge der Weiterentwicklung des Prototyps umgesetzt. Außerdem wurde der Prototyp soweit ergänzt, dass er die essentiellen Funktionen enthält und somit von einem breiteren Publikum getestet werden kann, um zusätzliches Feedback zu erhalten. Zu diesen wichtigen Funktionen zählen das Erstellen von Projekten, die Verwendung von Videos aus verschiedenen Quellen, sowie die Übersicht und persistente Speicherung von erstellten Annotationen.

Weiters wurden in diesem Kapitel einige Herausforderungen, sowie technische Details, die bei der Umsetzung für die iOS-Plattform aufgetreten sind, behandelt. Durch die diverse Einschränkungen ist es teilweise nicht einfach, auf Videos zuzugreifen. Auch im Bereich der Performance traten teilweise Probleme auf, die allerdings gelöst werden konnten. Insgesamt haben die positiven Aspekte des iOS-SDKs überwogen. Das SDK bietet eine Vielzahl an Frameworks und Klassen, die die Entwicklung des Prototyps beschleunigten. Abschließend wurden noch kurz die Erfahrungen der Veröffentlichung des Prototyps im App Store geschildert. Der Prototyp ist im Moment unter dem Namen *Sketch on Video* im App Store frei verfügbar.

KAPITEL 6

Feedback und mögliche Weiterentwicklung

Nachdem bisher die theoretischen Grundlagen und die Entwicklung des Prototyps näher betrachtet wurden, wird in diesem Kapitel die mögliche Weiterentwicklung der App aufgrund von Methoden des explorativen Designs beschrieben. Zunächst wird kurz auf die Begriffe User-Centered-Design sowie Co-Design eingegangen und deren Zusammenhang erläutert. Danach werden die Ergebnisse des Feedbacks zum Prototyp präsentiert. Das Feedback wurde zum einen dadurch erlangt, dass der Prototyp im Apple App Store veröffentlicht und mit einem Hinweis zum Einbringen von Feedback versehen wurde. Zum anderen wurden Gespräche mit sieben Personen geführt, die einen unmittelbaren Bezug zur (kollaborativen) Annotation oder Klassifikation von Video haben. Dabei wurden die möglichen Anwendungsgebiete einer solchen Tablet-App besprochen und auf die Usability, sowie Funktionalität des Prototyps näher eingegangen. Basierend auf dem Feedback, sowie den Erkenntnissen aus der theoretischen Auseinandersetzung mit dem Themenkomplex der Arbeit, werden zum Abschluss die Eckpunkte einer möglichen iterativen Weiterentwicklung der App präsentiert.

6.1 Vom User-Centered-Design zum Co-Design

Ein Schlagwort das im Zusammenhang mit der Software sowie App Entwicklung immer wieder auftaucht ist User-Centered.Design (UCD). UCD ist ein breit gefächerter Begriff, der im Wesentlichen einen Designprozess beschreibt, bei dem das Design während der Entwicklung durch den zukünftigen User beeinflusst wird [106]. Auf welche Weise der User auf den Designprozess Einfluss nimmt kann unterschiedlich sein. So können die User mittels Interview befragt werden, Usability-Tests durchgeführt werden oder der User am Arbeitsplatz beobachtet werden, um die Zusammenhänge und Abläufe der jeweiligen Aufgabe besser zu verstehen. In welchem Stadium des Designprozesses welche Methode verwendet wird ist nicht genau festgelegt. Zum Beispiel macht es Sinn die User vor Beginn der Entwicklungsphase zu interviewen, um die Bedürfnisse und Erwartungen herauszufinden. Auf der anderen Seite kann eine Befragung auch nach der Entwicklung eines Prototyps oder des fertigen Produkts sinnvoll sein, um auf Probleme aufmerksam zu werden und mögliche Lösungen zu validieren. Generell handelt es sich beim UCD um einen iterativen und dynamischen Prozess, der durch das von den Usern gegebene Feedback ermöglicht wird. Wer genau diese User sind ist nicht immer eindeutig. Einerseits gibt es User die direkt mit der entwickelten Software arbeiten und andererseits gibt es auch oft jemanden, dem diese User untergeordnet sind. Diese Person kann genauso als User gesehen werden. Allerdings werden sich seine Anforderungen an das System von denen der direkten User zumindest teilweise unterscheiden. Aus diesem Grund ist es wichtig sich darüber Gedanken zu machen, wer und in welchem Ausmaß die Software später direkt oder indirekt benutzen wird. Im Allgemeinen kann UCD aus unterschiedlichen Methoden, welche in verschiedenen Phasen des Designprozesses eingesetzt werden, gesehen werden. Diese haben das Ziel ein breiteres Verständnis für den User zu gewinnen, mit dessen Hilfe es möglich ist, das System zu verbessern und ein für den User zufriedenstellendes Ergebnis zu entwickeln.

UCD kann in verschiedenen Bereichen angewandt werden und hat in den letzten Jahren im Bereich der Softwareentwicklung eine weite Verbreitung gefunden [107]. Ein Grund dafür könnte das Erwachsenwerden der Computertechnologie sein. Dort wo neue technische Entwicklungen nicht mehr die treibende Kraft sind, wird mehr Wert auf die Ansprüche der User und die User-Experience gelegt. Klassischerweise wird UCD aus einer Expertensicht heraus betrieben [108]. Das heißt beispielsweise, dass ein Designer bestimmte Testfälle für einen Usability-Test zusammenstellt und aus deren Ergebnissen in weiterer Folge Schlüsse zieht. Der User kommt also nur als Tester, auf eine eher passive Weise zum Einsatz. Diesem Ansatz steht das Co-Design (auch partizipatives Design oder kooperatives Design) gegenüber. Co-Design hat skandinavische Wurzeln und wird dort bereits seit einigen Jahrzehnten im Zusammenhang mit der Softwareentwicklung praktiziert [109]. Dabei geht es darum, den Designer und User auf die gleiche Ebene zu stellen und eine Zusammenarbeit zu ermöglichen. User sind Experten ihrer eigenen Erfahrungen [110] und können dadurch eine wichtige Rolle in den verschiedenen

Phasen des Designprozesses spielen. Abbildung 6.1 stellt die klassische Variante des UCD dem Co-Design gegenüber. Wichtig für funktionierendes Co-Design ist es, dass Designer und User eine "gemeinsame Sprache" sprechen. Dazu müssen Werkzeuge und Artefakte entwickelt werden, welche zugleich vom Designer als auch vom User verstanden und genutzt werden können. Dadurch kann der User in weiterer Folge sowohl bei der Ideengenerierung, als auch bei der Konzeptentwicklung mitwirken.

Co-Design geht davon aus, dass auch der User in der Lage ist kreativen Input zu liefern. Das bedeutet, dass das übliche hierarchische Denkmuster, dass nur ganz spezifische Experten auf dem jeweiligen Gebiet dazu fähig sind, verworfen wird. Sanders E. & Stappers P. [108] führen dieses Umdenken unter anderem auf das Internet zurück. Dort gibt es mittlerweile unzählige Beispiele, bei denen sich Personen, die traditionellerweise kein Mitspracherecht an der Entwicklung hatten, Gehör verschaffen konnten. Durch die Gleichstellung, sowie Annäherung des Users an Designer und Entwickler entstehen neue Chancen und Möglichkeiten im Designprozess.

6.2 Co-Design mit Hilfe des entwickelten Prototyps

Eine Möglichkeit um eine Konversation zwischen User, Designer und Entwickler zu ermöglichen ist ein Prototyp. Dieser Weg wurde im Zuge der vorliegenden Arbeit gewählt und die zweite Version des entwickelten Prototyps dient als Basis für das Co-Design. Co-Design kann in verschiedenen Phasen betrieben werden. Gute Ergebnisse werden oft schon am Beginn, bei der Ideenfindung und Erarbeitung von verschiedenen Szenarien erzielt. Allerdings kann Co-Design genauso während des Designprozesses in wichtigen Entscheidungsphasen eingesetzt werden.

Im konkreten Fall ist das Grundkonzept der Tablet-App zum Annotieren von Videos schon zu Beginn festgestanden. Allerdings wurde nach der Entwicklung der ersten zwei Prototypen

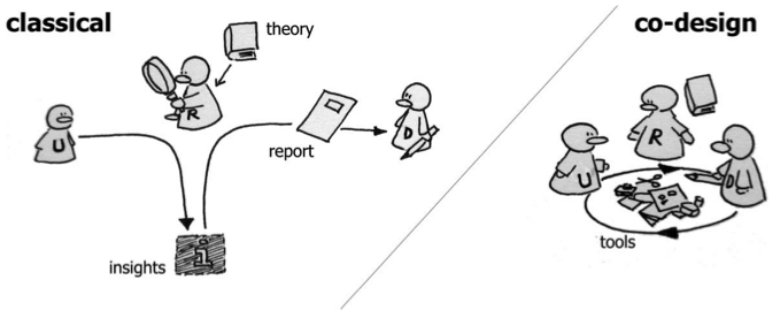

Abbildung 6.1: Vergleich vom klassischen UCD zum Co-Design [108]

und der Recherche deutlich, dass die Möglichkeiten einer Verwendung einer solchen App sehr unterschiedlich sind. Aus diesem Grund war eine der zentralen Fragen, in Hinsicht auf die zukünftige Weiterentwicklung, die Auswahl eines Anwendungsbereiches. Zusätzlich sollte Feedback zu den umgesetzten Interaktionen, dem User-Interface-Design und der allgemeinen Funktionalität des Prototyps eingeholt werden. Um Feedback und eine Konversation mit potentiellen Usern zu ermöglichen wurden zwei unterschiedliche Ansätze verwendet.

Feedback-Varianten

Zum einen wurden mit sieben Personen aus dem universitären Umfeld individuelle Gespräche geführt. Dabei wurden Personen ausgewählt, die einen unmittelbaren Bezug zur (kollaborativen) Annotation oder Klassifikation von Videos haben. Außerdem wurde darauf geachtet, dass ihre Tätigkeitsbereiche möglichst unterschiedlich sind. Zu den Bereichen in denen die befragten Personen Erfahrung haben zählen unter anderen die HCI, ubiquitous Computing, Soziologie, Psychologie, Medienanalyse und die Verwaltung von Informationssystemen. Bei den Gesprächen wurde zu Beginn der Prototyp kurz präsentiert und danach die Möglichkeit geboten, ihn selbst zu verwenden und zu testen. Einigen der Personen stand der Prototyp bereits vor dem Gespräch zur Verfügung. Während die Personen den Prototyp ausprobierten, wurde gleichzeitig ein Gespräch geführt. Im Gegensatz zu einem klassisches Interview sind dafür im Vorfeld keine fixen Fragen definiert worden, sondern nur ein paar Anhaltspunkte festgelegt worden, die je nach dem Verlauf des Gesprächs angesprochen wurden.

Einerseits wurde auf die Erfahrung der Gesprächspartner mit Videos und der Videoannotation eingegangen. Dabei wurde darüber gesprochen in welchen Bereichen und zu welchem Zweck sie bei ihrer Arbeit Videos einsetzen. Außerdem wurde gefragt, ob sie diese Videos bereits auf irgendeine Art und Weise annotieren und welche Anwendungen sie gegebenenfalls dazu verwenden. Andererseits wurde das Gespräch auch auf den Prototyp selbst gelenkt. Dabei wurde nach dem ersten Eindruck gefragt, sowie nach denkbaren Verwendungszwecken im eigenen Arbeitsumfeld und allgemein.Je nach Verlauf des Gespräches wurde teilweise näher auf den Prototyp eingegangen und diverse Interaktionen oder das User-Interface-Design besprochen. Bei anderen Gespräche sind wiederum mögliche Anwendungsszenarien genauer thematisiert worden.

Neben diesen Gesprächen gab es noch einen zweiten Weg auf dem Feedback eingeholt wurde. Wie bereits im Abschnitt 4.7 beschrieben, wurde der Prototyp mit der Möglichkeit Feedback per E-Mail zu senden im App Store veröffentlicht. Einige User nutzten diese Möglichkeit und schickten auf diesem Weg ihre Anregungen und Kritik. Insgesamt konnte dadurch innerhalb von drei Monaten Kontakt zu 24 Personen aufgenommen werden, wodurch sich aufschlussreiche E-Mail-Konversationen ergaben. Die Verfügbarkeit von App-Stores bietet eine neue Möglichkeit Nutzungsdaten, sowie Feedback von einer großen Anzahl an Usern weltweit

zu sammeln [111]. Durch diesen neuen Vertriebskanal ist es möglich interessierte Personen zu erreichen, die vorher nicht oder mit mehr Aufwand erreicht werden konnten. Für die User gibt es verschiedene Möglichkeiten Apps im App Store zu entdecken. Neben den diversen Kategorien und Top-Listen wird oft die integrierte Suchfunktion verwendet. Damit die App von interessierten Usern gefunden wird, ist es wichtig, bei der Einreichung der App aussagekräftige Keywords anzugeben. Diese Keywords werden dann später bei der Suche miteinbezogen. Neben der in dem Fall vom Prototyp gewählten Methode des Feedbacks per E-Mail gibt es noch anderen Möglichkeiten Nutzungsdaten zu erfassen. Apple bietet mittels iTunes Connect [112] grundlegende Informationen über die Anzahl und geographische Herkunft der Downloads einer App an. Zusätzlich gibt es diverse Frameworks, die in eine App integriert werden können und mit Hilfe derer genauere Daten erhoben werden können. Beispiele dafür sind Google Analytics [113] oder HockeyApp [114]. Solche Frameworks ermöglichen einen genaueren Einblick in das Userverhalten und es können zum Beispiel die Sitzungsdauer oder die Aufrufe von bestimmten Bildschirmen innerhalb der App gemessen werden.

Durch die Gespräche, sowie die E-Mail Konversationen, konnte essentielles Feedback gewonnen werden, welches in weiterer Folge genauer besprochen wird.

Mögliche Einsatzgebiete und Anwendungsszenarien

Ein Ziel war es, durch das erhaltene Feedback Anwendungsgebiete für die App zum Annotieren von Video festzulegen. Durch die zahlreichen Ideen, sowie Erfahrungen, konnten eine Vielzahl von Einsatzgebieten erarbeitet werden. Diese werden nachfolgend aufgezählt und kurz beschrieben.

- **Bildungsbereich & E-Learning** - Eines der am häufigsten genannten Einsatzgebiete, vor allem bei dem Feedback der User durch den App Store, ist der Bildungsbereich.

  "Quite like this app and can see its value in teaching and learning." - Mike

  Dort gibt es durchaus unterschiedliche Szenarien. Schüler oder Studenten könnten die App zum Aufarbeiten von aufgezeichneten Unterrichtseinheiten benutzen. Viele Inhalte werden mittlerweile bereits in Videoform angeboten, wodurch das Ansehen und Erarbeiten des Lernmaterials im eigenen Tempo ermöglicht wird. Allerdings gibt es oft keine einfache Möglichkeit sich zu den Videos Notizen zu machen. Dabei wurde während eines Gespräches die Analogie zu gedruckten Texten angesprochen, bei denen wichtige Stellen oft mit Hilfe eines Markers markiert werden. Die App bietet eine ähnliche Funktion für Videos, um wichtige und interessante Stellen in einem Video markieren zu können.

  Ein anderes Szenario ist der Einsatz als Tool zum Erstellen von Feedback. Dabei können Lehrer die von Schülern erstellten Videos ansehen und direkt mit Anmerkungen versehen. In einem konkreten Fall müssen die Schüler Videos von chemischen Experimenten

aufzeichnen und die App könnte verwendet werden, um den Verlauf des Experiments zusätzlich zu annotieren. Auch denkbar ist, dass Schüler Videos untereinander austauschen und sich gegenseitig Feedback geben. Eine weitere Möglichkeit für Lehrer ist die Verwendung zum Erstellen von kurzen Videoanleitungen für ihre Schüler. Durch die App hätten sie die Möglichkeit diese Videos mit Notizen und Zeichnungen zu ergänzen und wären somit in der Lage essentiellen Bestandteile besser herauszuarbeiten.

Positives Feedback zum Prototyp gab es über den App Store von Lehrern sowie von Schülern, die sich durchaus vorstellen können, eine solche App in ihren Lehr- bzw. Lernalltag zu integrieren.

- **Unterhaltung** - Bei den geführten Gesprächen fiel auf, dass beim Testen oft Personen, die in Videos vorkamen, nachgezeichnet oder zum Beispiel durch eine Brille oder Ähnliches humoristisch ergänzt wurden. Die einfache Möglichkeit in ein Video zeichnen zu können, könnte durchaus auch im Unterhaltungsbereich interessant sein. Mit Hilfe der integrierten Kameras in Tablets, sowie Smartphones, werden häufig Heimvideos produziert. Ein im Zuge des Feedbacks erwähntes Szenario wäre zum Beispiel, die App so zu erweitern, dass man leicht Sprechblasen, Pfeile, Herzen und ähnliches in Videos einfügen kann. Dadurch kann auf eine einfache Weise eine erweiterte Version des zuvor aufgenommenen Videos erschaffen werden.

  Ein anderer Anwendungsfall in diesem Bereich wäre das Annotieren von Urlaubsvideos. So könnte man die Aufnahmen, die während einer Reise entstanden sind, mit zusätzlichen Notizen oder Geschichten ergänzen. Durch die App wäre es dadurch möglich ein Reisetagebuch in Videoform zu realisieren.

- **Sportanalyse** - In Abschnitt 4.1 wurden bereits einige existierende Apps vorgestellt, die das Annotieren von Videos zur Sportanalyse ermöglichen. Deshalb war es wenig verwunderlich, dass dieses Einsatzgebiet auch bei den durchgeführten Gesprächen aufkam. Im Sport ist die Videoanalyse beispielsweise ein häufig verwendetes Mittel zur Verbesserung der Technik von Athleten. Die App könnte Trainer, sowie Athleten, dabei unterstützen Aufnahmen zu analysieren und kommentieren. Denkbar ist natürlich auch die spätere Analyse von aufgenommen Spielen oder Wettkämpfen. So könnte mit Hilfe der App ein aufgezeichnetes Fußballspiel auf unterschiedliche Kriterien analysiert werden. Durch die im Prototyp enthaltene Übersicht der Annotationen bekommt man gleichzeitig eine Liste mit den wichtigsten Momenten des Spiels.

- **Coaching** - Nicht nur im Sport wird die Videoanalyse verwendet um Leistungen zu verbessern. Es gibt auch andere Gebiete, bei denen diese Methode eingesetzt wird und die App somit hilfreich sein könnte. In den Gesprächen wurden in diesem Zusammenhang zwei Szenarien genannt. Zum einen die Analyse von Präsentationen und zum anderen die

Analyse von Verkaufsgesprächen. Ein geeignetes Mittel um die Präsentationstechnik zu verbessern ist das Aufnehmen der Präsentation auf Video. Ein Coach könnte sich dann diese Aufnahme ansehen und die Präsentation mit Hilfe der App annotieren. Dabei kann zum Beispiel die Körperhaltung des Vortragenden analysiert werden oder die verwendete Gestik. Durch die Verwendung der App wäre eine solche Analyse im Nachhinein ohne Probleme möglich und es könnten auf diesem Wege Verbesserungsvorschläge festgehalten werden.

Das zweite Szenario ist die Analyse von Verkaufsgesprächen. Verkäufer müssen oft intensive Schulungen absolvieren, bei denen sie auf Verkaufsgespräche vorbereitet werden. Dabei werden die Gespräche unzählige Male durchgespielt und trainiert, um auf verschiedene Situationen und Gesprächsverläufe einzugehen. Die App könnte wiederum dabei helfen die Probegespräche zu analysieren und ein besseres Feedback zu geben.

- **Forschung und Wissenschaft** - Im wissenschaftlichen Bereich gibt es grundsätzlich diverse Bereiche in denen die Annotation von Videos wichtig ist. (vgl. Abschnitt 2.2.1) Auch im Zuge der geführten Gespräche wurden Anwendungsfälle angesprochen. Zum Beispiel wird in der Verhaltensforschung oder Soziologie oft eine qualitative Videoanalyse durchgeführt. Für solche Anwendungen reicht die Funktionalität des Prototyps allerdings nur bedingt aus. Zur qualitativen Videoanalyse werden meist relativ komplexe Programme verwendet, die mit großen Videodatenbanken im Hintergrund arbeiten. Als Beispiel für solche Anwendungen wurden Atlas [115] und Maxqda [116] genannt. Damit ist es möglich Videomaterial zu organisieren, codieren, transkribieren und auszuwerten. Ähnlich wie bei den in Abschnitt 2.2.1 vorgestellten Anwendungen ist für die Verwendung dieser Programme eine gewisse Einarbeitungszeit nötig.

- **Videoproduktion** - Bei der Videoproduktion gibt es eine Vielzahl von unterschiedlichen Schritten. Ein Schritt ist das genaue Review von bereits geschnittenem Videomaterial. Dabei wird zum Beispiel darauf geachtet, dass keine falschen Frames vorhanden sind oder, dass nirgendwo ein Mikrofon im Bild zu sehen ist. Einer der Gesprächspartner hat Erfahrungen bei der TV-Produktion und kann sich vorstellen, dass genau für diesen Review-Prozess die App ein ideales Tool darstellt. Das schnelle Markieren von Stellen im Video die noch überarbeitet werden müssen wäre möglich. Einen ähnlichen Anwendungsfall schildert auch ein User, der den Prototyp mittels App Store gefunden hat. Er produziert Marketingvideos und kann sich vorstellen das App zu verwenden, um Rohfassungen anzusehen und etwaige Änderungen zu kommentieren. Anhand dieser Notizen kann dann einer seiner Mitarbeiter das Video verbessern.

- **Weitere Szenarien** - Neben den bereits erwähnten Einsatzgebieten wurden noch ein paar weitere Szenarien genannt, für die die App verwendet werden könnte.

Usability-Tests werden verwendet, um Software und Hardware durch zukünftige User zu testen. Diese Tests werden oft auf Video aufgezeichnet, um später Details genauer analysieren zu können. Dabei kann die App hilfreich sein, weil man gleichzeitig Notizen zu auftretenden Erkenntnissen festhalten kann.

Auch im Bereich der Qualitätssicherung werden Prozesse teilweise auf Video aufgenommen. Mit der App könnten die Prozesse genauer beschrieben werden und Vorgänge die im Video zu sehen sind, referenziert werden.

Ein User wies darauf hin, dass er zum Training von gehörlosen Personen Videos verwendet und die App ideal für seine Schüler wäre, um sich zu den Videos persönliche Notizen zu machen.

Positives Feedback

Der Prototyp wurde sowohl bei den Gesprächen, als auch bei den Usern, die ihn mittels App Store entdeckten, sehr positiv aufgenommen.

"This is a wonderful app. i love it." - Oguz

"This app is great. I've been looking for something like this." - Jonathan

Ï really like this app, it is very unique" - Zain

Dabei wurde vor allem die einfache und intuitive Bedienbarkeit der App gelobt. Das zentrale Konzept zum Hinzufügen der Annotationen mittels einer Drag-Geste vom Video in den Annotationsbereich funktioniert sehr gut und wird ohne Probleme verstanden. Einige User waren überrascht, wie einfach es mit Hilfe des Prototyps ist in bzw. über ein Video zu zeichnen. Auch die Verknüpfung der Annotationen mit der jeweiligen zugehörigen Zeit im Video wurde mehrmals lobend und als besonders wichtig hervorgehoben. Dadurch ergibt sich kombiniert mit der Video-Notes-Ansicht ein guter Überblick über die Annotationen und das Video selbst.

Kritisches Feedback und Verbesserungswünsche

Neben den allgemein sehr positiven Rückmeldungen, gab es natürlich auch kritisches Feedback, Verbesserungswünsche und diverse Erweiterungsvorschläge. Die mit Abstand am häufigste angesprochene fehlende Funktion, vor allem bei dem erhaltenen Feedback aus dem App Store, war das Exportieren von annotierte Videos. Viele User wären gerne dazu in der Lage, die von ihnen mit dem App erstellten Videos zu exportieren und somit mit anderen Personen zu teilen. Mit der aktuellen Version des Prototyps ist das nicht möglich. Der Wunsch das annotierte Video zu exportieren und mit anderen Personen zu teilen ist im Hinblick auf das kollaborative Arbeiten nachvollziehbar, und es finden sich bei den bereits aufgezählten Anwendungsszenarien

viele Fälle, bei denen das Video jeweils von mehreren Personen entweder nur betrachtet oder auch annotiert wird.

Neben der fehlenden Exportfunktion war der zweihäufigste Kritikpunkt die sehr eingeschränkten Annotationsmöglichkeiten der App. Im Prototyp können Annotationen nur mittels Toucheingabe in Form von roten Linien erstellt werden. Für viele User und Anwendungsfälle ist das nicht ausreichend. Diese Erkenntnis deckt sich mit den im Abschnitt 2.2.2 erarbeiteten Anforderungen und Kriterien eines Videoannotationsystem. Neben zusätzlichen Farben und einfachen geometrischen Formen wie Linien, Rechtecke und Kreise wurde angemerkt, dass eine Funktion zur Texteingabe wichtig wäre. Obwohl die Texteingabe mittels On-Screen-Tastatur nach dem ersten Prototyp durch Toucheingabe ersetzt wurde, wird sie von den Usern als wichtige zusätzliche Annotationsmöglichkeit gesehen. Ein weiterer Grund für den Wunsch nach einer Texteingabe könnte der begrenzte Platz zum Annotieren sein. So wurde von einigen Usern und auch während den Gesprächen angemerkt, dass die Halbierung des Bildschirms ein gewisser Nachteil im Bezug auf den zur Verfügung stehenden Platz für Annotationen ist.

Ein Problem bei der derzeitigen Interaktion zum Erstellen von Annotationen ist, dass eine Annotation immer nur genau für ein bestimmtes Frame gilt. Es stellte sich heraus, dass es für gewisse Anwendungsfälle durchaus wichtig ist, dass ein bestimmter Zeitraum festgelegt werden kann, für den eine Annotation gilt. Das hat auch damit zu tun, dass die Inhalte eines Videos dynamisch sind und sich bewegen bzw. verändern. Außerdem wurde auch angeregt, neben der Geltungsdauer einer Annotation auch die Geschwindigkeit mit der eine Annotation ein- und ausgeblendet wird variieren zu können.

Neben der fehlenden Möglichkeit annotierte Videos zu exportieren, gab es auch weitere Wünsche zum Import von Videos. Beim Prototyp können im Moment nur Videos aus der Photo-Library des iPads und YouTube geladen werden. Videos in die Photo-Library zu importieren um sie danach mit der App annotieren zu können ist umständlich. Viele User sind es gewöhnt, dass Dropbox oder ähnliche Dienste, um auf externe Daten zugreifen zu können, direkt in der App integriert sind.

Eine weitere Anregung, die im Zuge der Gespräche auftauchte, war das Pausieren und Abspielen des Videos durch einen Tap auf das Video zu ermöglichen. Dies war im originalen Interaktionskonzept eigentlich auch so geplant (vgl. 5.3). Allerdings hat der im Prototyp zum Abspielen von Videos verwendete MPMoviePlayerController diese Funktion standardmäßig nicht implementiert. Ein kleines Detail, das sich allerdings stark auf die Usability auswirkt.

6.3 Mögliche Weiterentwicklung

Das Feedback im Bezug auf mögliche Anwendungsszenarien der App war überraschend vielfältig. Es wurden sehr viele interessante Anwendungsfälle genannt, für die eine App zum intuitiven und kollaborativen Annotieren von Videos hilfreich wäre. Durch das erhaltende Feedback und

die damit zusammenhängende Recherche wurde entschieden, dass die App das einfache und schnelle annotieren von Videos ermöglichen soll und somit verschiedene Anwendungsgebiete abdecken kann, die nicht zu komplex sind. Das schließt Anwendungen im wissenschaftlichen Bereich aus, die zwar durchaus auch von einer solchen App profitieren könnten, aber oft viele weitere und umfangreichere Anforderungen mit sich bringen, welche wiederum den Rahmen der App überschreiten würden. Neben dem Feedback trug zu dieser Entscheidung auch bei, dass es zur allgemeinen Videoannotation weder für den Desktop, noch für Tablets, neben denen speziell für die Sportanalyse entwickelten (vgl. Abschnitt 4.1) kaum Anwendungen gibt.

Dieses Kapitel gibt in weiterer Folge einen Ausblick auf die mögliche Weiterentwicklung der App basierend auf dem Feedback, sowie den theoretischen Erkenntnissen dieser Arbeit.

Kollaborative Verbesserungen

Wie im vorhinein angenommen und durch das Feedback sowie die Recherche bestätigt, werden Annotationen meist nicht ausschließlich zum persönlichen Gebrauch erstellt. In vielen Situationen ist es wichtig das annotierte Video mit anderen austauschen zu können. Auch das gleichzeitige und gemeinsame Annotieren kann in gewissen Fällen sinnvoll sein und bei der zukünftigen Weiterentwicklung ist durchaus zu überlegen, ob man auch das synchrone annotieren des gleichen Videos von mehreren Usern auf verschiedenen Tablets ermöglichen sollte. In der nächsten Phase sollte der Fokus allerdings zunächst auf der Umsetzung von asynchronen Funktionen liegen. Um die Annotationen exportieren bzw. mit anderen Personen zu können gibt es verschiedene Möglichkeiten.

Eine der technisch einfachsten Varianten ist die Annotationen einzeln, als Bilder oder im PDF-Format zu exportieren. Vor allem bei der derzeitigen Implementierung des Prototyps sollte das ohne Probleme möglich sein, da die Annotationen bereits als JPG-Bild gespeichert werden. Die Bilder könnten in die Photo-Library gespeichert werden und zusätzlich bietet das iOS-SDK einen standardisierten Weg an, ein Bild durch die UIActivityViewController-Klasse auf verschiedene Arten zu teilen (Abbildung 6.2). Der vorgefertigte Dialog ermöglicht zum Beispiel das Senden als E-Mail oder Ausdrucken. Ein Problem bei dieser Variante ist, dass nur einzelne Bilder inklusive Annotation exportiert werden und nicht das gesamte Video. Für gewisse Szenarien ist das zu wenig, deshalb sollte es zusätzlich möglich sein das gesamte Video, inklusive der Annotationen, zu exportieren.

Um das Video inklusive den Annotationen zu exportieren, müssen die Annotationen in das Video gerendert werden. Dabei könnte das bereits in Abschnitt 4.6 angesprochene Framework AV Foundation hilfreich sein. Damit ist es möglich dem Video Überlagerungen, einen anderen Hintergrund oder Animationen hinzuzufügen [117]. Im konkreten Fall könnte durch Überlagerung der einzelnen Annotationen an der jeweiligen Stelle im Video ein neues Video generiert werden, welches anschließend exportiert und gespeichert werden kann. Aber auch bei dieser

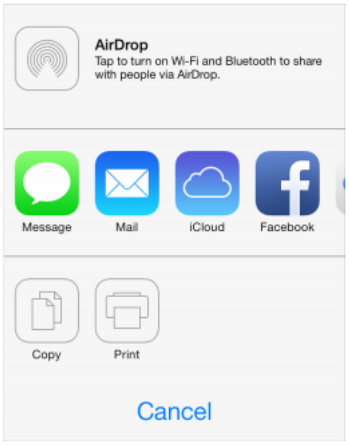

Abbildung 6.2: Dialog zum Teilen von Inhalten unter iOS 7

Variante gehen Informationen verloren, die unter Umständen wichtig sind. Hat man nämlich nur das Video inklusive der Annotationen zur Verfügung, weiß man zum Beispiel nicht, an welcher Stelle im Video sich die Annotationen befinden.

Um die Zeitinformation der Annotationen beim Austauschen nicht zu verlieren ist es sinnvoll diese Metadaten auch exportieren zu können. Dabei sollte das Format so gewählt werden, dass es auch von anderen Programmen zur Videoannotation gelesen werden kann. Geeignete und standardisierte Formate dafür sind MPEG-7 oder das Open Annotation Data Model. (vgl. Abschnitt 2.2.4) Beide Standards verwenden als Dateiformat XML und das iOS-SDK bietet geeignete Klassen, um XML-Dateien zu Lesen und Schreiben [118]. Im Moment werden die Zeiten als NSDate-Objekte in einer SQL-Lite Datenbank gespeichert um sie persistent verfügbar zu machen. Das Speichern im MPEG-7 oder Öpen Annotation Data ModelFormat wäre eine sinnvolle Ergänzung und würde den Austausch ermöglichen.

Um eine asynchrone, kollaborative Zusammenarbeit zu ermöglichen, sollten also die drei folgenden Varianten implementiert werden:

- Exportieren/Importieren der einzelnen Annotationen als Bilder oder gesammelt als PDF
- Exportieren/Importieren des Videos als Videodatei inklusive der erstellen Annotationen
- Exportieren/Importieren der Metadaten (Zeitinformation) mittels MPEG-7 oder Open Annotation Data Model

Diese drei Methoden sollten als Basis für die Weiterentwicklung gesehen werden. In diesem Zusammenhang liegt auf jeden Fall auch die Integration von Cloud-Datenspeicherdiensten nahe. Die vorhandene Anbindung an iCloud, Dropbox und Google Drive wird von vielen Usern

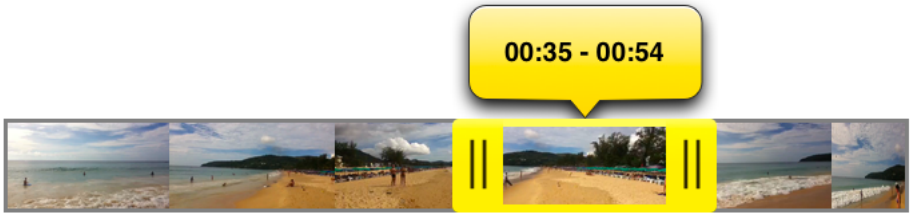

Abbildung 6.3: SAVideoRangeSlider Komponente zum Auswählen eines Zeitbereichs [119]

mittlerweile als selbstverständlich angesehen und hat auf der iOS-Plattform eine weite Verbreitung gefunden. Durch die Verwendung dieser Dienste wird der Austausch von Dateien zwischen verschiedenen Geräten, aber auch zwischen Usern deutlich vereinfacht.

Zusätzliche Annotationsmöglichkeiten

Wie in Abschnitt 2.2.2 beschrieben, ist eine wichtige Anforderung an ein Videoannotationsystem verschiedene Annotationsmöglichkeiten zu bieten. Bei einer Weiterentwicklung der App sind dabei besonders zwei Dinge wichtig. Zum einen ist das die Verbesserung und Erweiterung der vorhandenen Annotationstools. Dazu könnten folgende Funktionen implementiert werden.

- Auswahl von verschiedenen Farben
- Einfache geometrische Formen (Linie, Rechteck, Kreis, Pfeil)
- Einfügen von Bildern
- Redo/Undo
- Radiergummi
- Annotation durch Text mittels On-Screen-Tastatur

Diese Funktionen könnten in der Toolbar am unteren Bildschirmrand platziert werden, die bereits beim Erstellen einer Annotation angezeigt wird. Neben diesen zusätzlichen Funktionen wäre es auf andererseits wichtig die Granularität der Annotationen festlegen zu können. Im Moment gilt eine Annotation im Prinzip immer genau für ein Frame des Videos. Oft möchte man allerdings auch eine Annotation für einen gewissen Bereich im Video erstellen. Ein geeignetes Konzept um einen Zeitbereich auszuwählen ist ein Range-Slider. Er ähnelt einem gewöhnlichen Slider, allerdings können damit zwei Werte und somit einen Bereich festgelegt werden. In iOS wird dieses Prinzip bereits beim Trimmen von Videos angewandt. Das Open-Source-Control SAVideoRangeSlider (Abbildung 6.3) bildet diese Funktion nach [119]. Damit könnte man für

jede Annotation festlegen, ob sie nur für ein bestimmtes Frame oder für einen gewissen Zeitbereich im Video gelten soll. Auch das von einem User angesprochene Festlegen der Ein- und Ausblenddauer einer Annotation könnte durch eine solche Komponente bestimmt werden.

Weitere Funktionen

Im weiteren werden noch diverse zusätzliche Funktionen erläutert, welche die App verbessern würden. Um das Annotieren zu vereinfachen, könnten zusätzliche Playback-Möglichkeiten eingebaut werden. Sinnvolle Ergänzungen wären zum Beispiel das langsame Vor- und Zurückspulen, sowie das Weiterspringen um einzelne Frames. In diesem Zusammenhang sollte auch das Pausieren/Abspielen des Videos mittels einer Tap-Geste auf das Video ermöglicht werden, um die Handhabung zu vereinfachen.

Im Moment unterstützt die App nur das Querformat. Ein Vorschlag wäre bei der Annotationsansicht, durch rotieren des iPads ins Hochformat, automatisch in die Video Notes Ansicht zu wechseln. Das Feedback hat gezeigt, dass die zweigeteilte Ansicht, bei der auf der rechten Seite des Videos noch zusätzlicher Platz für Notizen ist, grundsätzlich sehr gut funktioniert. Das einzige Problem dabei ist, dass das Video dadurch relativ klein dargestellt wird. Dieses Problem könnte durch eine alternative Ansicht gelöst werden. Auch das Hinzufügen einer Zoom-Funktion für den Annotationsbereich wäre sinnvoll. Dadurch erhält man mehr Platz und hat die Möglichkeit auf Details einzugehen.

Eine zusätzliche und sehr Interessante Weiterentwicklung, wäre das Implementieren von automatisierten Ansätzen aus dem Bereich der Computer-Vision (vgl. Abschnitt 2.2.3). Dabei ist zu prüfen, inwiefern die Rechenleistung des iPads dafür ausreicht. Eine einfache Schnitterkennung und eventuell Konturerkennung könnten vermutlich durchaus umgesetzt werden und den Annotierprozess sinnvoll ergänzen.

Zusammenfassung

Die Verwendung von Methoden des UCD und explorativen Ansätzen spielt in der modernen Softwareentwicklung eine wichtige Rolle. Durch den Ansatz des Co-Designs konnte wertvolles Feedback für die eine zukünftige, iterative Weiterentwicklung der App gewonnen werden. Außerdem wurden dadurch eine Vielzahl von interessanten Anwendungsgebieten und -szenarien erarbeitet. Der mögliche Einsatz der App in so unterschiedlichen Gebieten wie dem Bildungsbereich, der Unterhaltung, der Sportanalyse, dem Coaching und der Videoproduktion trugen zu der Entscheidung bei, die App in weiterer Folge so zu gestalten, dass das einfache und intuitive Annotieren ohne Vorkenntnisse ermöglicht wird. Um die App vom Prototyp ausgehend weiter zu entwickeln ist es wichtig, im ersten Schritt zumindest die asynchrone Kollaboration durch geeignete Export- und Importfunktionen zu ermöglichen. Zusätzlich ist der Ausbau der An-

notationsmöglichkeiten ein wichtiger Faktor. Dazu gehört die Erweiterung der zur Verfügung stehenden Annotationstools genauso wie die Möglichkeit zur Anpassung der Granularität von Annotationen. Das überwiegend positive Feedback ist auf jeden Fall eine Bestätigung für das grundlegende Interaktionskonzept und dessen Umsetzung in Form des Prototyps.

KAPITEL 7

Zusammenfassung & Ausblick

Das Annotieren ist eine wichtige Form der intensiven Auseinandersetzung mit Texten, Bildern, Audioaufnahmen oder Videos und ermöglicht das Kommentieren sowie Reflektieren von unterschiedlichen Inhalten. Betrachtet man speziell das Medium Video, so hat es in den letzten Jahren einen sehr interessanten Wandel durchgemacht. Die Entwicklung weg von der professionellen Videoproduktion, hin zu der Erstellung von Videoinhalten durch Amateure und die gleichzeitig immer einfach werdende Verbreitung haben dazu beigetragen, dass Videos für unzählige verschiedene Zwecke verwendet werden. Aus diesem Blickwinkel betrachtet ist die Entwicklung von geeigneter Anwendungen zum Annotieren von Videos ein durchaus logischer Schritt. Das Problem bei bereits existierenden Anwendungen in diesem Bereich ist, dass sie meist sehr komplex und nicht besonders intuitiv zu bedienen sind.

Das Ziel dieser Arbeit war es, die Potentiale und Einsatzmöglichkeiten einer Tablet-App zum intuitiven und kollaborativen Annotieren von Video zu erforschen. Die Verwendung eines Tablets bietet durch die Eingabe mittels Touchscreen neue Möglichkeiten im Bezug auf das Interface- und Interaktionsdesign. Vor allem durch die gewonnene Nähe zum Nutzer bietet der Tablet als Plattform die ideale Voraussetzung für ein NUI und im speziellen Fall für eine App zum Annotieren von Videos. Es ist wichtig die Vor- und Nachteile der auf Gesten basierenden Interaktion zu verstehen und in das Design der App miteinzubeziehen.

Ein System zur Videoannotation besitzt außerdem gewisse Voraussetzungen die beachtet werden sollten. Neben der Navigation durch Videos, muss auch das Auswählen von Teilen des Videos mit unterschiedlicher Granularität ermöglicht werden sowie unterschiedliche Formen der Annotation verfügbar sein. Annotationen sind neben der persönlichen Verwendung auch oft für weitere Personen wichtig. Aus diesem Grund lag es Nahe sich auch den Aspekt des kollaborativen Arbeiten genauer anzusehen. Gelungene Kollaboration kann zu Lösungen führen die ohne Zusammenarbeit nicht möglich gewesen wären. Im Zusammenhang mit der Videoan-

notation gibt es unterschiedliche Ansätze Kollaboration zu ermöglichen. Vom einfachen asynchronen Austausch von Annotationen bis hin zur gleichzeitigen Annotation eines Videos durch mehrere Nutzer reichen die in dieser Arbeit vorgestellten Ansätze.

Neben der intensiven Beschäftigung mit den theoretischen Grundlagen wurde gleichzeitig ein Prototyp für das iPad entwickelt. Durch Gespräche mit ausgewählten Personen die Erfahrung im Bereich der Videoannotation besitzen und auf der anderen Seiter durch die Veröffentlichung des Prototypen im Apple App Store konnte wichtiges Feedback zur explorativen Weiterentwicklung gewonnen werden. Resultierend aus der Vielzahl der unterschiedlichen Anwendungsszenarien wurde die Entscheidung getroffen eine allgemeine App für das einfache und intuitive Annotieren ohne nötige Vorkenntnisse zu entwickeln. Das sehr positive Feedback zum Prototyp und der mögliche Einsatz in so unterschiedlichen Bereichen wie dem E-Learning, der Unterhaltung, der Sportanalyse, dem Coaching und der Videoproduktion zeigen zum einen dass das Interaktionskonzept der App funktioniert und zum anderen, dass durchaus ein Bedarf für eine solche App vorhanden ist.

Darüber hinaus wurden Verbesserungsmöglichkeiten erarbeitet, die für eine zukünftige Weiterentwicklung der App wichtig sind. Besonders der Ausbau der kollaborativen Funktionen in Form von geeigneten Möglichkeiten die Annotationen mit anderen Nutzern zu teilen, sowie die Erweiterung der Annotationsmöglichkeiten sind essentiell für eine sinnvolle Verbesserung.

Literaturverzeichnis

[1] Gonzalo Ramos and Ravin Balakrishnan. Fluid interaction techniques for the control and annotation of digital video. In *Proceedings of the 16th annual ACM symposium on User interface software and technology*, UIST '03, page 105–114, New York, NY, USA, 2003. ACM.

[2] Michael Kipp. Anvil: The video annotation research tool. *Handbook of Corpus Phonology. Oxford University Press, Oxford (to appear, 2011)*, 2007.

[3] IBM research - VideoAnnEx annotation tool. `http://www.research.ibm.com/VideoAnnEx/`.

[4] Behavioral research software and observation labs | the observer XT, September 2012.

[5] Jun Gong and Peter Tarasewich. Guidelines for handheld mobile device interface design. In *In Proceedings of the 2004 DSI Annual Meeting*, 2004.

[6] Sean E. Ellis and Dennis P. Groth. A collaborative annotation system for data visualization. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '04, page 411–414, New York, NY, USA, 2004. ACM.

[7] C.C. Marshall and A.J.B. Brush. Exploring the relationship between personal and public annotations. In *Proceedings of the 2004 Joint ACM/IEEE Conference on Digital Libraries, 2004*, pages 349–357, 2004.

[8] Dan Saffer. *Designing Gestural Interfaces*. O'Reilly Media, 1 edition edition, November 2008.

[9] James H. Carlisle. Evaluating the impact of office automation on top management communication. In *Proceedings of the June 7-10, 1976, national computer conference and exposition*, AFIPS '76, page 611–616, New York, NY, USA, 1976. ACM.

[10] ACM SIGCHI curricula for human-computer interaction : 2. definition and overview of human-computer interaction. `http://old.sigchi.org/cdg/cdg2.html#2_1`.

[11] Fakhreddine Karray, Milad Alemzadeh, Jamil Abou Saleh, and Mo Nours Arab. Human-computer interaction: Overview on state of the art. *International Journal on Smart Sensing and Intelligent Systems*, 1(1):137–159, 2008.

[12] Daniel Wigdor and Dennis Wixon. *Brave NUI World: Designing Natural User Interfaces for Touch and Gesture*. Elsevier, April 2011.

[13] CLI-GUI-NUI Grafik Wikipedia. `http://en.wikipedia.org/w/index.php?title=File:CLI-GUI-NUI.png&oldid=475196243`, December 2013. Page Version ID: 475196243.

[14] Gideon Steinberg. Natural user interfaces. In *ACM SIGCHI Conference on Human Factors in Computing Systems*, 2012.

[15] B. Shneiderman. Direct manipulation: A step beyond programming languages. *Computer*, 16(8):57–69, 1983.

[16] E.A. Johnson. Touch displays a novel input/output device for computers. *Electronics Letters*, 1(8):219–220, 1965.

[17] Nimish Metha. A flexible machine interface. *MA Sc. Thesis, Department of Electrical Engineering, University of Toronto*, 1982.

[18] Jefferson Y. Han. Low-cost multi-touch sensing through frustrated total internal reflection. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*, UIST '05, page 115–118, New York, NY, USA, 2005. ACM.

[19] Jeff han: The radical promise of the multi-touch interface | video on TED.com. `http://www.ted.com/talks/jeff_han_demos_his_breakthrough_touchscreen.html`.

[20] Bill Buxton. Multi-touch systems that i have known and loved. `http://www.billbuxton.com/multitouchOverview.html`.

[21] Shumin Zhai, Per Ola Kristensson, Caroline Appert, Tue Haste Andersen, and Xiang Cao. Foundational issues in touch-screen stroke gesture design-an integrative review. *Foundations and Trends in Human-Computer Interaction*, 5(2):97–205, 2012.

[22] Meredith Ringel Morris, Jacob O. Wobbrock, and Andrew D. Wilson. Understanding users' preferences for surface gestures. In *Proceedings of Graphics Interface 2010*, GI '10, page 261–268, Toronto, Ont., Canada, Canada, 2010. Canadian Information Processing Society.

[23] Mark Johnson. *The body in the mind: The bodily basis of meaning, imagination, and reason.* University of Chicago Press, 1987.

[24] Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. User-defined gestures for surface computing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, page 1083–1092, New York, NY, USA, 2009. ACM.

[25] Gordon Kurtenbach, Thomas P. Moran, and William Buxton. Contextual animation of gestural commands. In *Computer Graphics Forum*, volume 13, page 305–314, 1994.

[26] Andrew Bragdon, Robert Zeleznik, Brian Williamson, Timothy Miller, and Joseph J. LaViola,Jr. GestureBar: improving the approachability of gesture-based interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, page 2269–2278, New York, NY, USA, 2009. ACM.

[27] Olivier Bau and Wendy E. Mackay. OctoPocus: a dynamic guide for learning gesture-based command sets. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, UIST '08, page 37–46, New York, NY, USA, 2008. ACM.

[28] A. Chris Long,Jr., James A. Landay, Lawrence A. Rowe, and Joseph Michiels. Visual similarity of pen gestures. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, CHI '00, page 360–367, New York, NY, USA, 2000. ACM.

[29] A. Chris Long, James A. Landay, and Lawrence A. Rowe. quill: A gesture design tool for pen-based user interfaces. Technical report, 2001.

[30] Gestures | android developers. `http://developer.android.com/design/patterns/gestures.html`.

[31] iOS human interface guidelines: Interactivity and feedback. `https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/InteractivityInput`.

[32] Huawei Tu, Xiangshi Ren, and Shumin Zhai. A comparative evaluation of finger and pen stroke gestures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, page 1287–1296, New York, NY, USA, 2012. ACM.

[33] Ahmed Sabbir Arif and Cristina Sylla. A comparative evaluation of touch and pen gestures for adult and child users. 2013.

[34] Ken Hinckley, Koji Yatani, Michel Pahud, Nicole Coddington, Jenny Rodenhouse, Andy Wilson, Hrvoje Benko, and Bill Buxton. Pen + touch = new tools. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, UIST '10, page 27–36, New York, NY, USA, 2010. ACM.

[35] Donald A. Norman and Jakob Nielsen. Gestural interfaces: a step backward in usability. *interactions*, 17(5):46–49, September 2010.

[36] Cisco visual networking index: Forecast and methodology, 2012–2017 [visual networking index (VNI)]. `http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html`.

[37] Majority of tablet users watch video on their device, 1 in every 4 viewers pay to watch. `http://www.comscore.com/Insights/Press_Releases/2012/6/Majority_of_Tablet_Users_Watch_Video_on_their_Device`.

[38] Oskar Juhlin, Goranka Zoric, Arvid Engström, and Erika Reponen. Video interaction: a research agenda. *Personal and Ubiquitous Computing*, pages 1–8.

[39] Habibul Haque Khondker. Role of the new media in the arab spring. *Globalizations*, 8(5):675–679, 2011.

[40] David Kirk, Abigail Sellen, Richard Harper, and Ken Wood. Understanding videowork. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, page 61–70, New York, NY, USA, 2007. ACM.

[41] Kris Luyten, Kristof Thys, Steven Huypens, and Karin Coninx. Telebuddies: social stitching with interactive television. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '06, page 1049–1054, New York, NY, USA, 2006. ACM.

[42] Cédric Courtois and Evelien D'heer. Second screen applications and tablet users: constellation, awareness, experience, and interest. In *Proceedings of the 10th European conference on Interactive tv and video*, EuroiTV '12, page 153–156, New York, NY, USA, 2012. ACM.

[43] VideoANT: extending online video annotation beyond content delivery. *TechTrends*, 54(3):45–49, May 2010.

[44] A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, 2000.

[45] Stamatia Dasiopoulou, Eirini Giannakidou, Georgios Litos, Polyxeni Malasioti, and Yiannis Kompatsiaris. A survey of semantic image and video annotation tools. In Georgios Paliouras, Constantine D. Spyropoulos, and George Tsatsaronis, editors, *Knowledge-Driven Multimedia Information Extraction and Ontology Evolution*, number 6050 in Lecture Notes in Computer Science, pages 196–239. Springer Berlin Heidelberg, January 2011.

[46] VideoANT - video annotation tool [academic technology services, UMN]. `http://ant.umn.edu/`.

[47] Mathias Lux and Michael Riegler. Annotation of endoscopic videos on mobile devices: a bottom-up approach. In *Proceedings of the 4th ACM Multimedia Systems Conference*, MMSys '13, page 141–145, New York, NY, USA, 2013. ACM.

[48] Diogo Cabral, João G. Valente, Urândia Aragão, Carla Fernandes, and Nuno Correia. Evaluation of a multimodal video annotator for contemporary dance. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, page 572–579, New York, NY, USA, 2012. ACM.

[49] Video § annotations at harvard. `http://www.annotations.harvard.edu/icb/icb.do?keyword=k80243&pageid=icb.page466612`.

[50] Khushboo Khurana and MB Chandak. Study of various video annotation techniques. *International Journal of Advanced Research in Computer and Communication Engineering*, 2(1), 2013.

[51] Youngtae Seok and Howon Lee. WalkieTagging: efficient video annotation method based on spoken words for smart devices. In *2012 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, pages 459–462, 2012.

[52] Irena Koprinska and Sergio Carrato. Temporal video segmentation: A survey. *Signal Processing: Image Communication*, 16(5):477–500, January 2001.

[53] Guozhu Liu and Junming Zhao. Key frame extraction from MPEG video stream. In *2010 Third International Symposium on Information Processing (ISIP)*, pages 423–427, 2010.

[54] Yueting Zhuang, Yong Rui, T.S. Huang, and S. Mehrotra. Adaptive key frame extraction using unsupervised clustering. In *1998 International Conference on Image Processing, 1998. ICIP 98. Proceedings*, volume 1, pages 866–870 vol.1, 1998.

[55] C. Sujatha and U. Mudenagudi. A study on keyframe extraction methods for video summary. In *2011 International Conference on Computational Intelligence and Communication Networks (CICN)*, pages 73–77, 2011.

[56] Dan B. Goldman, Chris Gonterman, Brian Curless, David Salesin, and Steven M. Seitz. Video object annotation, navigation, and composition. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, UIST '08, page 3–12, New York, NY, USA, 2008. ACM.

[57] A.D. Bagdanov, M. Bertini, A. Del Bimbo, G. Serra, and C. Torniai. Semantic annotation and retrieval of video events using multimedia ontologies. In *International Conference on Semantic Computing, 2007. ICSC 2007*, pages 713–720, 2007.

[58] Phillipe Salembier and Thomas Sikora. *Introduction to MPEG-7: Multimedia Content Description Interface*. John Wiley & Sons, Inc., New York, NY, USA, 2002.

[59] Mario Döller and Nikolaus Lefin. Evaluation of available MPEG-7 annotation tools. *Proceedings of IMEDIA/I-SEMANTICS*, 7, 2007.

[60] Open annotation data model. `http://www.openannotation.org/spec/core/`.

[61] Open annotation. `http://www.openannotation.org/`.

[62] Graham Klyne and Jeremy Carroll. Resource description framework (RDF): concepts and abstract syntax. Technical report.

[63] Daniel Stoller-Schai. E-collaboration: die gestaltung internetgestützter kollaborativer handlungsfelder. *Institut of Information Management*, (2767):332, 2003.

[64] Kollaboration. `http://www.duden.de/rechtschreibung/Kollaboration`.

[65] Barbara Gray. *Collaborating: Finding common ground for multiparty problems*, volume 329. Jossey-Bass San Francisco, 1989.

[66] Michael Schrage. *Shared Minds: The New Technologies of Collaboration*. Random House Inc., New York, NY, USA, 1990.

[67] Pierre Dillenbourg, Michael J. Baker, Agnès Blaye, and Claire O'Malley. The evolution of research on collaborative learning. *Learning in Humans and Machine: Towards an interdisciplinary learning science.*, pages 189–211, 1995.

[68] Kjeld Schmidt and Liam Bannon. Taking CSCW seriously: Supporting articulation work. *Computer Supported Cooperative Work (CSCW)*, 1(1-2):7–40, 1992.

[69] Ansgar Gerlicher. Computer-supported cooperative work (CSCW) — kollaborative systeme und anwendungen. In Roland Schmitz, editor, *Kompendium Medieninformatik*, pages 143–195. Springer Berlin Heidelberg, Berlin, Heidelberg.

[70] CSCW-Matrix Grafik Wikipedia. `http://en.wikipedia.org/wiki/File:Cscwmatrix.jpg`.

[71] Robert Johansen. *GroupWare: Computer Support for Business Teams*. The Free Press, New York, NY, USA, 1988.

[72] Hala Skaf-Molli, Claudia Ignat, Charbel Rahhal, and Pascal Molli. New work modes for collaborative writing. pages 176–182, July 2007.

[73] J. Grudin. Computer-supported cooperative work: history and focus. *Computer*, 27(5):19–26, 1994.

[74] Stephanie Teufel. *Computerunterstützung für die Gruppenarbeit*. Addison-Wesley, 1995.

[75] Clarence A. Ellis, Simon J. Gibbs, and Gail Rein. Groupware: Some issues and experiences. *Commun. ACM*, 34(1):39–58, January 1991.

[76] Niall Cook. *Enterprise 2.0: How Social Software Will Change the Future of Work*. Gower Publishing, Ltd., January 2008.

[77] Saul Greenberg, Carl Gutwin, and Andy Cockburn. Awareness through fisheye views in relaxed-WYSIWIS groupware. In *Proceedings of the Conference on Graphics Interface '96*, GI '96, page 28–38, Toronto, Ont., Canada, Canada, 1996. Canadian Information Processing Society.

[78] Carl Gutwin and Saul Greenberg. Workspace awareness for groupware. In *Conference Companion on Human Factors in Computing Systems*, CHI '96, page 208–209, New York, NY, USA, 1996. ACM.

[79] Isaak Kavasidis, Simone Palazzo, Roberto Di Salvo, Daniela Giordano, and Concetto Spampinato. An innovative web-based collaborative platform for video annotation. *Multimedia Tools and Applications*, March 2013.

[80] Fish4Knowledge homepage. `http://groups.inf.ed.ac.uk/f4k/`.

[81] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "GrabCut": interactive foreground extraction using iterated graph cuts. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, page 309–314, New York, NY, USA, 2004. ACM.

[82] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, January 1988.

[83] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986.

[84] Renan G. Cattelan, Cesar Teixeira, Rudinei Goularte, and Maria Da Gra\cca C. Pimentel. Watch-and-comment as a paradigm toward ubiquitous interactive video editing. *ACM Trans. Multimedia Comput. Commun. Appl.*, 4(4):28:1–28:24, November 2008.

[85] Vivian G. Motti, Roberto Fagá,Jr., Renan G. Catellan, Maria da Gra\cca C. Pimentel, and Cesar A.C. Teixeira. Collaborative synchronous video annotation via the watch-and-comment paradigm. In *Proceedings of the Seventh European Conference on European Interactive Television Conference*, EuroITV '09, page 67–76, New York, NY, USA, 2009. ACM.

[86] Lara Schibelsky Godoy Piccolo and Maria Cecília C. Baranauskas. Desafios de design para a TV digital interativa. In *Proceedings of VII Brazilian Symposium on Human Factors in Computing Systems*, IHC '06, page 1–10, New York, NY, USA, 2006. ACM.

[87] Gang Zhai, G.C. Fox, M. Pierce, Wenjun Wu, and H. Bulut. eSports: collaborative and synchronous video annotation system in grid computing environment. In *Seventh IEEE International Symposium on Multimedia*, pages 9 pp.–, 2005.

[88] Stefan Wilk, Stephan Kopf, and Wolfgang Effelsberg. Social video: A collaborative video annotation environment to support e-learning. *World Conference on Educational Multimedia, Hypermedia and Telecommunications 2013*, 2013(1):1228–1237.

[89] Guerrilla HCI: article by jakob nielsen. `http://www.nngroup.com/articles/guerrilla-hci/`.

[90] Coach's eye video app. `http://www.coachseye.com/`.

[91] Video analysis & sports coaching app | ubersense. `http://www.ubersense.com/`.

[92] CoachMyVideo - anytime, anywhere video analysis™. `http://www.coachmyvideo.mobi/`.

[93] Scribbeo^{TM} app iOS software | scribbeo^{TM}. http://scribbeo.com/.

[94] Apple developer. https://developer.apple.com/.

[95] Android SDK | android developers. http://developer.android.com/sdk/index.html.

[96] Leah Findlater, Jacob O. Wobbrock, and Daniel Wigdor. Typing on flat glass: Examining ten-finger expert typing patterns on touch surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, page 2453–2462, New York, NY, USA, 2011. ACM.

[97] File system programming guide: File system basics. https://developer.apple.com/library/mac/documentation/FileManagement/Conceptual/FileSystemProgrammingGuide/FileSystemOverview/FileSystemOverview.html.

[98] iOS technology overview: Core services layer. https://developer.apple.com/library/ios/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/CoreServicesLayer/CoreServicesLayer.html#//apple_ref/doc/uid/TP40007898-CH10-SW30.

[99] LBYouTubeView. https://github.com/larcus94/LBYouTubeView.

[100] iOS technology overview: Media layer. https://developer.apple.com/library/ios/documentation/miscellaneous/conceptual/iphoneostechoverview/MediaLayer/MediaLayer.html#//apple_ref/doc/uid/TP40007898-CH9-SW6.

[101] AV foundation for iOS and OS x - apple developer. https://developer.apple.com/av-foundation/.

[102] Core data programming guide: Introduction to core data programming guide. https://developer.apple.com/library/ios/documentation/cocoa/conceptual/CoreData/cdProgrammingGuide.html.

[103] TestFlight. https://testflightapp.com.

[104] Sketch on video. http://sketchonvideo.com/.

[105] HTML KickStart HTML elements & documentation. http://www.99lime.com/elements/.

[106] Chadia Abras, Diane Maloney-krichmar, and Jenny Preece. User-centered design. In *In Bainbridge, W. Encyclopedia of Human-Computer Interaction. Thousand Oaks: Sage Publications*. Publications, 2004.

[107] Karel Vredenburg, Ji-Ye Mao, Paul Smith, and Tom Carey. A survey of user-centered design practice. pages 471–478, 2002.

[108] Elizabeth B.-N. Sanders and Pieter Jan Stappers. Co-creation and the new landscapes of design. *CoDesign*, 4(1):5–18, 2008.

[109] Joan Greenbaum and Morten Kyng. *Design at Work: Cooperative Design of Computer Systems*. CRC Press, April 1991.

[110] Froukje Sleeswijk Visser, Pieter Jan Stappers, Remko van der Lugt, and Elizabeth B-N Sanders. Contextmapping: experiences from practice. *CoDesign*, 1(2):119–149, 2005.

[111] Emiliano Miluzzo, Nicholas D. Lane, Hong Lu, and Andrew T. Campbell. *Research in the App Store Era: Experiences from the CenceMe App Deployment on the iPhone*.

[112] iTunes connect. `https://itunesconnect.apple.com/WebObjects/iTunesConnect.woa`.

[113] Google analytics. `http://www.google.at/intl/de/analytics/`.

[114] HockeyApp - the platform for your apps. `http://hockeyapp.net/features/`.

[115] ATLAS.ti: the qualitative data analysis & research software. `http://www.atlasti.com/index.html`.

[116] MAXQDA - professionelle QDA software für die qualitative datenanalyse - MAXQDA – the art of data analysis. `http://www.maxqda.de/`.

[117] AVFoundation tutorial: Adding overlays and animations to videos. `http://www.raywenderlich.com/30200/avfoundation-tutorial-adding-overlays-and-animations-to-videos`.

[118] NSXMLDocument class reference. `https://developer.apple.com/library/mac/documentation/cocoa/reference/foundation/Classes/NSXMLDocument_Class/Reference/Reference.html#//apple_ref/doc/uid/20002406-2228`.

[119] andrei200287/SAVideoRangeSlider · GitHub. `https://github.com/andrei200287/SAVideoRangeSlider`.