FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Entwurf und Entwicklung einer verbesserten Middleware für das Tangible Table ToolkiT

MAGISTERARBEIT

zur Erlangung des akademischen Grades

Magister der Sozial- und Wirtschaftswissenschaften

im Rahmen des Studiums

Informatikmanagement

eingereicht von

Dieter Schwarzinger
Matrikelnummer 9502732

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung
Betreuer: Thomas Grechenig

Wien, 23.08.2010

(Unterschrift Verfasser) (Unterschrift Betreuer)

Entwurf und Entwicklung einer verbesserten Middleware für das Tangible Table ToolkiT

MAGISTERARBEIT

zur Erlangung des akademischen Grades

Magister der Sozial- und Wirtschaftswissenschaften

im Rahmen des Studiums

Informatikmanagement

eingereicht von

Dieter Schwarzinger

9502732

ausgeführt am

Institut für Rechnergestützte Automation

Forschungsgruppe Industrial Software

der Fakultät für Informatik der Technischen Universität Wien

Betreuung:

Betreuer: Thomas Grechenig

Wien, 23.08.2010

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benützt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Wien, am 23.08.2010 ----------

Dieter Schwarzinger

Danksagung

Ich möchte die Gelegenheit nutzen, um kurz jene Personen namentlich zu ehren, die mir in der anstrengenden Zeit während meiner Magisterarbeit mit Rat und Tat zur Seite standen und aus deren Zuspruch ich immer neue Kraft schöpfen konnte.

In allererster Linie bedanke ich mich bei meinem Betreuer Univ.-Prof. DI Dr. Thomas Grechenig, dass ich im Rahmen des Seminars für DiplomandInnen über dieses Thema schreiben durfte.

Daneben gilt meine besondere Anerkennung Dr. Martin Tomitsch, der mich selbst über noch so weite geografische Entfernungen unterstützte und mich regelmäßig mit nützlichen Ratschlägen und seinem umfangreichen Fachwissen versorgte. Da ich aber auch am INSO Institut wichtige Praxisarbeiten verrichten musste, möchte ich auch DI Christoph Wimmer Respekt zollen, der mich vor Ort über technische Details informierte und hilfreich beriet.

Zu guter Letzt bin ich meiner Familie und ganz besonders meinen Eltern Christine und Erich Schwarzinger zu Dank verpflichtet, welche mich jahrelang während meiner schulischen und universitären Ausbildung unterstützten und stets aufmunterten, falls es einmal nicht so gelaufen ist, wie ich es erhofft hatte.

Kurzfassung

Die Bedeutung des Forschungsbereichs der Tangible User Interfaces (TUI) hat in den letzten Jahren stark zugenommen. TUIs unterstützen auf natürliche Weise die haptischen Fähigkeiten des Menschen, indem digitale Informationen durch physische Repräsentationen visualisiert werden. Manipulationen des realen Objekts haben direkten Einfluss auf seine virtuelle Verkörperung im System. Je nach Applikation kann dabei beinahe jeder beliebige Gegenstand digital erweitert und als tangibles Steuerungsobjekt dargestellt werden. Die Bedienung des TUIs ist intuitiv und ohne langwierige Einschulung nachvollziehbar. Außerdem erleichtert die interaktive Eingabemöglichkeit kooperatives Arbeiten in der Gruppe.

Allerdings kam es bislang zu keiner Standardisierung der Computerschnittstellen. Zusätzlich sind die Anschaffung und Lizenzierung kommerzieller Systeme mit einem hohen Kostenaufwand verbunden. Als kostengünstige Alternative zu herkömmlichen TUI Systemen wird im Rahmen der vorliegenden Magisterarbeit das Tangible Table ToolkiT (4T) Framework, entwickelt von der Forschungsgruppe INSO der Technischen Universität Wien, sowie die zugrundeliegenden Technologien im Detail vorgestellt. Es dient der Gestaltung von TUI Applikationen, insbesondere jener der tischbasierten Variante Tangible Tabletop. Zusammen mit dem 4T Rückprojektionstisch bildet das Framework ein innovatives TUI System. Des Weiteren werden auch die Unterschiede von TUIs gegenüber herkömmlichen GUIs vergleichend diskutiert.

Die vorliegende Magisterarbeit umfasst die technische Beschreibung und intensive Analyse des 4T Frameworks sowie seine Erweiterung um reacTIVision. Im Zuge des 4T Projekts wurde ein Framework geschaffen, das mittels optischer Trackingtechnologien (ARToolKit, reacTIVision) und geeignetem Rahmensystem (OpenTracker) vielfältige Gestaltungsmöglichkeiten bietet. Dabei wird die didaktische Relevanz physischer Verkörperungen und Manipulationen zur Gestaltung intuitiver Computerschnittstellen anhand erkenntnistheoretischer Aspekte betrachtet. Der Designer muss über keine bemerkenswerten Programmierkenntnisse verfügen und kann sich auf die Umsetzung seiner Ideen konzentrieren. Zusammenfasend kann die vorliegende Arbeit als Leitfaden für die Entwicklung neuer TUI Anordnungen herangezogen werden.

Keywords: Tangible Table ToolkiT, Tangible User Interface, intuitive Interaktion, Objekt Tracking, Augmented Reality, physische Manipulation

Abstract

The area of tangible user interfaces (TUI) has spread widely in the last few years and its significance is continuously growing. TUIs support human tactile cognition in a natural way by visualizing digital information through physical representations. Therefore manipulations of real objects influence directly their virtual embodiments in the system. Depending on the TUI application nearly any object can be digitally enhanced and used as a tangible control device. The handling of TUIs is intuitive and does not require long instructions. Furthermore the interactive input method facilitates cooperation in groups.

Yet no common standard has been established among its various interface implementations. At the same time commercial implementations are still subject to high costs and tight licensing issues. In the context of this diploma thesis the Tangible Table ToolkiT (4T) framework is presented as a low priced alternative to conventional TUI systems and described in detail as well as its underlying technologies. Provided by the research group for Industrial Software (INSO) it serves as a design tool for developing TUI applications, in particular for tangible tabletops. Together with the 4T rear projection table the framework builds an innovative TUI system on site. In addition differences between TUIs and GUIs are comparatively discussed.

This thesis focuses on the detailed technical description and intensive analysis of the 4T framework as well as its expansion by adding reacTIVision. In the course of the 4T project a framework has been established which offers various approaches for designing TUI-based applications. Prerequisites for this purpose are both the implementation of optical tracking technologies (ARToolKit, reacTIVision) and the usage of another extensible framework (OpenTracker). Another concerning matter regards the didactical relevance of physical embodiments and manipulations to generate intuitive computer interfaces on the basis of cognitive science. Designers do not need to be experienced programmers. They can concentrate their efforts on putting their ideas into action. Furthermore it provides reliable background information as a guideline for the design of new TUI systems.

Keywords: Tangible Table ToolkiT, tangible user interface, intuitive interaction, object tracking, augmented reality, physical manipulation

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Thema dieser Magisterarbeit ist die Erweiterung bzw. Fertigstellung des Tangible Table ToolkiT[1] (4T) Projekts (Framework und Rückprojektionstisch) sowie seine detaillierte technische Beschreibung und Analyse. Das System wurde von der Forschungsgruppe für Industrielle Software (INSO)[2] der Technischen Universität Wien entwickelt, um Designern – ohne Know-how der eingesetzten Technologie – Gestaltungsmöglichkeiten im Bereich der Tangible Tabletop Applikationen zu bieten. Die Arbeiten am 4T gliedern sich in mehrere Abschnitte.

Zuerst wird am 4T Rückprojektionstisch eine für das optische Tracking geeignete Kamera installiert und kalibriert. Zeitgleich während/mit der Fertigstellung des 4T Interfaces erfolgt die zweite Phase, bei der es die 4T Middleware zu erweitern bzw. anzupassen gilt. Dabei wird die ursprünglich verwendete Trackingtechnologie ARToolKit durch das zuvor nicht verfügbare reacTIVision Toolkit ersetzt. Schlussendlich soll eine bereits existierende 4T Applikation adaptiert, getestet und auf dem erweiterten System korrekt ausgeführt werden.

1.1 Problemstellung

Hiroshi Ishii und Brygg Ulmer gaben bereits 1997 einen ersten Einblick in die frühen Forschungsarbeiten im Bereich Tangible User Interface (TUI), als sie ihre Abhandlung über Tangible Bits (Ishii & Ullmer, 1997) der Öffentlichkeit präsentierten. Hierbei steuert der Benutzer[3] – im Gegensatz zum herkömmlichen Graphical User Interface (GUI), welches Tastatur, Maus und Monitor verwendet – durch Manipulation physischer Artefakte (z.B. tangible Objekte, Pucks, Blöcke, etc.) die Eingabe. Die Steuerungsobjekte sind mit digitaler Information verknüpft und repräsentieren auf dem TUI ihre Bedeutung für das System gemäß ihrer Anordnung/Interaktion in der Realität. Tangible Tabletop beschreibt diese Benutzerschnittstelle als interaktive Tischoberfläche.

Das institutseigene 4T wurde erstellt, um die Entwicklung von interaktiven Tisch-Applikationen voranzutreiben und umzusetzen. Außerdem sind kommerzielle Lösungen im Tangible Tabletop Bereich aufgrund ihrer Komplexität sehr kostspielig.

---

[1] Ein Toolkit ist eine Softwaresammlung von Bibliotheken, Schnittstellen und Klassen. Dem Entwickler werden standardisierte Methoden und Funktionen bereitgestellt, um neue Applikationen leichter erstellen zu können.

[2] http://www.inso.tuwien.ac.at/

[3] In diesem Dokument wird aus Gründen der besseren Lesbarkeit die grammatisch maskuline Form (generisches Maskulinum) verallgemeinernd für Männer und Frauen verwendet, also in Fällen, in denen das Geschlecht nicht bekannt oder für den jeweiligen Zusammenhang unwichtig ist. So können mit der Bezeichnung „der Benutzer“ männliche und weibliche Personen gemeint sein.

4T setzt sich aus einer interaktiven Arbeitsfläche, einem Projektor und einem Arbeitsrechner zusammen. Der Rückprojektionstisch wird im Rahmen der Magisterarbeit durch die Installation einer Digitalkamera mit USB/FireWire-Anschluss komplettiert. Die Kamera verfolgt die physischen Steuerungsobjekte (Pucks), welche sich auf der Arbeitsfläche befinden, anhand ihrer optischen Markierungen. Die Vollendung des Hardwareaufbaus für den Rückprojektionstisch stellt auch die erste große Aufgabe dar.

Die 4T Middleware beinhaltet zwei wichtige Applikationen, den Konfigurator für Einstellungen am Framework (Anzahl der interaktiven Steuerungsobjekte, Verhältnis zwischen Projektor und Trackingfläche) und den Simulator, der die jeweilige Tangible Tabletop Anwendung und Interaktion der Artefakte ohne Hardwareeinsatz am Bildschirm nachahmt.

Bei optischen Trackingsystemen finden bislang zwei Technologien Verwendung, die für die gängigsten Betriebssysteme verfügbar sind:

- Das ARToolKit Framework (http://www.hitl.washington.edu/artoolkit/) dient zur Entwicklung von Augmented Reality Applikationen. Dabei werden virtuelle Bilder über Ausschnitte der realen Welt gelegt, um neue Interaktionsmöglichkeiten für den Benutzer zu schaffen. Die Softwarebibliothek ist bereits in 4T integriert, da es auch bei zweidimensionalem Tracking erfolgreich eingesetzt werden kann.
- Das reacTIVision Toolkit (Kaltenbrunner & Bencina, 2007) ermöglicht – wie das ARToolKit – Tracking über Marker, die an Artefakten befestigt wurden. Das Hauptaugenmerk liegt bei tischbasierten TUIs und interaktiven Oberflächen mit Multi-Touch Funktion. Daher bietet sich im Rahmen der vorliegenden Magisterarbeit seine vollständige Eingliederung in das 4T Projekt des INSO an.

Die vollständige Integration des reacTIVision Toolkits in die 4T Middleware ermöglicht robustes und exaktes zweidimensionales Tracking am Rückprojektionstisch. Dadurch wird die Entwicklung von neuen TUI Applikationen für Designer bedeutend attraktiver.

1.2 Motivation

4T gibt dem interessierten Designer die Möglichkeit zur Erschaffung verschiedenster Tangible Tabletop Applikationen, ohne dass dieser Kenntnis über die Technik der zugrunde liegenden Hard- und Middleware haben muss. Damit ist er in der Lage, selbstständig innovative Multimediaanwendungen in diesem Bereich zu entwerfen.

Hierzu wird das bereits existierende 4T Framework am Institut betrachtet und sein Aufbau samt Erweiterungen (Hard- und Software) detailliert beschrieben. Schluss-

endlich wird eine bestehende TUI Applikation ausgewählt, an die neuen Bedingungen der Middleware angepasst und am optimierten System getestet.

1.3 Zielsetzung

Durch die Darstellung der bisherigen institutsseitigen Arbeit am 4T Framework und seiner Fertigstellung werden die Entwicklung von Hard- und Software im Tangible Tabletop Bereich und seine Grundlagen präsentiert. Im Zuge dessen werden auch die Unterschiede von TUIs gegenüber herkömmlichen GUIs betrachtet und Anwendungsbeispiele dem Leser näher gebracht. Zudem wird die didaktische Relevanz physischer Verkörperungen und Manipulationen zur Gestaltung intuitiver Computerschnittstellen anhand erkenntnistheoretischer Aspekte betrachtet.

In weiterer Folge werden die beiden Open-Source Technologien für optisches Tracking – das (bereits im 4T integrierte) ARToolKit und das reacTIVision Toolkit – ausführlich beschrieben. Nachdem auch das reacTIVision Toolkit über das OpenTracker Rahmensystem in die 4T Middleware implementiert worden ist, werden beide miteinander verglichen und auf ihre Funktionalität in Verbindung mit der vorhandenen Systemkonfiguration evaluiert. Nebenbei wird 4T um eine weitere Softwarebibliothek erweitert.

Schlussendlich steht es aber dem Benutzer frei, ob er das bereits erprobte ARToolKit oder das neu integrierte reacTIVision Framework bei seinen TUI Applikationen einsetzen möchte. Der modulare Aufbau des 4T Frameworks ermöglicht stets einen bequemen und einfachen Wechsel zwischen den beiden optischen Trackingverfahren.

1.4 Aufbau der Arbeit

- Kapitel 2 beschreibt die Grundidee von Tangible Interfaces, vergleicht die herkömmliche grafische mit der angreifbaren Benutzerschnittstelle und zeigt anhand bekannter Tangible Tabletop Beispiele die Anwendungsmöglichkeiten in diesem Bereich.
- Anschließend beleuchtet Kapitel 3 die didaktischen Zusammenhänge von physischen Verkörperungen und zeigt neue Wege in der Unterrichtsgestaltung.
- Das bestehende System vor seiner Middleware Erweiterung wird dann in Kapitel 4 vorgestellt und genau beschrieben.
- Im darauf folgenden Kapitel 5 wird auf die Erweiterung durch das reacTIVision Toolkit in 4T eingegangen und im Detail ausführlich erläutert.
- Abschließend fasst Kapitel 6 die wesentlichen Erkenntnisse zusammen.

2 Tangible Computing

Das folgende Kapitel bietet eine allgemeine Einführung in Tangible Interfaces. Dabei werden die Unterschiede zur traditionellen grafischen Benutzerschnittstelle betrachtet. Abschließend werden anhand von ausgewählten Beispielen gelungene Umsetzungen auf dem Gebiet präsentiert.

2.1 Tangible User Interface

Die Vision, die Hiroshi Ishii am Massachusetts Institute of Technology (MIT)[4] Media Laboratory erdachte und 1997 gemeinsam mit Brygg Ullmer in der Abschrift über *Tangible Bits* (Ishii & Ullmer, 1997) zu Papier brachte, präsentiert sich einem Außenstehenden vielleicht wie aus einem Science Fiction Film entsprungen.

Digitale Information (Bits) erscheint den Benutzern greifbar bzw. tangibel, indem sie mit Objekten und Flächen aus dem täglichen Umfeld verknüpft wird, die sich im Zentrum als auch am Rand ihrer Aufmerksamkeit befinden. Dieser Brückenschlag zwischen Virtualität und Realität erweitert die Umgebungswahrnehmung der Menschen und ihren Umgang mit physischen Artefakten.

Hiroshi Ishii nennt diesen Ansatz der Interaktion zwischen Mensch und Computer tangible Benutzerschnittstelle, engl. *Tangible User Interface (TUI)*. Im Gegensatz zu grafischen Benutzerschnittstellen können mehrere Personen ein TUI gleichzeitig bedienen. Die Benutzer sind also praktisch in der Lage, an ein und demselben Tangible Interface problemlos zu arbeiten. Aber auch mehrere TUIs sind miteinander vernetzbar. Diese kooperative Möglichkeit bietet u.a. das elektro-akustische Musikinstrument reacTable (siehe Abbildung 2.1) von Jordà et al. (2005), ein TUI in Form eines Tisches, dessen reacTIVision Framework auch im 4T zum Einsatz kommt.

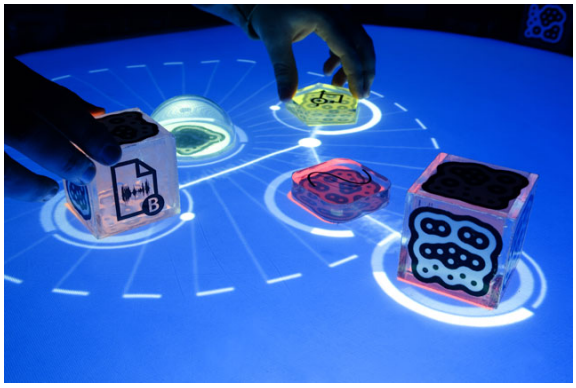


Abbildung 2.1: reacTable

[4] http://web.mit.edu/

Herkömmliche Benutzerschnittstellen werden zwar immer komplexer, erlauben es aber nicht, dass der Benutzer viel mehr als seine visuelle Wahrnehmung einsetzt, um sich Problemlösungen zu stellen.

Intuitive Interaktion (Strubel & Zimmermann, 2005) und die Möglichkeit, alle menschlichen Sinne (vor allem den Tastsinn als manipulativen Faktor) zu nutzen, sind wesentliche Argumente dafür, dass TUI Anwendungen einem breiteren Publikum vorgestellt werden. Zudem ist es ein weiterer Schritt, der zur Verbesserung von Mensch-Computer-Interaktion, engl. *Human-Computer Interaction (HCI)*, beiträgt. Dieses Teilgebiet der Informatik bemüht sich durch benutzergerechte Gestaltung von interaktiven Systemen und ihren Mensch-Computer-Schnittstellen, die menschliche Fähigkeiten und Eigenschaften enger mit einzubeziehen.

Diese grundlegenden Gedanken greift die *Marble Answering Machine* (Crampton Smith, 1995) von Durrell Bishop auf, die er als Student am Royal College of Art (RCA)[5] entworfen hat (siehe Abbildung 2.2).

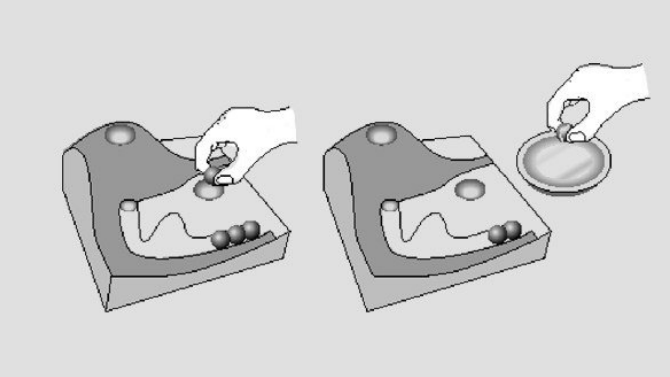


Abbildung 2.2: Marble Answering Machine

Der Anrufbeantworter speichert eingegangene Sprachmitteilungen in Form von Murmeln. Der Benutzer kann nun die Nachricht nicht nur „angreifen“, sondern sie abhören oder die Nummer des versäumten Anrufs im erweiterten Telefon wählen, indem er die Murmel auf der entsprechenden Vertiefung platziert.

Im Grunde lassen sich viele Szenarios rund um TUIs generieren, da die physische Verkörperung digitaler Information nahezu alle denkbaren Formen annehmen kann. Diese Symbiose zwischen menschlicher Umwelt und virtuellem Raum bietet viele Möglichkeiten zur Umsetzung.

[5] http://www.rca.ac.uk/

Allerdings stehen Industriedesigner der momentanen Entwicklung im TUI Bereich kritisch gegenüber. Sie bemängeln, dass sich die Benutzerschnittstellen bei ähnlichen *Tangible Products* (Djajadiningrat et al., 2004) vom technischen Prinzip her kaum oder gar nicht unterscheiden. Die innovative Realisierung physischer Manipulation ist ihrer Meinung bereits wieder festgefahren, wenn Rotation und Translation die primären Charakteristika der Interaktion durch den Benutzer sind.

Der Designprozess von tangiblen Interaktionsobjekten sollte keiner Standardisierung unterworfen sein, sondern auf eine mehr wahrnehmungsorientierte Präsentation hinarbeiten, um so die haptischen Fähigkeiten des Menschen (Erkennen von Druck, Berührung und Vibrationen auf der Haut) besser einzubinden.

2.1.1 TUI versus GUI

Im Bereich HCI ist die grafische Benutzeroberfläche, engl. *Graphical User Interface (GUI)*, bislang die vorherrschende Variante der Computernutzung. Ein GUI gibt dem Benutzer die Möglichkeit, Anwendungen mittels grafischer Elemente (Icons, Toolbars, Dialogfenster, etc.) zu steuern. Für gewöhnlich geschieht das mit der Maus als Steuergerät. Die Tastatur erlaubt komplexere Eingaben (z.B. bei Textverarbeitungsprogrammen) und Tastenkombinationen, die die Eingabemöglichkeiten erweitern. Über einen Bildschirm lassen sich die Aktionen verfolgen, nach dem Motto *„what you see is what you get“*.

Im Wesentlichen trennen jedoch diese Art der Bedienung den virtuellen Raum von der Außenwelt, da die Information zwar am Bildschirm (ähnlich einem gemalten Bild) wahrgenommen wird, jedoch nirgends körperlich präsent ist. Es besteht keine Synergie.

Das Modell-Präsentation-Steuerung Modell, engl. *Model-View-Controller (MVC)*, in Abbildung 2.3 zeigt die Beziehung zwischen Computer In- und Output bei einem GUI (Ullmer & Ishii, 2001). Das Modell enthält die darzustellenden Daten. Die physische Steuerung per Maus und Tastatur ist deutlich von der digitalen Präsentation mittels Bildschirm getrennt (durch den schwarzen Balken symbolisiert).

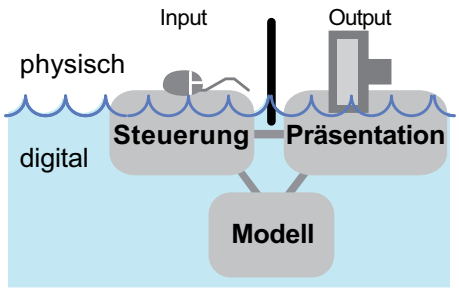


Abbildung 2.3: GUI Interaktionsmodell – MVC Modell

Das erweiterte MVC Modell, engl. *Model-Control-Representation (physical and digital) (MCRpd)*, ergänzt die reine digitale Präsentation aus dem Vorgänger um die tangible physische Komponente, beide werden zusammengefasst (siehe Abbildung 2.4). Sowohl Steuerung als auch digitale (Rep-d) und physische (Rep-p) Repräsentation sind im TUI integriert und mit dem zugrundeliegenden digitalen Datenmodell verknüpft. Die dargestellte Information ist im Gegensatz zum GUI für den Benutzer tatsächlich greifbar.

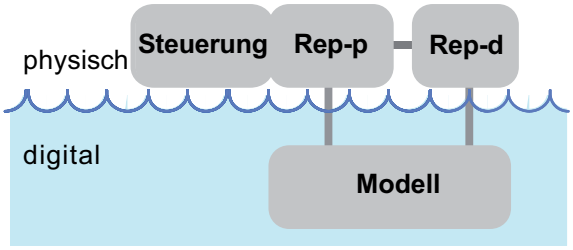


Abbildung 2.4: TUI Interaktionsmodell – MCRpd Modell

Hier soll also eine Umgebung geschaffen werden, die nicht mehr zwischen künstlich und wirklich unterscheidet, sondern beides harmonisch verknüpft (siehe Abbildung 2.5). Alltagsgegenstände und räumliche Gegebenheiten, etc. – Dinge, mit denen der Mensch permanent aktiv in Berührung kommt – repräsentieren zugleich digitale Information, die in der Form nun angreifbar und (durch Drehen, Verschieben, Aufheben, etc.) manipulierbar ist. Sie treten hierbei jedoch nicht als Fremdkörper in den Vordergrund, sondern sind in die tägliche Umgebung als *greifbare Medien* nahtlos integriert.

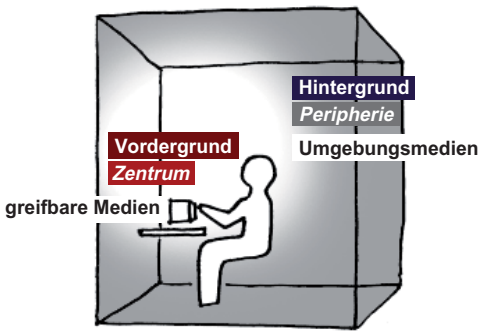


Abbildung 2.5: Die Aufmerksamkeit des Benutzers im Raum

Auch der Hintergrund des Geschehens spielt eine große Rolle. Sogenannte *Umgebungsmedien* wirken als sekundäre Informationsquellen in einer erweiterten Peripherie. Licht, Schatten, Geräusche, Wind- und Wasserbewegungen dienen als Trägermaterial. Die Datenerfassung wirkt passiv und kann beispielsweise wie ein Fenster, durch das man

täglich das Wetter außerhalb des Wohnraums verfolgen kann, ohne permanent darauf achten zu müssen, verstanden werden.

Somit wäre nicht mehr allein die Computerschnittstelle, sondern die ganze Umwelt – vom Zentrum unserer Aufmerksamkeit bis zur Peripherie – Interface zum virtuellen Raum (siehe Abbildung 2.6).

Abbildung 2.6: Vom GUI zum TUI

Bei TUIs finden sich durchaus Ähnlichkeiten zu weiteren HCI Modellen wieder. So wird *Ubiquitous* bzw. *Pervasive Computing* als erstmalige Methode verstanden, bei der zwar Computer und Datenverarbeitung allgegenwärtig sind, jedoch in den Hintergrund treten und in Folge dessen unsichtbar werden. Man spricht auch vom „*verschwundenen Computer*“. Der Alltag und die Umgebung des Menschen sind durch den erweiterten Rechnereinsatz bereichert, der durch seine fehlende wahrnehmbare Präsenz allerdings völlig unaufdringlich wirkt (Weiser, 1991). Dieser Denkansatz bezieht aber auch GUIs mit ein, sodass er sich doch erheblich vom tangiblen Grundgedanken abhebt.

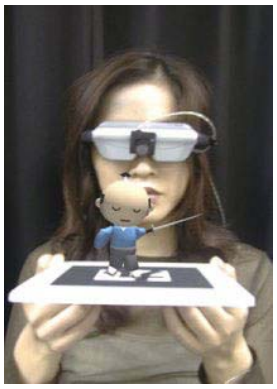


Abbildung 2.7: AR erweitert die Realität um virtuelle Objekte

Augmented Reality (AR) wiederum sieht sich als Erweiterung bzw. Vergrößerung des realen Raumes (Wellner et al., 1993). Über den Wahrnehmungsbereich der realen Welt hinaus werden zwei- oder dreidimensionale virtuelle Objekte gelegt, welche dann über

technische Hilfsmittel, wie z.B. ein Head-Mounted Display (HMD)[6], für den Benutzer sichtbar werden (siehe Abbildung 2.7). TUIs sollen Realität und Cyberspace auch miteinander symbiotisch verknüpfen, allerdings indem digitale Information physisch greifbar gemacht wird.

Die wichtigsten vier Charakteristika des TUIs nach Ullmer & Ishii (2001) sollen anhand des bereits vorgestellten MCRpd Modells noch einmal formuliert werden (siehe Abbildung 2.8):

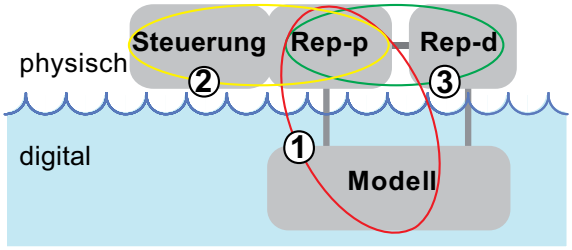


Abbildung 2.8: MCRpd Modell – Merkmale von TUIs

1. Physische Repräsentation (Rep-p) ist an ihre zugrundeliegende digitale Information (Modell) gebunden.
2. Physische Repräsentationen verkörpern Mechanismen zur interaktiven Manipulation (Steuerung).
3. Physische Repräsentationen sind an ihre digitale Repräsentationen (Rep-d) deutlich erkennbar gekoppelt.

Schlussendlich ergibt sich die letzte Eigenschaft aus den drei vorangegangenen:

4. Der physische Zustand der Objekte verkörpert zu einem großen Teil den digitalen Zustand des Systems.

Demnach fusionieren die digitale und die greifbare Welt miteinander, physische Gegenstände (des Alltags) verkörpern digitale Informationen. Der so geschaffene barrierefreie Raum erlaubt intuitive Interaktionen und Manipulationen mit den tangiblen Steuerungsobjekten.

2.1.2 Tangible Tabletop

Das tischbasierte TUI, ein sogenanntes *Tangible Tabletop*, ist eine Variante, wie man dieses spezielle Computerschnittstellen Konzept umsetzen kann.

[6] Ein HMD ist ein visuelles Ausgabegerät, das der Benutzer in einer transparenten Brille auf dem Kopf trägt. Es projiziert digitale Bilder entweder auf einen integrierten Bildschirm (vor das Auge) oder direkt auf die Netzhaut.

Der Tisch bzw. seine Fläche (engl. Tabletop) wird seit jeher von Menschen als Zentrum der Kommunikation genutzt. Im Alltag und Berufsleben ist es ein gewohntes Bild, wenn mehrere Personen sich daran zusammensetzen, um z.B. zu diskutieren, zu essen oder zu spielen.

Wie bereits erwähnt, werden bei TUIs Flächen und Gegenstände mit digitaler Information verknüpft. So liegt es auf der Hand, dass auch Tischoberflächen und darauf platzierte Objekte physische und gleichzeitig digitale Repräsentationen im menschlichen Aktionsbereich werden.

Der Aufbau eines solchen tangiblen Tisches ist abhängig von der eingesetzten Tracking Technologie, welche die darauf befindlichen Objekte erkennt und verfolgt. Zwei der gängigsten Methoden sind: Tracking mittels elektromagnetischer Radiofrequenzen bzw. unter Verwendung von optischen Referenzmarkierungen. Beide Varianten sind für das 4T Framework essentiell und in Kapitel 4.2 ausführlich beschrieben.

2.2 Anwendungsbeispiele

Drei ausgewählte Tangible Tabletop Beispiele (reacTable, Music Table und Audiopad) illustrieren die Besonderheiten und Charakteristika tischbasierter TUIs. Außerdem ist ihre zugrunde liegende Tracking Technologie auch im 4T Framework zu finden.

Es wurden ganz bewusst nur interaktive Musikinstrumente als Exemplare ausgesucht, weil vor allem die computermusikalische Darbietung ein ausgezeichnetes Forschungsfeld für neuartige HCI Modelle darstellt (Jordà et al., 2007), wie folgende Kriterien zeigen:

- Die herausragende Kombination aus Präzision und Freiheit macht die musikalische Darbietung zu einem außergewöhnlichen Klangerlebnis, selbst wenn ein hohes Maß an Erfahrung und Können dem Musiker abverlangt wird.
- Ausdruck und Kreativität vereinen sich mit Unterhaltung.
- Kontinuierliche und multidimensionale Steuerung tendiert dazu, allgegenwärtige Datenstrukturen (z.B. Dateien, Ordner, Hyperlinks, etc.) zu meiden, die noch immer eine vorrangige HCI Stellung einnehmen.
- Digitale Hilfsmittel zur musikalischen Darbietung können eine soziale und kollektive Erfahrung darstellen, die sowohl Zusammenarbeit als auch Wettstreit integriert.
- Computermusik bietet den idealen Spielraum, um die verschiedenen Interaktionen von Anfängern und Experten als auch Erwachsenen und Kindern auf diesem Gebiet zu vergleichen und zu erforschen.

Aufgrund ihrer meist komplexen elektroakustischen Zusammensetzung stellen computermusikalische Darbietungen besonders hohe Ansprüche an die Usability einer

Computerschnittstelle. Diese Vielschichtigkeit wird durch die enorme Manipulationsmöglichkeit der zahlreich vorhandenen Parameter erzielt und gibt Designern die Gelegenheit, viele neue Interaktionstechniken zu erforschen und auf dem HCI Gebiet verstärkt zu experimentieren. Die so neu entwickelten und intuitiven Wechselwirkungstechniken müssen sich natürlich nicht zwangsläufig nur auf computerunterstützte Musikinstrumente beschränken, die Anwendungsgebiete von TUIs (jeder Form) sind zahlreich und werden laufend erweitert.

Den Entwicklern zukünftiger Tangible Tabletop Applikationen sollen diese Beispiele aber auch als Anregung dienen, wie TUIs und ihre Steuerungsobjekte verschieden kombiniert und konzipiert werden können, um intuitive und haptische Interaktion während der Benutzereingabe zu fördern, ohne dass die Bedienung des Systems dabei auf irgendeine Art und Weise aufdringlich oder erzwungen erscheint.

2.2.1 Der ReacTable

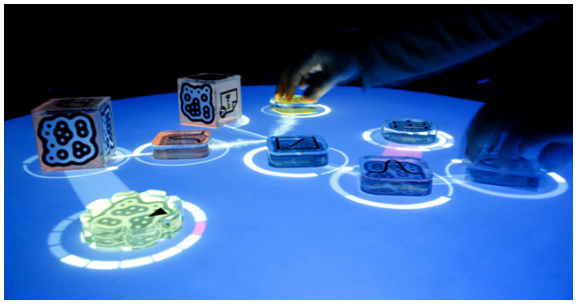


Abbildung 2.9: Der reacTable, ein interaktives Musikinstrument

Der *reacTable*[7] ist ein gemeinschaftliches interaktives Musikinstrument, das an der Universität Pompeu Fabra[8] in Barcelona von der Music Technology Group (MTG) entwickelt wurde (Jordà et al., 2005). Der Tisch erlaubt mehreren Personen durch individuelle Platzierung von Objekten (mit Multi-Touch Unterstützung) gemeinsam zu musizieren (siehe Abbildung 2.9).

[7] http://www.reactable.com/
[8] http://www.upf.edu/

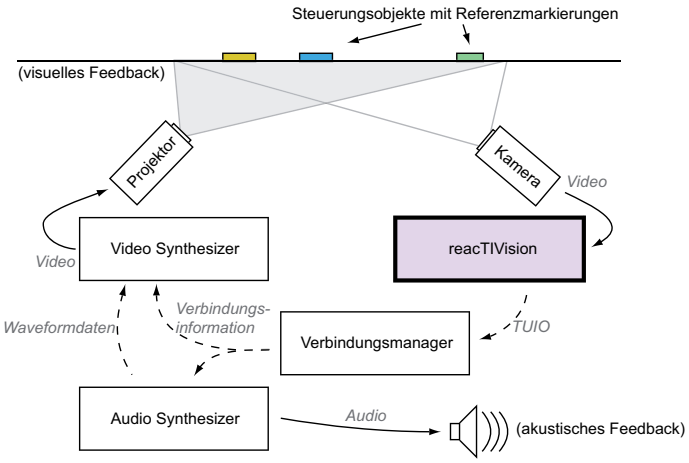


Abbildung 2.10: Aufbau und Funktionalität des reacTables

Bestandteile sowie Funktionsweise des reacTables

Der funktionale Aufbau des reacTables wird in Abbildung 2.10 grob illustriert: Symboltragende Steuerungsobjekte, sogenannte *Tangibles*, bilden Audiostrukturen und werden von einer Digitalkamera anhand ihrer eindeutigen Referenzmarkierung, engl. *Fiducial Identification* (Fiducial ID oder nur Fiducial), erkannt. Dadurch wird die physische Anordnung und Position auf dem Tisch vom reacTIVision Framework (siehe Kapitel 5.1) erfasst und an den Verbindungsmanager weitergeleitet. Dieser verarbeitet die Information und übermittelt die entsprechenden Daten an den Video- und Audio-Synthesizer. Die beiden liefern dann den Benutzern sowohl akustische als auch visuelle Rückmeldungen (durch einen Projektor).

Der reacTable ist rund, besitzt einen Radius von 45 cm und trägt eine lichtdurchlässige, blau beleuchtete Oberfläche (für optisches Tracking und Projektionen), die einen Durchmesser von 76 cm hat. Seine Höhe beträgt 90 cm. Momentan sind zwei reacTable Versionen verfügbar, die sich lediglich durch den Einsatz verschiedener Audio-Synthesizer unterscheiden (für professionelle Musiker bzw. für öffentliche Vor-führungen). Obgleich das Interface von einer einzelnen Person problemlos bedient werden kann, wurde der reacTable mehr für den gemeinschaftlichen Einsatz konzipiert.

Beschreibung sowie Funktion der Tangibles und ihrer Klassen

Auf dem Tisch befinden sich tangible Objekte aus Plexiglas, die bausteinartig wirken und geometrische Figuren darstellen. Bei ihren auffälligen Zeichen handelt es sich um Fiducials, die das reacTIVision Framework als Referenzmarkierungen für das optische Tracking benötigt (siehe Abbildung 2.11).

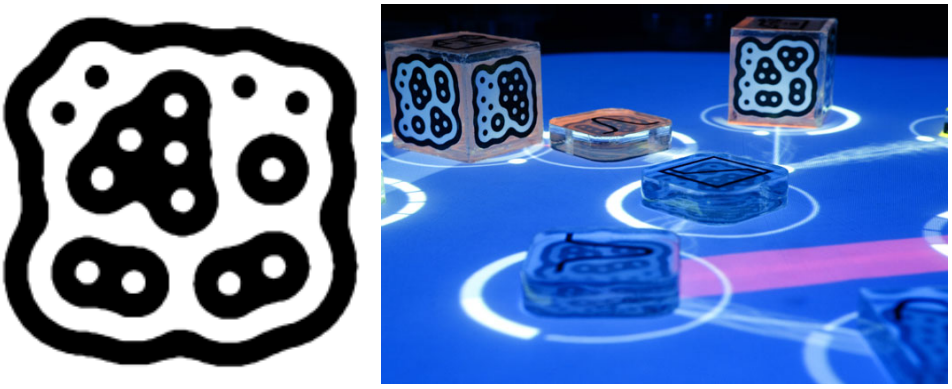


Abbildung 2.11: Fiducial (l.) und tangibles Objekt aus Plexiglas (r.)

Insgesamt gibt es sechs verschiedene tangible Klassen (Jordà et al., 2007), die durch ihre jeweilige geometrische Form unterschieden werden, so sind beispielsweise alle Controller rund. Jede Klasse besteht wiederum aus vielen Unterklassen (siehe Tabelle 2.1).

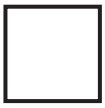
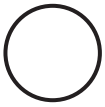
Klasse	Verbindungen	Form	Beispiele
Generator	**1** Audio out **n** Control in		Rechteckwelle Sampler
Audio Filter	**1** Audio in **1** Audio out **n** Control in		Resonanzfilter Flanger
Controller	**1** Control out		Sinuswellen LFO 12-Stufen Sequenzer
Audio Mixer	**2** Audio in **1** Audio out **n** Control in		Mixer Bus Ringmodulator
Control Filter	**1** Control in **1** Control out		Dezimator Sample & Hold
Global	**n** Control in		Metronom Tonalizer

Tabelle 2.1: Die sechs Tangible Klassen des reacTables

Die Tangibles haben aufgrund ihrer Formen und Fiducials unterschiedliche Funktionen (z.B. Filter, Sampler, Mixer, etc.), je nachdem können sie miteinander kombiniert

werden. Ähnlich einem Baukasten setzt man also seine Komposition auf der Tischfläche zusammen. Jede Konstellation ruft andere elektro-akustische Ergebnisse hervor.

Alle tangiblen Objekte sind von einer animierten Kreisprojektion umgeben, die optische Rückmeldung auf das Verhalten sowie Parameter- und Konfigurationswerte liefert – „*the reacTable projection wraps the physical objects with virtual auras*“ (Jordà et al., 2005). Ein 180 Grad Pegel zeigt den aktuellen Rotationsstatus des Steuerungsobjekts an. Auf der anderen Halbkreisbahn lässt sich über den weißen Punkt des Schiebereglers mit dem Finger ein weiterer Parameter editieren. Andere Klassen wie Tonalizer oder 12-Stufen Sequenzer werden über einen fingeraktivierbaren unterteilten Bedienkreis eingestellt.

Abbildung 2.12: Verbindungslinien und virtuelle Auren – Datenfluss und Status

Wie bereits erwähnt können Tangibles untereinander Verbindungen eingehen, der entstehende Datenstrom wird durch eine weiße Linie symbolisiert. Da jede mögliche Objektkombination zu einem anderen (akustischen) Ergebnis führt, nimmt auch die Verbindungslinie jeweils eine unverwechselbare Gestalt an, um u.a. Flussrichtung, Frequenz und Intensität darzustellen (siehe Abbildung 2.12).

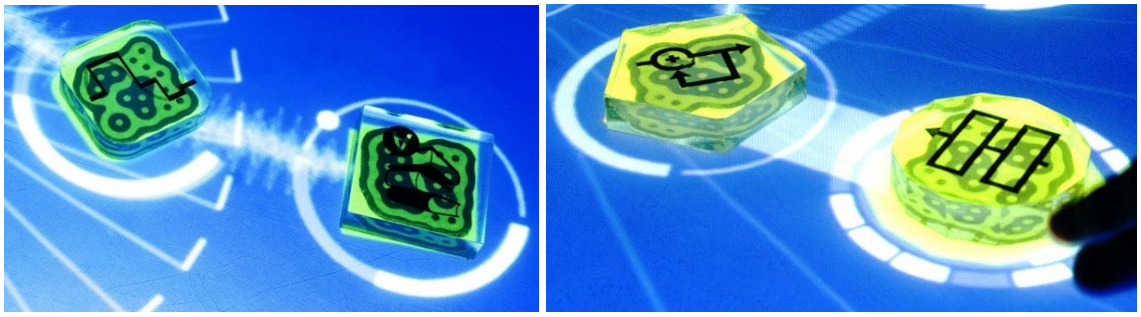


Abbildung 2.13: links Audio Line, rechts Control Line

Bedeutung der Tangibles

Generator und Audio Filter visualisieren ihre Zusammensetzung in Form von resultierenden Wellenlängen. Controller, Control Filter und Global machen ihren Einfluss auf andere Objekte durch variierende Linien (Dichte und Intensität) geltend. Je nachdem, um welche Art der Verknüpfung es sich handelt, spricht man von einer *Audio* bzw. *Control Line* (siehe Abbildung 2.13).

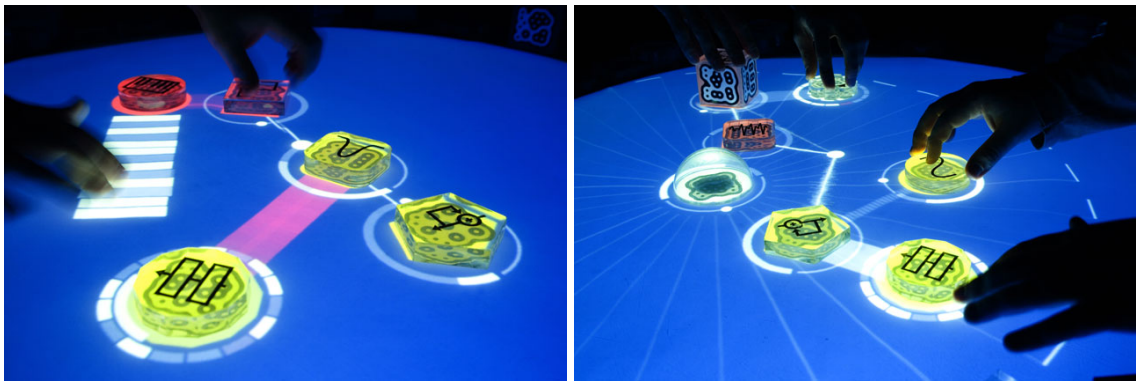


Abbildung 2.14: Rotation, Hand- und Fingerinteraktion

Alle Tangibles werden auf die gleiche Art und Weise verwendet (Translation, Rotation, Hand- und Fingerinteraktion), das akustische und optische Resultat ist jedoch je nach Funktionalität und Anordnung unterschiedlich. Der reacTable wird also durch Manipulation der Steuerungsobjekte und durch vereinfachte Multi-Touch Operationen bedient (siehe Abbildung 2.14).

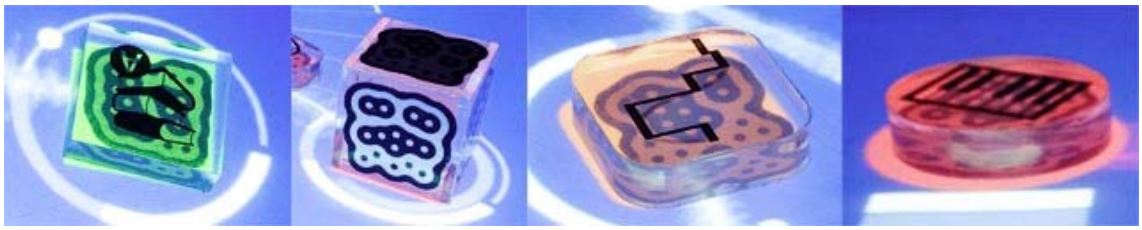


Abbildung 2.15: Generator (und Sampler), Audio Filter und Controller (v.l.n.r.)

Quadrat als Tonerzeuger (*Generator*) bzw. Sampler: Durch Rotation des Sound Generator Quadrats ändert sich seine Frequenz. Der Schieberegler lässt die Lautstärke regeln. Der Sampler fügt aufgenommene Töne (z.B. Schlagzeug Beats, Gitarren Riffs, etc.) ein. Per Handkantenschlag bzw. Fingerberührung unterbricht die Audio Line (sie wird gestrichelt), der Ton verstummt. Durch Berührung des Objekts schaltet man den Ton wieder ein.

Der Generator kann nicht nur Töne und Samples erzeugen. Ein Baustein der Klasse kann mittels Line-In und Voice Control ein Mikrofon oder andere klassische Musikinstrumente in den reacTable integrieren.

Effekte (z.B. Band-Pass, Hoch-Pass, Tief-Pass, etc.) lassen sich durch *Audio Filter* realisieren. Hierfür bringt man ein Filter Objekt in den Aktionsradius eines kompatiblen Tangible. Der Grad der Filterung wird durch die schon bekannte Rotation verändert, der Schieberegler repräsentiert nun den Dry/Wet-Level.

Mittels *Controller* sendet man Steuerungsdaten an das nächstgelegene Objekt. So lassen sich z.B. durch Drehung Schallwellenfrequenzen verändern oder pulsierende Effekte audio-visuell in eine Komposition integrieren.

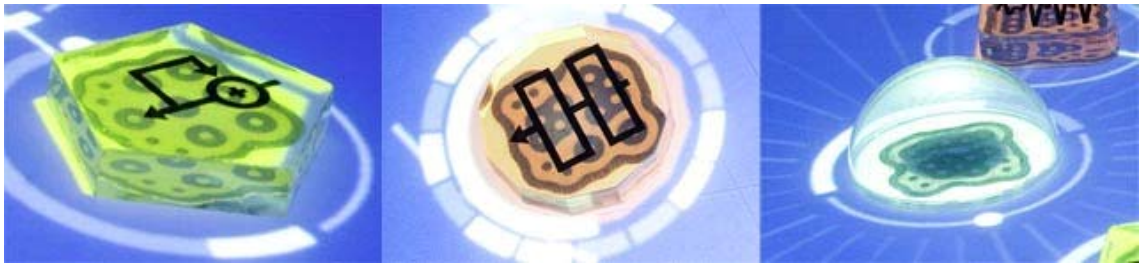


Abbildung 2.16: Audio Mixer, Control Filter und Global (v.l.n.r.)

Weit komplexere Melodien (Loops, etc.) werden durch *Audio Mixer* und *Control Filter* generiert. Das *Global* Element beeinflusst jedes Steuerungsobjekt, dass sich in seiner Umgebung befindet. Der Radius lässt sich durch den Schieberegler skalieren und die Rotation steuert seine Taktrate als Metronom oder Zeitmesser.

Hard- und Softwarekomponenten

Unterhalb der Tischoberfläche befindet sich das eigentliche „Herz“ des reacTables, die Bildverarbeitungssoftware reacTIVision. Sie erhält Input über eine hochauflösende Infrarot Digitalkamera. Der visuelle Output erfolgt über einen Projektor.

reacTIVision erkennt und verfolgt die markierten Tangible anhand ihrer Fiducial IDs in Echtzeit. Für fehlerfreies Tracking benötigt die Digitalkamera mindestens eine Auflösung von 640 x 480 Pixel bei 30 fps (Kaltenbrunner & Bencina, 2007). Die Rechenprozesse erfolgen auf einem gängigen Desktop PC (2 GHz AMD Athlon), die gesammelten Daten (Objekttranslation und -rotation sowie Hand- und Finger-interaktion) werden über das systemeigene TUIO Protokoll (Kaltenbrunner et al., 2005) an den Client übermittelt.

Der *Verbindungsmanager* empfängt und verarbeitet diese Meldungen. Da alle Steuerungsobjekte am Tisch eindeutige, signifikante Fiducials tragen, werden alle Verbindungen, Objektzustände und Parameter dynamisch erfasst. All diese Werte bilden die Berechnungsgrundlage für das Verbindungsnetzwerk, welches der Verbindungsmanager erstellt. Die Netzwerkinformationen werden wiederum an den Audio- und den Video-Synthesizer übermittelt.

Der *Audio-Synthesizer* verwertet die Eingabe des Verbindungsmanagers und dient der Klangerzeugung. Er instanziiert die Tangible Klassen und wandelt ihren Konfigurationsstatus in die entsprechenden Töne um.

Auf der anderen Seite verdeutlicht der *Video-Synthesizer* die Toneffekte mit optischem Feedback. Er ist dafür zuständig, dass alle Verbindungen, Einstellungen und Parameter am Tisch über einen Projektor sichtbar werden. Dazu ist er auch mit dem Audio-Synthesizer verbunden.

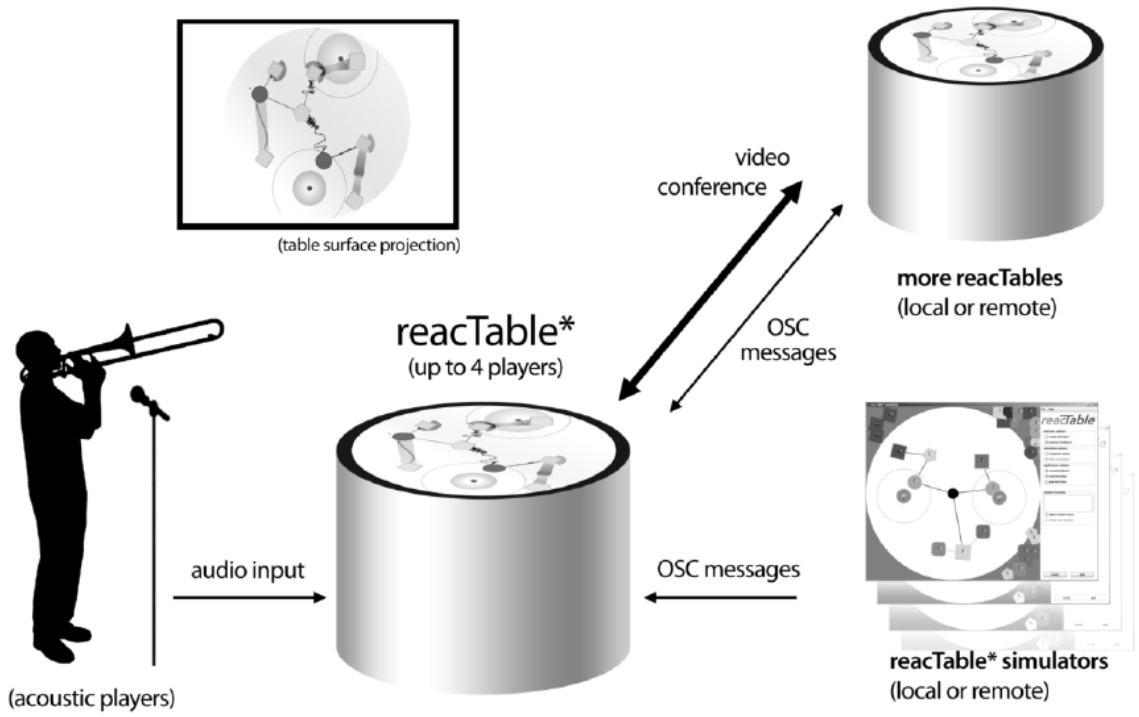


Abbildung 2.17: die gemeinschaftliche reacTable Nutzung (remote oder lokal)

Gemeinschaftliches Arbeiten

Die Zusammenarbeit mehrerer Anwender ist am reacTable sowohl lokal als auch global möglich (siehe Abbildung 2.17). An einem Tisch können aufgrund seiner baulichen Gegebenheiten zwei bis vier Personen agieren. Zum einen können die Benutzer den reacTable unter sich aufteilen, d.h. bei vier Akteuren nimmt jeder ein Viertel des Tisches ein. Die Rollenverteilung ist dann ähnlich einem Orchester oder einer Musik

Band, wo jedes Mitglied ein bestimmtes Instrument spielt. Zum anderen können sie die Tischfläche gemeinsam nutzen, indem einer z.B. das Basisgerüst der Komposition erstellt und ein anderer dieses mittels Filter etc. dynamisch verändert.

Mehrere reacTable können auch in einem Netzwerk global operieren. Die Objekte und Verbindungen aller Tische werden auf den anderen Clients digital repräsentiert. Zwar werden die physischen Repräsentationen nur auf den jeweiligen Tischen greifbar, ihre digitale Information wird aber an alle Installationen gesendet. Somit sind alle Konferenzteilnehmer in der Lage über die zusammengeschlossenen Tische miteinander zu musizieren und zu interagieren.

Während der Entwicklung des reacTable Prototypen wurde auch ein entsprechender *Simulator* entworfen, der die Funktionalität des gesamten Tisches aufweist jedoch über ein GUI angesteuert wird. Deshalb kann er auch in eine Konferenzschaltung eingebunden werden.

2.2.2 Der Music Table

Der *Music Table* besteht aus einer Reihe von Spielsteinen bzw. Blöcke, die sich zu musikalischen Mustern zusammenfügen lassen, um audiovisuelle Kompositionen auf einem Tangible Tabletop darzustellen (Berry et al., 2006). Das TUI mit AR Elementen ist besonders für Kinder geeignet und entwickelt worden, um diese in einer spielerischen Art und Weise an die Welt der Musik heranzuführen (siehe Abbildung 2.18). Dabei steht die intuitive Interaktion mit den tonerzeugenden Blöcken im Vordergrund.

Abbildung 2.18: Der Music Table im Einsatz

Bestandteile sowie Funktionsweise des Music Tables

Die Steuerungsobjekte bzw. Blöcke sind mit optischen Markern des ARToolKit Frameworks versehen und werden von einer über der Tischfläche montierten Kamera erkannt. Ihre Markierungen repräsentieren verschiedene Töne und Funktionen sowie dazugehörige computeranimierte *Ton-Kreaturen* (engl. Note-Creatures), die gemäß den AR Prinzipien über die realen Gegenstände gelegt werden. Die Muster, welche die

Blöcke entsprechend ihrer Anordnung auf dem Tisch annehmen, generieren Töne. Gleichzeitig erhält der Benutzer zu seinen Aktionen permanent optisches Feedback. Wie bereits erwähnt wurde, ist das ARToolKit eine Softwarebibliothek zur Positionsbestimmung dreidimensionaler virtueller Objekte in AR Systemen (Kato & Billinghurst, 1999).

Statt dem üblichen HMD wird beim Music Table allerdings ein großer Plasma-Bildschirm eingesetzt, um den gesamten Operationsbereich aus der Vogelperspektive anzuzeigen. Diese Variante kommt mehr den kindgerechten Anforderungen und Bedürfnissen bei technischen Umsetzungen dieser Art entgegen. Der große Bildschirm und die Steuerungsobjekte am Tisch sind die beiden zentralen Aktivitätsebenen, denen die Aufmerksamkeit der Kinder gelten soll. Dagegen würden HMDs nur negativen Einfluss auf ihre Hand-Augen-Koordination ausüben (Pesce, 2000).

Abbildung 2.19: Karten statt Blöcke aus einer frühen Music Table Version (v.l.n.r.)

Beschreibung sowie Funktion der Blöcke und Ton-Kreaturen

Auf dem Tisch befinden sich im Normalfall bis zu zehn *Ton-Blöcke* (engl. Note-Blocks), die jeweils einen Ton repräsentieren und zusammen ein Musikstück ergeben. In einer frühen Version des Music Tables waren es Karten (siehe Abbildung 2.19), um Töne und Funktionen akustisch und visuell darzustellen, doch aufgrund der besseren Handhabung hat man sich schlussendlich für Blöcke aus Schaumstoff entschieden. Da ein Quader sechs Seiten besitzt, kann er auch genauso viele Funktionen verkörpern (z.B. Instrumenten Block).

Abbildung 2.20: computeranimierte Ton-Kreaturen überlagern die Blöcke

Der Bildschirm zeigt das Geschehen von oben (siehe Abbildung 2.20). Daher ist die Anordnung der Note-Blocks im Grunde einem Notenblatt bzw. einer Tonleiter nachempfunden. Die Tonhöhe steigt von unten nach oben. Ein Block erzeugt daher bei vertikaler Verschiebung (in Relation zur Kameraposition) Akkorde. Von links nach rechts wird seine Position auf einer Zeitebene dargestellt, horizontal nebeneinander gereihte Tonblöcke ergeben somit eine sich wiederholende Tonsequenz. Das System spielt die Abfolge solange, bis ein mitwirkendes Element verändert bzw. ein neues Muster erstellt wird.

Abbildung 2.21: kurzer Ton – links leise, rechts laut

Das ARToolKit Framework erkennt auch gekippte Marker (von der Kamera in Trapez Form wahrgenommen) und hat für jeden eine festgelegte Ausrichtung im Koordinaten-system vorgegeben. Diese Besonderheiten werden für die Einstellung der Lautstärke und Abspieldauer einzelner Töne ausgenutzt. Kippt man einen Ton-Block nach links oder rechts, wird sein Ton kürzer oder länger. Die Lautstärkenregelung geschieht durch Rotation des Blocks ähnlich wie die Drehknopfsteuerung bei Radios. Nach links gedreht wird der Ton leiser, nach rechts lauter.

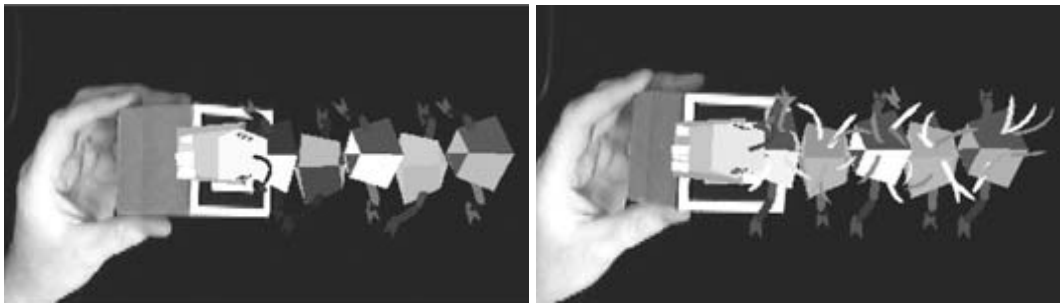


Abbildung 2.22: langer Ton – links leise, rechts laut

Bei diesen Abstimmungen kommen auch die Ton-Kreaturen zum Vorschein. Ihre optische Darstellung spielt eine wesentliche Rolle für das kindgerechte Konzept des Music Tables. Die nahezu lebensechten Gesten und Verhaltensweisen der virtuellen Tiere sollen die Aufmerksamkeit der Kinder an sich binden. Alle Visualisierungen am Music Table sollen das Zentrum der kindlichen Wahrnehmung einnehmen und ein gesteigertes Interaktionsverlangen mit dem Musikinstrument bei den Teilnehmern hervorrufen.

Bedeutung der Blöcke und Ton-Kreaturen

Das AR System legt die dreidimensionalen computeranimierten Figuren über die Markierungen der Ton-Blöcke. Die zunächst kurzen Ton-Kreaturen werden mit ansteigender Abspieldauer immer länger. Je lauter ein Ton gespielt wird, desto mehr Stacheln wachsen einem virtuellen Geschöpf. Umgekehrt wird es glatter, je leiser der Ton wird (siehe Abbildung 2.21 und Abbildung 2.22).

Abbildung 2.23: Ton-Kreaturen „wandern“ von den Blöcken zur Sammelkarte

Um nicht alle Note-Blocks wieder neu ordnen zu müssen, kann eine Tonfolge von bis zu drei Blöcken durch *Sammelkarten* (engl. Phrase Tile) gespeichert werden. Dazu legt der Benutzer einfach eine leere Sammelkarte vor die Kamera, welche seine AR Markierung erkennt und der Kopiervorgang wird eingeleitet. Der Prozess wird durch die „Wanderung“ der Ton-Kreaturen zur Sammelkarte visualisiert. Er ist beendet, wenn

über dem Marker der Karte ein Instrumentensymbol am Bildschirm erscheint (siehe Abbildung 2.23). Das System hat dann die Musteranordnung der Blöcke erfolgreich gesichert. Danach können die Note-Blocks entfernt und neu angeordnet werden. Die gespeicherten Muster können durcheinander gemischt werden, indem man eine Sammelkarte dreht. Das Muster wird wieder gelöscht, indem die Karte hoch gehalten bzw. näher zur Kameralinse bewegt wird, bis das Instrumentensymbol verschwindet.

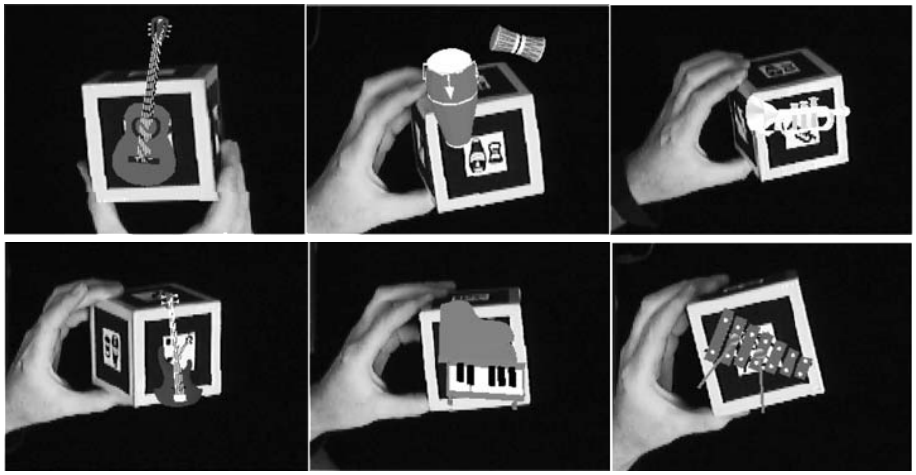


Abbildung 2.24: Der Instrumenten Block – Gitarre, Trommel, Trompete, Bass, Piano und Metallophon

Der *Instrumenten-Block* (engl. Instrument-Block) repräsentiert auf jeder seiner sechs Seiten ein anderes Musikinstrument, das beim Einsatz auf dem Music Table über den entsprechenden Marker akustisch realisiert und optisch dargestellt wird (siehe Abbildung 2.24). Der Schaumstoffblock ist größer als die anderen und hat die Funktion eines Synthesizers.

Lerneffekt bei Kindern

Evaluierungen des Music Tables haben den Lerneffekt auf Kinder (im Alter zwischen sechs und elf Jahren) sowie die allgemeine Usability für diese Zielgruppe untersucht. Die Ergebnisse wurden anschließend diskutiert und für Verfeinerungen an der Benutzerschnittstelle berücksichtigt. Darunter fällt z.B. die Änderung der Steuerungselemente von Karten zu Schaumstoffblöcken, welche die Handhabung des Systems erleichtern.

Ein Forscher erklärte zunächst die interaktiven Aufgabenstellungen. Dann wurde jedem Kind die Möglichkeit geboten, eigenständig am Music Table für eine Stunde zu spielen, was zu interessanten Resultaten führte. Mit steigendem Alter wurden auch aufwendigere Aktionen am Tisch durchgeführt. Kinder, die bereits Erfahrung mit Musikinstrumenten gesammelt hatten, waren schneller imstande komplexere Zusammenhänge

zu verstehen und umzusetzen während jüngere und unerfahrene Teilnehmer eher nur die grundlegenden Anordnungen am Music Table erstellten.

Abbildung 2.25: Der Bush Telegraph

Die beschriebenen Interaktionsgegenstände und Funktionalitäten des Music Tables sind nur als eine aktuelle Momentaufnahme zu verstehen, da das Projekt ständig durch Evaluierungen weiterentwickelt und verbessert wird. Der *Bush Telegraph* (Berry et al., 2004) ist ein Nachfolgemodell und richtet sich mehr an jugendliche Benutzer (siehe Abbildung 2.25). Das Design ist in Anlehnung an die Videospiele Automaten der Achtziger Jahre entstanden. Die Steuerungselemente sind erneut Karten, doch dieser Apparat erlaubt eine gemeinschaftliche Nutzung über eine Netzwerkverbindung zu einem anderen Tisch.

2.2.3 Das Audiopad

Das dritte und letzte Beispiel, das im Rahmen der Tangible Tabletop Systeme präsentiert wird, ist die elektronische Musiksteuereinheit *Audiopad* von Patten et al. (2002). Die bereits mehrfach ausgezeichnete Musikschnittstelle (u.a. bei Designing Interactive Systems 2004 – DIS2004)[9] kombiniert die von knopfbasierten Mischpulten bekannte Steuerung mit innovativer multidimensionaler Trackingtechnologie (siehe Abbildung 2.26). So lassen sich in kürzester Zeit Audio Samples, Tonaufnahmen oder synthetische Melodien zu musikalischen Kompositionen zusammenstellen und modifizieren.

[9] http://www.sigchi.org/DIS2004/mid.php

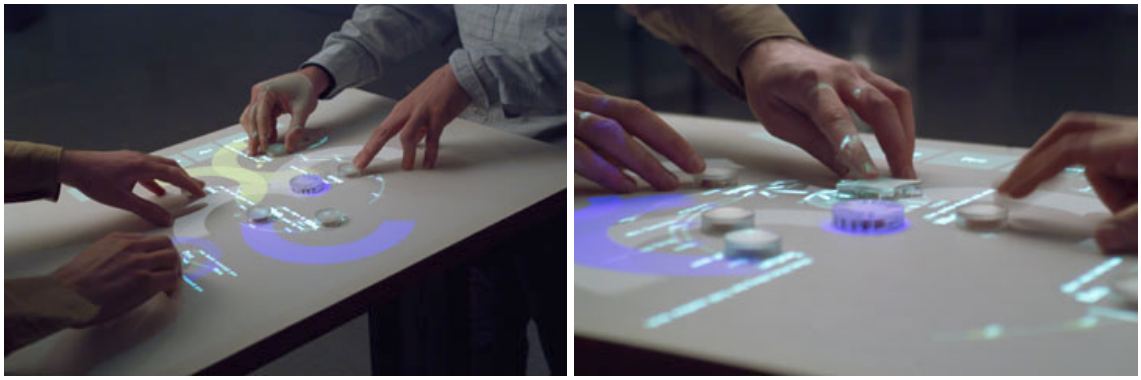


Abbildung 2.26: Das Audiopad

Bestandteile sowie Funktionsweise des Audiopads

Im Gegensatz zu reacTable und Music Table erfolgen Identifizierung und Tracking der Interaktionsobjekte, sogenannte *Pucks*, mittels elektromagnetischer Wellen eines Sensorfeldes nach dem RFID Prinzip (siehe Kapitel 4.2.1). Hierfür tragen die Pucks als Transponder ein bis zwei Radiofrequenz (RF)-Etikettierungen, auch RF-Tags (oder RF-Marker) genannt, die es der Hardware (Empfänger, Transmitter) erlaubt, Position und Orientierung auf der Arbeitsfläche exakt zu bestimmen.

Der Benutzer tätigt durch Verschiebung der Pucks seine Eingabe, um musikalische Parameter (z.B. Lautstärke, Effekte, etc.) einzustellen. Die Audio Informationen und Daten werden über einen Projektor von oben auf und um die Pucks projiziert. So soll sich der Musiker auf das Wesentliche konzentrieren und möglichst viel Flexibilität und Kontrolle über sein Handeln erhalten (siehe Abbildung 2.27).

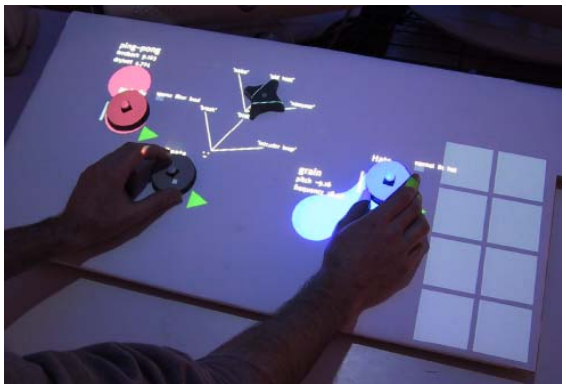


Abbildung 2.27: beidhändige Interaktion mit dem Audiopad

Beschreibung sowie Funktion der Pucks und ihrer RF-Tags

Die eingesetzten RF-Tags, die am Audiopad mit digitaler Information verknüpft werden, sind simpel konstruierte Transponder. Die sogenannten L-C-Tags sind nicht nur preisgünstiger, sondern auch robuster und weniger störanfällig bei der Datenüber-

tragung als andere RF-Marker, allerdings verfügen sie nur über eine geringe Reichweite. Ein L-C-Tag besteht aus einem gewickeltem Kupferdraht und einem Kondensator, dadurch ist er bei einer bestimmten Resonanzfrequenz (je nach Induktivität und Kapazität) schwingungsfähig. Da es sich um passive RF-Tags handelt, müssen sie die dafür notwendige Energie aus einer elektromagnetischen Quelle beziehen, um in Schwingung zu geraten. Als Resonanz bezeichnet man nun jenen Vorgang, bei dem der L-C-Tag durch Energiezufuhr des Transmitters zur Schwingung (bei Eigenfrequenz) anregt wird (siehe Kapitel 4.2.1).

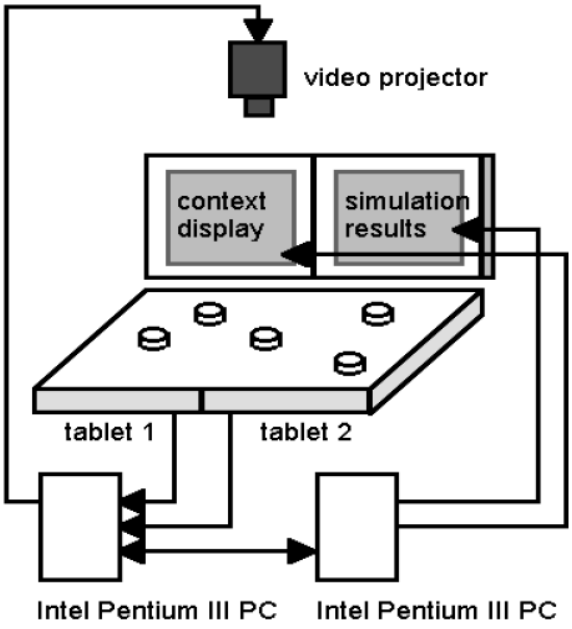


Abbildung 2.28: Aufbau des Sensetables

Die Audiopad Hardware ist eine Weiterentwicklung des vorangegangenen *Sensetables* (Patten et al., 2001). Sein Aufbau ist in Abbildung 2.28 grob beschrieben. Auf der Tischfläche befinden sich die Pucks, die durch das darunter liegende Sensorfeld aktiviert werden. Zwei Arbeitsrechner befinden sich noch weiter unterhalb und betreiben das System. Der eine erhält von der Sensoroberfläche Trackingdaten und visualisiert die daraus resultierenden Informationen über einen Projektor, der oberhalb des Sensetables angebracht ist. Der andere versorgt zwei Bildschirme mit relevanten Informationen bezüglich aller Interaktionen, die am Tisch vor sich gehen. Außerdem dient er zu Simulationszwecken.

Eine modifizierte Variante der Sensorvorrichtung von Ellies Enchanted Garden™ (Zowie™ Intertainment)[10] dient zur Lokalisierung der L-C-Tags. Die optimierte

[10] Zowie™ Intertainment ist mittlerweile Bestandteil der LEGO® Gruppe. Mehr dazu unter http://www.lego.com.

zweidimensionale Trackingfläche besitzt im Inneren Induktionsspulen („Antennen"), die ein elektromagnetisches Wechselfeld erzeugen. Sie reagieren auf die jeweiligen Amplitudenveränderungen in der Resonanzfrequenz einzelner Tags. Je nach Schwingungsweite kann nun die Position des L-C-Tags (auf 4 mm genau) bestimmt werden. Um neben dem Standort auch noch dessen Orientierung festzustellen, verfügen manche Objekte über zwei L-C-Tags. Die relativen Positionen beider Tags liefern die Ausrichtung des Pucks.

Ein elektromagnetisches Tracking System hat gegenüber dem optischen Pendant einen klaren Vorteil, den das Audiopad zu nutzen weiß: Lichtverhältnisse spielen keine Rolle. Optisches Tracking benötigt meist Digitalkameras mit hoher Auflösung, um stabil und korrekt arbeiten zu können. Wenn die visuelle Rückmeldung (via Projektor) aus der gleichen Richtung wie die Erkennung der Marker erfolgen soll (fast immer von unterhalb der Tischfläche), müssen beide Komponenten in unterschiedlichen Lichtbereichen operieren. Sichtbares Licht bleibt der Datenrepräsentation vorbehalten, also muss die Kamera im Infrarot Bereich eingesetzt werden. In beiden Fällen müssen ausreichende Lichtbedingungen gewährleistet sein (Paradiso et al., 2000).

Des Weiteren können Objekte, die mit RF-Tags ausgerüstet sind, weit mehr als nur durch Bewegung im Sensorfeld manipuliert werden. Wie bereits erwähnt, liefern zwei implantierte L-C-Tags Position und Orientierung des Artefaktes. Wird nun ein Tag durch einen am Objekt angebrachten Druckknopfschalter deaktiviert, wird das System auf diese Zustandsänderung hingewiesen und reagiert. Per Knopfdruck kann also ein neues Menü aktiviert (bzw. auch wieder deaktiviert) werden, wobei das Steuerungs-element nun eine andere Funktion als bisher in der Applikation einnimmt und demzufolge neue Resultate erzeugt. Der andere Tag ist immer aktiv und dient nach wie vor der Objektlokalisierung im Sensorfeld. Die mechanische Komponente ist allerdings in der finalen Version des Audiopads nicht integriert.

Der Software Layer, der u.a. die Knopfdruckerkennung und Datenkalkulationen handhabt, wird *Tag Server* genannt. Er leitet auch die entsprechenden Informationen an die Video und Audio Komponenten des Systems weiter.

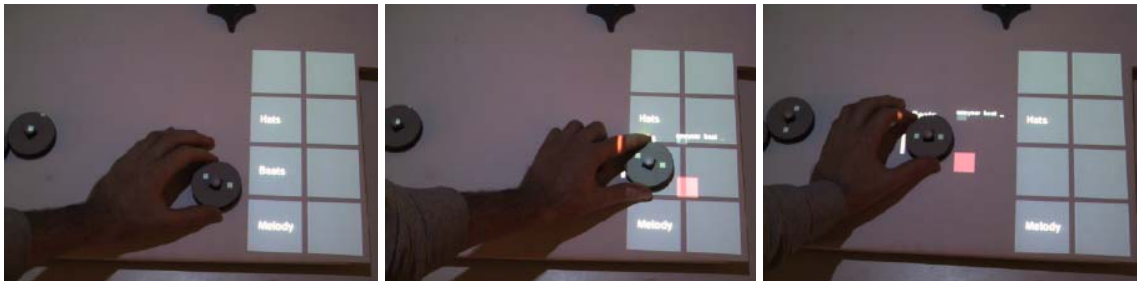


Abbildung 2.29: Zuweisung einer Samples Gruppe im Track Manager

Bedeutung der Pucks

Eine Audiopad Session beginnt, indem man einem Puck eine Samples Obergruppe (z.B. Hats, Beats, Melody, etc.) zuweist. Dazu bewegt der Benutzer den Puck in den *Track Manager* und wählt aus dem Rastergitter (2 x 4 Felder) seine gewünschte Gruppe anhand ihrer Bezeichnung. Danach zieht er den Puck wieder aus dem Gitter. Nun ist der Puck (namentlich und akustisch) belegt, die Samples Obergruppe scheint im Track Manager nicht mehr auf (siehe Abbildung 2.29). Nun können daraus bestimmte Samples aus weiteren Untergruppen gespielt und verändert werden.

Zu jeder Aktion werden auf der Sensorfläche und um die Pucks herum Informationen und Resultate über den Projektor begleitend eingeblendet. So weiß der Benutzer sofort, welche Einstellungen er vornehmen kann und zu welchen Resultaten sie führen.

Puck	Form	Bedeutung
Audio Track		physische Repräsentation von digitalen Audio Samples/Tracks, z.B. Hats, Beats, Melody, etc.
Selektor		Auswahlmöglichkeiten durch „virtuelle Blase“, z.B. Perc, Lead, Texture, Glitch, Kick, etc.
Mikrophon		akustische Wiedergabe, Lautstärkenregelung

Tabelle 2.2: Die drei verschiedenen Pucks des Audiopads

Insgesamt können am derzeitigen Audiopad bis zu neun Pucks gleichzeitig erkannt und in Aktion treten. Neben den Pucks, die mit Samples Gruppen belegt werden können, gibt es noch den Selektor und das Mikrophon (siehe Tabelle 2.2).

Audio Track Puck: Diese Steine binden Samples Gruppen, die wiederum aus Untergruppen bestehen. Dabei steht die jeweils aktuelle Samples Bezeichnung oberhalb des Pucks, links ist eine Balkenanzeige für die Lautstärke. Über der Bezeichnung findet sich noch ein Pluszeichen (+). Dieser Hotspot deutet an, dass hier optionale Einstellungen mit dem Selektor zugefügt werden können. Ein weiterer befindet sich unterhalb des Pucks und ist für Effekte zuständig, der erst bei laufendem Audio Sample sichtbar wird.

Im Audiopad werden bei einer Session zwischen fünf bis acht Audio Track Pucks eingesetzt. Durch einen Halbkreisbogen, der sich um den Puck dreht (ähnlich wie eine

rotierende Schallplatte), wird ein gerade spielender Audio Track visualisiert. Dieser dient auch der Lautstärkenanzeige, je dicker desto lauter der Track.

Selektor Puck: Der Auswahl Puck wird von einer „virtuellen Blase“ umgeben, die sich an Pucks bindet, wenn man ihn annähert. Dadurch ergeben sich weitere Einstellungsmöglichkeiten (z.B. Soundwahl, Effekte, etc.).

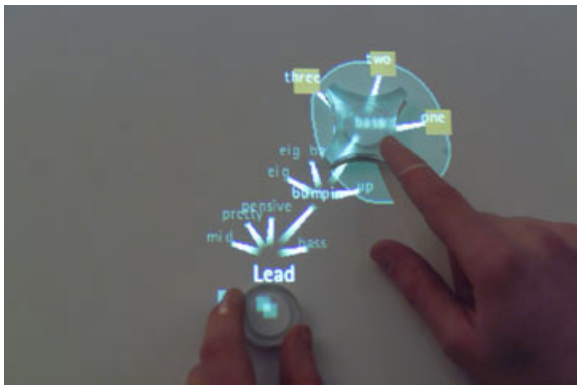


Abbildung 2.30: Audiopads baumstrukturiertes Menü zur Auswahl eines einzelnen Samples

Soundwahl / Selektor und Audio Track Puck (siehe Abbildung 2.30): Wenn der Selektor von oben in die Nähe eines Pucks (mit zugewiesener Samples Obergruppe) gebracht wird, bindet sich seine virtuelle Blase an den Puck. Ein Pfeil, der nun auf den Puck (bzw. auf sein Pluszeichen) zeigt, deutet an, dass hier eine Option wählbar ist. Intuitiv führt man den Selektor noch näher und ein baumstrukturiertes Menü öffnet sich. Von der Wurzel (ehemals +) ausgehend lassen sich nun Kanten mit dem Selektor wählen, die zu Knoten und weiteren Kanten führen. Jede Richtung steht für ein anderes mögliches Audio Sample. Dabei sind ähnlich klingende Samples im Baum nahezu einander gruppiert. Der letzte Knoten bezeichnet das Audio Sample, das der Puck repräsentieren soll. Der Vorgang kann jederzeit abgebrochen werden, wenn der Selektor vor der endgültigen Auswahl von dem Baum weg bewegt wird. Das (neu) gewählte Sample wird über dem Puck visuell sichtbar.

Wenn nun erneut ein Audio Sample mit dem Selektor ausgesucht werden soll, so öffnet sich in der Baumstruktur die zuletzt gewählte Kante, um aus dieser Gruppe zu wählen, falls ein ähnlich klingendes Sample gewünscht ist.

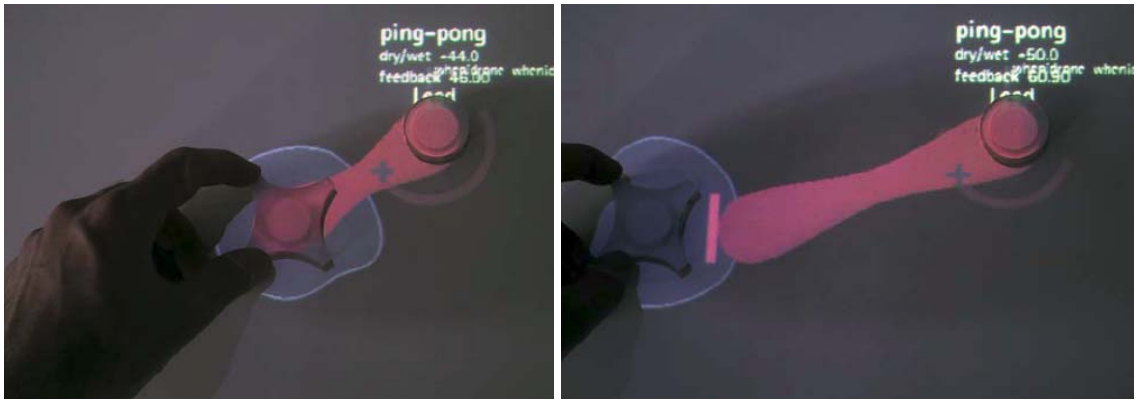


Abbildung 2.31: zweidimensionale Effektauswahl nutzt relative Positionierung

Effektwahl / Selektor und Audio Track Puck (siehe Abbildung 2.31): Wenn man den Selektor von unten an einen Puck (der gerade ein Audio Sample abspielt) heranführt, so zeigt der Pfeil auf einen weiteren Hotspot (+), der für die Effekteinstellung zuständig ist. Durch Translation eines oder beider Pucks verändert sich ihre relative Position zueinander, dadurch modifiziert der Benutzer die Effekt Parameter. Visualisiert wird alles durch eine farblich markierte Verbindung zwischen Selektor und Audio Track Puck, zusätzlich werden die Zahlenwerte auf der Sensorfläche angezeigt. Die Einstellungen können abgebrochen werden, wenn der Selektor vom Tisch aufgehoben wird.

Abbildung 2.32: Bedienung des gleitenden Menüs für verwandte Audio Samples

Menüauswahl von verwandten Sounds / Audio Track Puck (siehe Abbildung 2.32 und Abbildung 2.33): Um rasch zwischen verwandten Audio Samples zu wechseln, besitzen Audio Track Pucks das bogenförmige gleitende Menü, engl. *Floating Menu*. Die Auswahl erfolgt in der *Selection Area*, welche fünf zugehörige Audio Samples enthält. Der Benutzer führt hierzu den Puck einfach auf den Menüpunkt, den er bevorzugt, und schon wird die Alternative akzeptiert. Verlässt der Puck die Selection Area, ordnet sich das Floating Menu neu um den Puck an, der nun wieder frei bewegbar (in der *Movement Area*) ist.

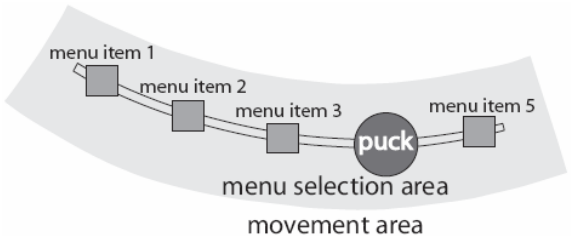


Abbildung 2.33: Audiopads gleitendes Menü

Mikrophon Puck / Lautstärkenregelung (siehe Abbildung 2.34): Je näher ein Puck an das Mikrophon gelangt, desto dicker wird der Halbkreisbogen um ihn, desto lauter wird sein Track gespielt und umgekehrt. Dabei ist es irrelevant, ob man ein oder mehrere Audio Tracks oder das Mikrophon bewegt.

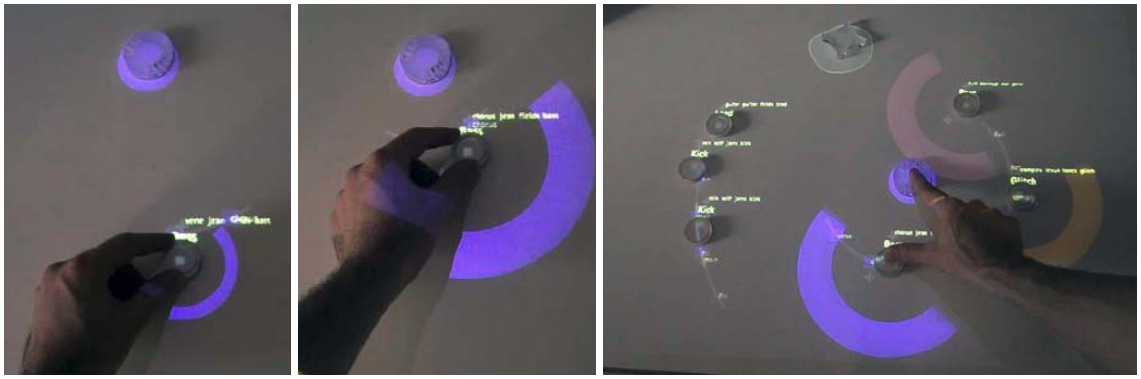


Abbildung 2.34: Lautstärkenregelung (Mikrophon und Audio Track)

2.3 Schlussfolgerungen

Der reacTable hat bereits in vielen Einsätzen gezeigt, dass sein zugrunde liegendes reacTIVision Framework die idealen Voraussetzungen und Möglichkeiten für optisches Tracking mitbringt. Vor allem die hohe Robustheit und die niedrige Fehleranfälligkeit bei der Objekterkennung und -verfolgung sprechen für ein ausgereiftes System, welches auch im 4T Projekt seine Einsatzberechtigung findet. reacTIVision ist jedoch erst seit kurzem für den öffentlichen Gebrauch freigegeben und konnte daher zu Beginn der 4T Entwicklung noch nicht berücksichtigt werden. Im weiteren Verlauf dieser Arbeit werden die Erkenntnisse der nachträglichen reacTIVision Integration in das 4T Framework ausführlich dokumentiert.

Auch der Music Table ist ein gutes Beispiel für die vielschichtigen Anwendungsmöglichkeiten von TUIs, insbesondere Tangible Tabletops. Durch kindgerechte Usability und Design werden erforschungsorientierte Lernmethoden offeriert, die konventionelle Herangehensweisen außer Acht lassen. Dadurch werden Eigeninitiative und Interaktion der Kinder verstärkt angeregt.

Das Audiopad führt vor, wie man auch mit wenigen Steuerungsobjekten ein TUI entwickeln kann, welches über mächtige Interaktionstechniken verfügt. Die Entwicklung und Evaluierung solcher Techniken spielt eine große Rolle in einem Designprozess. Dabei unterstützt hierbei vor allem ein elektromagnetisches Tracking System die technische Realisierung, da man Fehlerquellen optischer Trackingverfahren (z.B. nicht ausreichende Lichtverhältnisse) vernachlässigen und durch mechanische Elemente (z.B. Druckknopfschalter) neue Funktionalitäten hinzufügen kann. Nur wenn es gelingt nachvollziehbare Bedienmechanismen zu schaffen, können Handlungsabläufe von Benutzern intuitiv erkannt und umgesetzt werden.

Das 4T Framework setzt sich zum Ziel, möglichst universell einsetzbar zu sein. Daher müssen sowohl optische als auch sensorgesteuerte Tracking Technologien berücksichtigt werden, wenn es darum geht, neue Tangible Tabletop Anwendungen zu entwickeln.

3 Physische Verkörperung und Manipulation

Das nun folgende Kapitel befasst sich mit neuen, innovativen Ideen der *Erkenntnistheorie* und des *Interaction Designs*. Es werden Methoden und Prinzipien vorgestellt, die den Einsatz von physisch greifbaren Elementen zur erweiterten Wissensvermittlung bzw. Bewusstseinsbildung forcieren, um das Verständnis für theoretische und praktische Informationen zu verbessern. Besonders der Einsatz von TUIs kann existierende Lehrmethoden in ihrer starren Komplexität für Schüler (vor allem für Kinder) als auch Lehrer auflockern und so den Unterricht neu beleben.

3.1 Einführung

Der Computer hat bereits seit einigen Jahren längst Einzug in Lehreinrichtungen wie Schule und Universität gehalten. Eigens entwickelte Lernprogramme repräsentieren anhand ihrer klassischen GUI den Unterrichtsstoff oder bieten Lösungshilfen zu fachspezifischen Problemstellungen an, die der Studierende im Zuge seiner schulischen bzw. akademischen Ausbildung erfolgreich bewältigen muss.

Das klassische Eingabeschema über Tastatur und Maus liefert für viele Unterrichts- bzw. Aufgabenfelder zufriedenstellende Ergebnisse. Allerdings werden oft komplexe Szenarios für Problemlösungen künstlich geschaffen, welche vor allem die haptischen Fähig- und Fertigkeiten des Menschen nicht genügend ausreizen oder ganz außer Acht lassen.

Genau dieser Schwachpunkt ist das entscheidende Qualitätsmerkmal von tangiblen Lösungen. TUIs sind ganz gezielt auf menschliche Interaktionen mit Objekten ihrer räumlichen Umgebung ausgerichtet. Physische Interaktion ist ein wichtiges Merkmal und Kriterium unserer Bildung, welches unterschiedliche Herangehensweisen zu Problemlösungen offeriert und zum „Nachdenken über die Welt“ anregt. Personen werden auch an einem TUI leichter zu *rechnergestützter Kooperation und Kommunikation* (CSCW[11]) ermuntert, während sie gemeinsam mit physischen Repräsentationen digitaler Informationen hantieren und in Gruppen zusammenarbeiten. Tangible Tabletops geben hier sehr gute Beispiele ab (z.B. reacTable, Music Table), die eindeutige Vorteile gegenüber einer entsprechenden GUI Umsetzung aufweisen (Xu, 2005).

3.2 Wissen vermitteln – Wissen verkörpern

Der menschliche Körper spielt eine entscheidende Rolle im Laufe des Lebens. Er hilft uns Erfahrungen zu sammeln, die Welt um uns zu verstehen und mit ihr zu interagieren. Die herkömmliche Kombination von Tastatur und Maus schränkt jedoch seine

[11] Computer-supported cooperative work (CSCW)

Handlungsmöglichkeiten ein, da ein GUI immer nur ein und dieselbe Herangehensweise für Computereingaben liefert, egal ob es sich um Textverarbeitungs-, Musik- oder Grafikprogramme handelt (Klemmer et al., 2006).

In der modernen Kognitionswissenschaft steht *Verkörperung* (engl. Embodiment) als Maß dafür, wie die Natur des menschlichen Geistes durch seine physische Repräsentation in der Welt geprägt ist. Erst unser Körper ermöglicht anhand seiner sensorischen und motorischen Fähigkeiten Ideen, Gedanken und Konzepte zu formen und umzusetzen.

Wahrnehmende und kognitive Strukturen, Prozesse und Operationen bestimmen, wie die Einzelheiten des menschlichen Körpers in komplexen physischen, sozialen und kulturellen Umgebungen agieren. Man spricht auch von *verkörperter Wahrnehmung* (engl. Embodied Cognition), welche primär den Menschen als aktiv handelnde Person in den Vordergrund stellt und in ihm vielmehr als bloß den passiven Computeranwender sieht. Sie betont damit den Zusammenhang zwischen dem menschlichen Körper, seiner Wahrnehmung, seinen erlebten Erfahrungen und seinen Fähigkeiten (Antle, 2009).

Kinder erlernen abstrakte Konzepte anhand von physischen Metaphern, die ursprünglich aus der natürlichen Umgebung des Menschen kommen. Durch wiederholte Verknüpfung des schwer vorstellbaren Begriffs mit seiner Verkörperung erkennen sie den Zusammenhang und verstehen seine Bedeutung. Im weiteren Verlauf helfen die Assoziationen komplexe Zusammenhänge abstrakter Konzepte zu erfassen und eigenständig umzusetzen.

Klemmer et al. (2006) führen in ihrer Arbeit zur Verbesserung von Interaction Design vor, warum Verkörperung für menschliche Erfahrungen und Handlungen von entscheidender Rolle ist. Dabei stützen sie ihre Theorien auf psychologische, soziologische und philosophische Aspekte unserer Entscheidungsfindung und unseres Handelns, welche in den folgenden fünf Unterkapiteln beschrieben sind. Sie sollen Entwickler zu neuen Wegen im Interaction Design inspirieren und damit zu einer harmonischen Vereinigung von physischer und digitaler Welt beitragen.

3.2.1 Denken durch Handeln

Gedanken und Aktionen sind eng miteinander verknüpft. Beide regen zu logischem Denken an und sind ein wichtiger Bestandteil des menschlichen Lernverhaltens. Erst durch wiederholtes Ausprobieren und Testen werden komplexe Strukturen erkannt und schlussendlich erfolgreich verarbeitet.

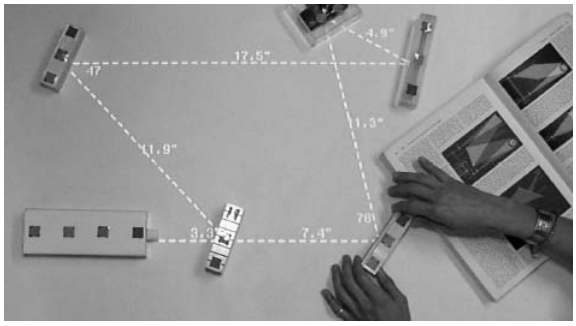


Abbildung 3.1: Illuminating Light lehrt optische Systeme

Die *Montessori Methode* (Montessori, 1964) unterstützt den Ansatz „Denken durch Handeln“ und propagiert, dass die körperliche Bindung bzw. Beschäftigung mit physischen Objekten aktives Lernen (im Volksschulalter) fördert. Diese Tatsache spielt aber auch im späteren Lebensverlauf weiterhin eine wesentliche Rolle, wie die *Illuminating Light* Werkbank des MIT veranschaulicht (Underkoffler & Ishii, 1998). Hier können Studenten optische Phänomene erforschen, indem sie tangible Steuerungsobjekte auf einer Tischfläche platzieren und bewegen (siehe Abbildung 3.1).

Die kognitive Entwicklung während der Kindheit ist ohne körperliche Interaktionen mit der Umwelt undenkbar. Denkprozesse laufen nicht nur separat im Gehirn ab, wie so oft behauptet wird. Im Gegenteil: Sie sind eng mit ihren physischen Aktivitäten verbunden. Diese Kombination physiologischer Vorgänge ist von essentieller Bedeutung, um die menschliche Wahrnehmung zu verstehen (Pecher & Zwaan, 2005).

Auch bei der Erstellung von Prototypen spielt die Verkörperung (von Ideen und Gedanken des Designers) eine besondere Relevanz. Ein physisches Modell kann durch neue Perspektiven überraschen, indem es unerwartete Aspekte der Realisierung offeriert, die der Entwickler ansonsten nicht erfahren hätte. Die sichtbare Repräsentation eines Problems legt relevante Zusammenhänge und aufkommende Schwierigkeiten frei, was wiederum zur Lösung des selbigen führt.

3.2.2 Leistungsfähigkeit

Der zentrale Gegenstand einer Handlung ist oft auch gleichzeitig eine Ausdehnung der Interaktionsmöglichkeiten. Die „Verlängerung des eigenen Arms“ kann sich auf unterschiedliche (und ungewöhnliche) Art und Weise manifestieren, z.B. durch Erhöhung des Aktionsradius, durch Verstärkung der Kraft oder durch eine wahrnehmbare Berührung der Hautoberfläche. Damit beeinflusst die Erweiterung nicht nur die zu vollziehende Tätigkeit an sich, sondern ist auch ein verstärkender Faktor der eigenen Fähigkeiten, die in Folge dessen trainiert und verbessert werden (McLuhan, 1994).

Die Hände sind unsere komplexen Tast- und Greifinstrumente, die durch die Anordnung ihrer Knochen, Muskeln und Nerven Manipulationen aller Art ermöglichen. Sie arbeiten zusammen und erlauben (nicht nur am Computer) asynchrone Benutzereingaben, um Aufgaben schnellstmöglich fertigzustellen.

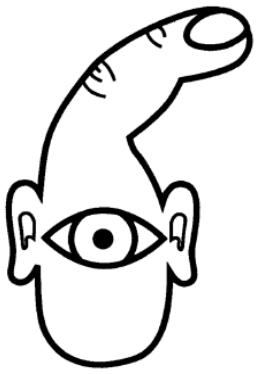


Abbildung 3.2: Das GUI Denkmodell eines Benutzers

Obwohl unsere Hände perfekte Voraussetzungen für den beidhändigen Einsatz mitbringen (z.B. Chirurgen, Musiker, Puppenspieler, etc.), werden sie vom klassischen GUI Eingabeschema dennoch vernachlässigt, wie Abbildung 3.2 in der Grafik bildlich veranschaulicht. Scheinbar jede auszuführende Benutzeraktion nutzt lediglich einen Bruchteil der vorhandenen Möglichkeiten (z.B. Fingerdruck). Die Reduzierung auf Maus und Tastatur ist insofern zu überdenken, da sie die genannten Aspekte und Eigenschaften vorhandener haptischer Fähigkeiten völlig außer Acht lässt.

Ein weiterer Grund, der gegen GUIs spricht, liegt in der menschlichen Fähigkeit, Bewegungsabläufe und Körperpositionen zu erkennen, zu speichern und abzurufen. Die *Kinästhetik* befasst sich mit der Erforschung dieser Bewegungsmuster, die in täglichen Abläufen völlig selbstverständlich ausgeführt werden. Das Phänomen der kinästhetischen Speicherung bekannter motorischer Handlungen ist bei einer gewöhnlichen Computerschnittstelle aber von Nachteil. Dieselbe Körperbewegung (z.B. Mausklick) wird für unterschiedliche Programmaufgaben herangezogen und kann deswegen nicht eindeutig einer bestimmten Tätigkeit zugewiesen werden, wie das z.B. beim Radfahren, beim Schwimmen oder beim Musizieren der Fall ist.

Die logische Konsequenz muss nun die vermehrte programmtechnische Umsetzung von tangiblen Lösungsalternativen sein, bei der die Vorteile kinästhetischer Speicherung und beidhändigen Arbeitens optimal ausgenutzt werden. Zwei Hände beschleunigen den Prozess und steigern die Leistung durch mehr Handlungs- bzw. Bewegungsfreiheit.

3.2.3 Sichtbarkeit

Die Sichtbarkeit einer Arbeit offenbart sich im Ergebnis bzw. im Produkt, welches daraus entsteht. Dabei spielt das Ausmaß, inwieweit die Aktivitäten einer Handlung Beteiligten und Beobachtern durch ihre Ausübung vor Augen geführt werden, eine entscheidende Rolle. Tätigkeiten, die sich erkennbar voneinander unterscheiden, unterstützen Kooperation und Koordination. Sie werden daher noch am Ehesten als unverfälschte Leistungen angesehen, da man ihre Abläufe beobachten kann, wie das z.B. beim Audiopad der Fall ist (siehe Abbildung 3.3 bzw. Kapitel 2.2.3).

Abbildung 3.3: Interaktion mit Audiopad – Blickwinkel eines passiven Beobachters

Naturgemäß profitieren Schüler von den Erfahrungen und dem Wissen ihrer Lehrer. Durch Beobachtung des Meisters bei seiner Arbeit und gemeinsamer Ausübung erlernen sie das gewünschte Handwerk, indem eindeutige Bewegungsmuster und Handlungsstränge der Tätigkeit imitiert und schließlich selbstständig perfektioniert werden (Lave & Wenger, 1991). Das zeitgemäße User Interface sollte ebenso (wie ein Lehrer) die Fähigkeit besitzen, die Benutzer zur aktiven Mitwirkung und Gestaltung anzuregen, indem es sichtbar einzigartige Interaktionsmöglichkeiten bietet.

Leider kann auch in diesem Fall kein GUI einen konstruktiven Ansatz zur technischen Umsetzung bieten. Sowohl bei versierten Computerexperten als auch bei Anfängern werden die Arbeitsschritte mit Tastatur und Maus von einer außenstehenden Person als identisch wahrgenommen, da die eigentliche Tätigkeit quasi unsichtbar und de facto nur am Monitor stattfindet.

3.2.4 Risikobereitschaft

Neben der Sichtbarkeit einer Handlung und seinem Ergebnis ist die bewusste Kenntnis von (kalkuliertem) Risiko ein positiver Impuls für engagiertes Arbeiten und ein hilfreicher Ansporn bei Entscheidungsfindungen. Die physische Manipulation eines Körpers ist im Gegensatz zu einer digitalen Bearbeitung permanent, sie kann nicht durch Backup oder Undo-Funktion wiederrufen werden. Diese Feststellung, dass jeder

Eingriff eine Reihe von (un)erwünschten Konsequenzen nach sich zieht, schärft die Konzentration und die Sinne des Benutzers während der Interaktion (Dreyfus, 2001).

Vertrauen, Hingabe, Verantwortung und Konzentration sind nur einige Faktoren sozialer bzw. individueller Aktivitäten, die durch Risikobereitschaft aufgebaut und verstärkt in den Fokus gerückt werden. In gemeinschaftlichen Arbeitsbeziehungen, die über eine große Distanz hinaus geführt werden, kann Risikobewusstsein zu mehr Respekt und Vertrauen unter allen Beteiligten führen (Olson & Olson, 2000). Nichtsdestotrotz sind Personen in Vieraugengesprächen jedoch leichter imstande, auftretende Probleme und Missverständnisse zu klären.

3.2.5 Digitalisierung versus Realität

Interaction Designer befinden sich auf einer stetigen Gratwanderung, wenn es sich um die digitale Neugestaltung von zuvor nichtdigitalen Zuständen der physischen Welt dreht, die uns täglich umgibt. Einerseits soll das Resultat ihrer Arbeit Funktionalitäten offerieren, die in der bisherigen Umsetzung noch nicht vorhanden waren. Andererseits sind bestehende und vor allem bewährte Eigenschaften bzw. wichtige physische Aspekte im Umgang mit dem ursprünglichen System aber weiterhin beizubehalten (Suchman, 1994).

Obwohl jedes (Re-)Design einer Technologie auf das menschliche Verständnis der realen Welt aufbaut, werden bedeutsame unsichtbare Gesichtspunkte der existierenden Arbeitspraxis ignoriert bzw. nicht erkannt. Das Ergebnis ist zwar technisch überlegen, hinkt aber dem Original als unvollkommenes Replikat entscheidend hinterher.

Technischer Fortschritt mit gleichzeitigem Beziehungsverlust zur realen Welt ist kein Gewinn und nicht wünschenswert. Vielmehr ist die Optimierung des Vorangegangenen, welche zu einer verbesserten Neuauflage führt, anzustreben.

Bevor man also neue Computerschnittstellen schafft, die nur eine künstliche Nachahmung der Realität sind, sollte man gleich auf die Realität zurückgreifen. Es ist nämlich bei Weitem schwieriger, die individuell hervortretenden Charakteristika eines existenten Arbeitsprozesses zu simulieren als ihn in die Wirklichkeit zu implementieren. Man spricht in diesem Zusammenhang auch von *Embodied Virtuality* (Weiser, 1991). Sie erzeugt im Gegensatz zur VR keine Scheinbilder, die die Bedeutung so mancher Arbeitspraxis negieren, sondern bettet wichtige physische Strukturen bzw. Gegebenheiten der Realität in die virtuelle Welt ein.

Zusammenfassend gesehen sind physische Verkörperungen bei der Gestaltung von innovativen Interaction Design Prozessen durchwegs zu begrüßen. Sie vereinen die positiven Eigenschaften der realen als auch der digitalen Welt. Dadurch kann der

Mensch seine haptischen Fähigkeiten voll und ganz zur Geltung bringen und gewohnten Arbeitsabläufen bzw. Praktiken nachgehen. Daher sollten Designer bei ihren zukünftigen technischen Umsetzungen sorgfältig überlegen, bevor sie vorteilhafte Aspekte unserer physischen Welt durch digitale Imitationen ersetzen und simulieren.

3.3 Child Computer Interaction

Designer von TUIs müssen sich im Klaren sein, dass die Entwicklung von Programmen für Kinder, im Gegensatz zu der von Erwachsenen, markante Unterschiede aufweist. Farben und Bewegungen sind z.B. entscheidende Designmerkmale. Computerschnittstellen müssen auffallen, um Beachtung und Aufmerksamkeit auf sich zu lenken (Berry et al., 2006).

Unter dem Begriff *Child Computer Interaction* (CCI) versteht man jenes HCI Teilgebiet, das seinen wissenschaftlichen Fokus besonders auf die Bedürfnisse von Kindern ausrichtet (Markopoulos et al., 2008).

3.3.1 Kinder und Technologie

Das uneingeschränkte, kindliche Interesse gilt Technologien, die vor allem Kontrolle ermöglichen, soziale Erfahrungen forcieren und ausdrucksstarke Werkzeuge zur kreativen Gestaltung bereit stellen. Kinder wollen ihre Neugierde befriedigen, indem sie wiederholt mit Gegenständen und räumlichen Strukturen experimentieren, um verschiedene Kombinationsmöglichkeiten zu erforschen und auszutesten. Gleichzeitig wollen sie aber nie die Übersicht über das Geschehen verlieren (Druin & Solomon, 1996). Im Zuge dessen fällt das Erlernen und das Verstehen von recht abstrakten Begriffen aus Mathematik (z.B. Division, Multiplikation), Physik (z.B. Schwerkraft), etc. nicht mehr schwer.

Dafür müssen allerdings auch die Parameter stimmen. Nur in einer Umgebung, welche Kindern die einfache Änderung und flexible Zusammensetzung räumlicher Gegebenheiten erlaubt, kann auch Wissen darüber erlangt werden. Allzu komplexe Anordnungen limitieren die Vielfalt an möglichen Kombinationen und somit auch den gewünschten Lerneffekt (Antle, 2009).

Abstrakte und formale Operationen erlernt ein heranwachsender Mensch ab einem Alter von 11 Jahren. Erst dann ähneln seine Denkprozesse langsam denen eines Erwachsenen. Im Entwicklungsstadium (7 bis 11 Jahre) davor benötigt ein Kind jedoch konkrete, körperliche Interaktion mit den Gegenständen seiner Umwelt, welche selbst stark an den jeweiligen formalisierten Gedanken gebunden sind (Wade, 1995).

Experten (Fröbel, Piaget, Montessori, etc.) sind sich einig, dass Kinder durch körperliche Interaktion mit speziellen physischen Objekten die Welt um sich herum

besser erforschen und kennen lernen können. Diese eigens entwickelten Artefakte werden in Fachkreisen als sogenannte *Manipulative* bezeichnet (Montessori, 1949). Dabei kann es sich z.B. um Holzbausteine handeln, aber auch um tangible Steuerungselemente. Die Manipulativen beflügeln die Phantasie der Kinder.

Daneben benötigen Kinder nach der Ansicht von Papert einen Freiraum zur individuellen Erforschung ihrer Vorstellungskraft, die sogenannte *Mikrowelt*. Eine überschaubare „Welt in der Welt" mit interaktiven (Frei-)Räumen bzw. (Spiel-)Plätzen, die es zu erkunden gilt. Hier können sich Kinder zurückziehen und hineinversetzen, um ihre Gedanken zu formen, zu äußern und damit zu experimentieren (Papert, 1980). Ein tischbasiertes TUI könnte so einen Ort verkörpern.

3.3.2 TUIs für Kinder

Abbildung 3.4: Child Computer Interaction

CCI beschäftigt sich mit der Frage, wie Kinder mit Gegenständen unseres technisierten Alltags (z.B. Mobiltelefon, Computer, Spielkonsolen, etc.) zurechtkommen und in weiterer Folge interagieren (siehe Abbildung 3.4).

Die ersten Erfahrungen dahingehend werden bereits in immer jüngeren Jahren gemacht. Gerade deshalb ist es wichtig, dass vor allem kindgerechte Produkte entwickelt werden, die das Wohlergehen und die geistige Entwicklung junger Anwender berücksichtigen.

Daher gilt die besondere Aufmerksamkeit vor allem der Gestaltung einer Computerschnittstelle, die Kinder sowohl fasziniert als auch spielerisch zum Lernen anregt. TUIs können für den Lernerfolg hier von Vorteil sein, wie die folgenden Punkte unterstreichen:

- *Schnell erlernbar*: Aufgrund ihres intuitiven Designs sind TUIs natürliche Schnittstellen. Die wichtigsten Funktionen werden rasch wahrgenommen und erlernt, sodass sich Kinder gleich auf die wesentlichen Aufgaben konzentrieren können.

- *Alternative Eingabemöglichkeit*: Die große Auswahl an möglichen Interaktionen erlaubt unterschiedliche Problemlösungen durch die Kombination physischer Objekte und das Setzen notwendiger Maßnahmen. Der Benutzer nimmt die Manipulationen nicht über ein GUI vor, sondern greift tatsächlich ins Geschehen ein. Die körperliche Auseinandersetzung der Kinder mit dem TUI verstärkt ihre Kontrolle über den Computer. Das wiederum führt dazu, dass sie nicht so schnell das Interesse verlieren und mit Enthusiasmus bei der Sache (bzw. Aufgabenstellung) bleiben.
- *Versuch und Irrtum*: Das Objekt von Interesse ist auf einem TUI stets präsent. Getätigte Aktionen (z.B. Translation, Rotation, etc.) können wiederrufen werden, falls sie kein erwünschtes Resultat liefern.
- *Kooperation*: Der große Vorteil eines TUIs ist die Tatsache, dass Applikationen nicht mehr auf einen einzelnen Benutzer beschränkt sind. Kinder können mit ihren Freunden gemeinsam die Steuerungsobjekte bedienen und zusammen Lösungen zu Problemstellungen ausarbeiten. Der soziale Umgang mit Gleichaltrigen in der Gruppe fördert auch die Produktivität, Aufgaben werden rascher und leichter gemeistert.

Das sind nur einige positive Aspekte, die für den Einsatz von TUIs in den verschiedenen Entwicklungsstadien von Kindern sprechen. Der entscheidende Gesichtspunkt dafür ist jedoch, dass Verkörperungen jenes geistige Ebenbild erzeugen, welches später auch ohne seine physische Repräsentation erkannt und im richtigen Kontext verwendet wird. Was zunächst noch mit Kinderhänden angegriffen werden muss, kann in Zukunft gedanklich reproduziert und im entscheidenden Augenblick abgerufen werden (Antle, 2009).

3.4 Schlussfolgerungen

Die simple Bedienung von intuitiven TUI Systemen beruht auf dem Prinzip der verkörperten Wahrnehmung und ermöglicht die vollkommene Nutzung unserer haptischen Begabungen. Das und die Tatsache, dass physische Verkörperungen und die Verwendung der realen Welt als Schnittstelle zu weit weniger komplexen Anforderungen an die Entwickler führen (keine Nachahmung erprobter Gegebenheiten notwendig), lässt sich auch auf 4T anwenden. Das Framework erlaubt rasche und flexible Entwicklung von tischbasierten TUI Applikationen.

Bestärkt durch diese Erkenntnisse ist 4T die optimale Chance, die gewonnenen Einsichten und Erfahrungswerte in Form von innovativen (als auch kindgerechten) Computerschnittstellen umzusetzen.

4 Tangible Table ToolkiT (4T)

Abbildung 4.1: Das 4T Logo

Das nachstehende Kapitel verdeutlicht die ursprüngliche Zusammensetzung (bzw. die Software- und Hardwarekomponenten) des *Tangible Table ToolkiT (4T)*, wie sie vor der vollständigen Integration von reacTIVision bereits bestanden hat. Der Entwicklungsprozess des Projekts wird von der konzeptionellen Planung bis zu seiner praktischen Umsetzung im Detail erläutert.

Des Weiteren dient dieser Abschnitt sowohl als Anleitung und Hilfestellung für Entwurf, Design und Implementierung des 4T und zukünftiger Erweiterungen seiner Architektur als auch der Entwicklung und Verwendung neuer Applikationen, da alle funktionalen und nicht-funktionalen Anforderungen an das Framework ausführlich erklärt werden.

4.1 Projektbeschreibung

Das 4T Projekt ist ressourcensparend für die Verwendung auf einem herkömmlichen PC entwickelt worden und besteht aus einer interaktiven Arbeitsfläche (für elektromagnetisches bzw. optisches Tracking) einschließlich mehrerer tangibler Steuerungselemente (Pucks). Eine Middleware ermöglicht die Kommunikation zwischen PC und Ausgabeeinheit (Tisch). Die Visualisierung erfolgt durch einen Projektor von unten auf die Tischfläche, daher auch die Bezeichnung „4T Rückprojektionstisch“ (siehe Abbildung 4.2).

Weder auf nationalem noch internationalem Markt gibt es im Moment ein ähnliches Setup, um Object Tracking Applikationen im überschaubaren und kostengünstigen Rahmen innovativ und flexibel umzusetzen.

4.1.1 Konzept

Momentan gibt es eine Vielzahl an Ideen und Designvorschlägen, die sich mit der Realisierung von TUI bzw. tangiblen Applikationen in allen möglichen Formen beschäftigen (Paradiso et al., 2000). Doch keine technische Umsetzung ist mit der anderen kompatibel, die Software ist untereinander nicht auswechsel- bzw. erweiterbar. Immerhin gibt es ansatzweise Versuche einer Standardisierung auf diesem Gebiet, allerdings konnte sich bislang keiner entscheidend durchsetzen.

4T dient nun Designern und Entwicklern als Grundlage für schnelle und flexible Gestaltung von zukünftigen TUI Applikationen. Dahingehend vereint es bereits vorhandene optische und elektromagnetische Tracking Technologien zur Erfassung von Steuerungsobjekten zu einem einheitlichen modularen Framework. Auf diesem Weg eröffnet 4T nach eigener Prämisse neue Herangehensweisen und präsentiert innovative Lösungsansätze zu aktuellen Problemstellungen, die durch Tangible Computing abgedeckt und elegant gelöst werden können.

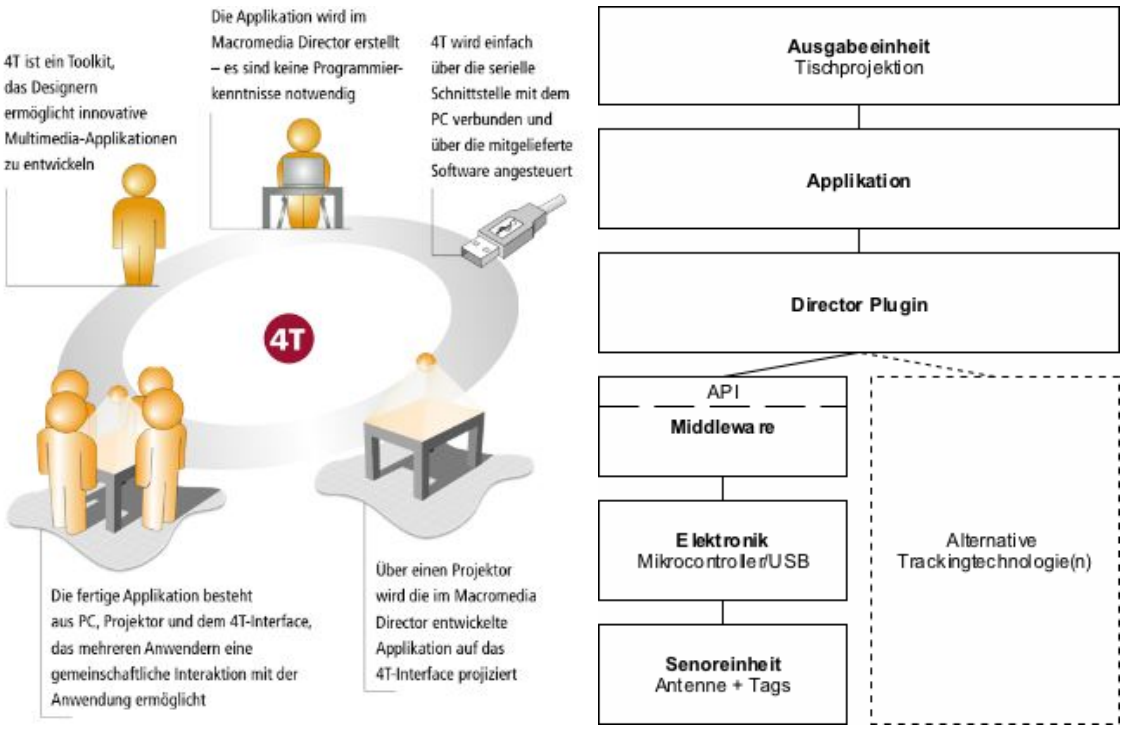


Abbildung 4.2: Arbeiten mit 4T

Ein weiteres projektrelevantes Ziel des INSO Entwicklerteams ist die Verwirklichung einer preiswerten Alternative zu herkömmlichen kommerziellen (tischbasierten) Produkten wie z.B. Microsoft Surface™[12]. Dabei handelt es sich um einen Tabletop Computer mit integrierter Multi-Touch Oberfläche (siehe Abbildung 4.3). Solche TUI Systeme sind in der Anschaffung und Lizensierung weit kostenintensiver als 4T. Das Framework setzt auf Open Source Software und sein Rückprojektionstisch besteht aus handelsüblichen Hardwarekomponenten.

[12] http://www.microsoft.com/surface/

Abbildung 4.3: Microsoft Surface™ – kommerzielles TUI

Das 4T Toolkit ermöglicht Designern und Entwicklern den simplen Einstieg in die Welt des Tangible Computing. Dahingehend unterstützt es seine Benutzer bei der Entwicklung von sowohl kommerziell als auch nicht-kommerziell genutzter Software. Man muss allerdings nicht vordergründig über die hard- und softwaretechnische Zusammensetzung Bescheid wissen (z.B. keine signifikanten Programmierkenntnisse) und kann sich ausschließlich auf sein kreatives Schaffen konzentrieren. Die Middleware unterstützt den Benutzer dahingehend, indem sie neben der programmierorientierten Gestaltung (durch Java) auch Schnittstellen zu Multimedia-Anwendungen schafft, die vor allem ihren Fokus verstärkt auf den Authoring-Prozess richten.

Bei der Wahl der primären Benutzergruppe gilt das Hauptaugenmerk vornehmlich Personen aus dem IT-Bereich bzw. dessen Umfeld. Zwar wird es auch unter ihnen keinen oder kaum jemanden mit Erfahrung im Bereich tangibler Applikationen geben, weil zur Zeit relativ wenig solcher Systeme erhältlich bzw. im Einsatz sind. Doch vor allem Designer, Softwarearchitekten oder Entwickler verfügen über entsprechendes Basiswissen, um die technischen Hintergründe zu verstehen und zweckdienlich einzusetzen.

Als sekundäre Benutzergruppe sollen Menschen aus Wissenschaft und Forschung (z.B. Universität) durch das Produkt 4T angesprochen werden. Bereits jetzt führen Wissenschaftler verstärkt Untersuchungen und relevante Studien auf dem Gebiet tangibler Technologien durch, um daraus Rückschlüsse zur Verbesserung gemeinschaftlicher Arbeitsprozesse zu gewinnen (Billinghurst & Kato, 2002).

4.1.2 Planung und Umsetzung

Wenn Benutzer zum ersten Mal mit dem 4T Rückprojektionstisch (Abmessung LxBxH = 108 cm x 83 cm x 108 cm) in Kontakt treten, wird ihnen zunächst sicher die interaktive Arbeitsfläche (Abmessung LxB = 105 cm x 80 cm) auffallen. Sie besteht aus

Plexiglas und ist sandgestrahlt, um mögliche, störende Reflexionen beim optischen Tracking zu vermeiden (siehe Abbildung 4.4).

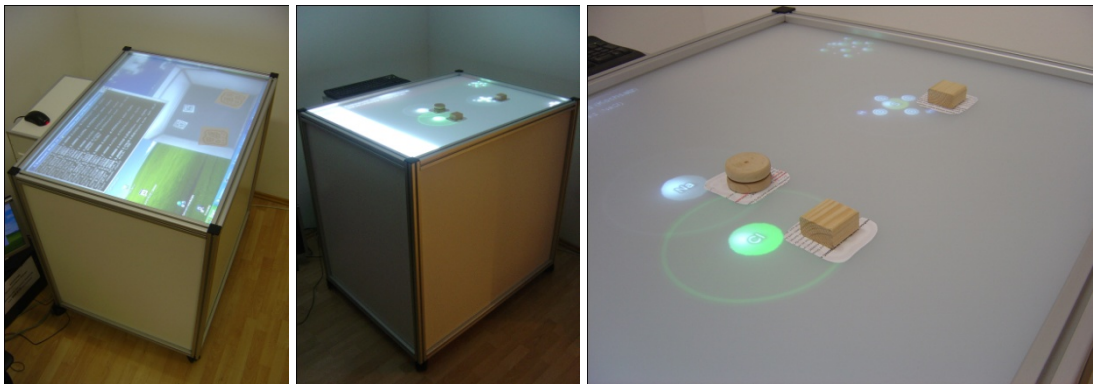


Abbildung 4.4: 4T Rückprojektionstisch

Auf dem 4T Rückprojektionstisch befinden sich die tangiblen Steuerungselemente bzw. Pucks. Lediglich diese Objekte sind zur Eingabe am TUI erforderlich, die direkt über die Oberfläche stattfindet. Daneben dient sie auch der visuellen Darstellung von Rückmeldungen des Systems. Somit erfolgen Steuerung und Repräsentation über ein und dieselbe interaktive Schnittstelle gemäß dem bereits vorgestellten MCRpd Modell (siehe Kapitel 2.1.1).

In einem ersten Designentwurf entstand auch die Idee, dass die optische Ausgabe nicht nur über die interaktive Oberfläche des 4T Rückprojektionstisches erfolgen möge. Mittels Spiegelungen sollten Projektionen auf Wänden, Zimmerdecken oder Glastüren realisiert werden. Damit hätte sich das Anwendungsgebiet auf Türschließsysteme, Codeabfragen und ähnliches erweitern lassen. Diese Idee ist zwar momentan nicht umgesetzt, allerdings ließe sich das 4T Tisch-Interface durch ein paar schnelle Handgriffe durchaus dahingehend erweitern.

Die Visualisierung geschieht aktuell über einen *NEC MultiSync LT245 DLP*[13] *Projektor*[14], der mit weiteren Hardwarekomponenten (IR-Strahler, Lüfter) und einer *Unibrain Fire-i™ Digital Board Kamera*[15] unterhalb der Arbeitsfläche angebracht ist (siehe Abbildung 4.5). Für die Kamera ist neben dem 4,3 mm Standardobjektiv optional auch ein 2,1 mm Weitwinkelobjektiv vorgesehen, beide ohne IR-Sperrfilter. Nur der Computer, auf dem das 4T Framework und weitere Software installiert sind, befindet sich neben dem Rückprojektionstisch.

[13] Digital Light Processing (DLP)
[14] http://www.nec-display.com/ap/en_projector_old/lt/index265.html
[15] http://www.unibrain.com/Products/VisionImg/Fire_i_BC.htm

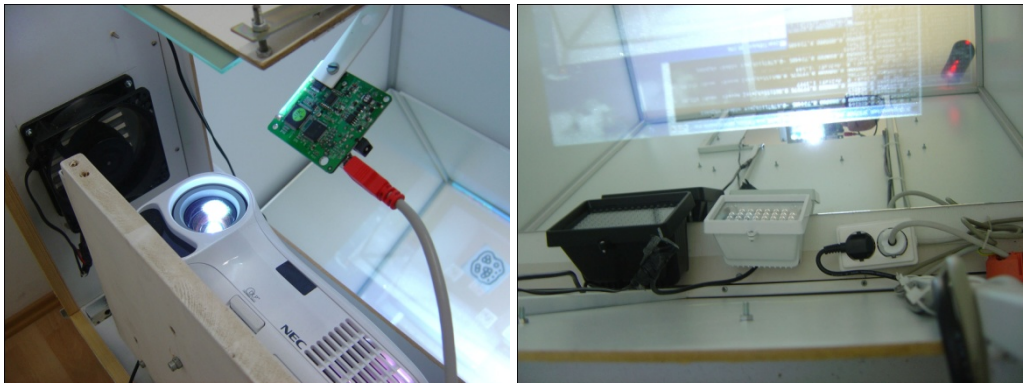


Abbildung 4.5: links – DLP Projektor und Digitalkamera, rechts – IR-Strahler und Spiegel

Der Benutzer sieht jederzeit die Auswirkungen seines Handelns. Die Geschehnisse werden in Echtzeit auf die Tischfläche projiziert. Gemäß dem TUI Konzept entsprechen alle physischen Steuerungselemente ihren digitalen Repräsentationen.

Die Pucks werden anhand optischer Markierungen wahrgenommen. Ursprünglich war diese Variante nur als Alternative zur elektromagnetischen Umsetzung mittels RF-Verfahren gedacht. Da aufgrund diverser Implementierungsschwierigkeiten jedoch keine akzeptablen Ergebnisse zustande kamen, wurde die Methode vom Team verworfen und vollständig durch optisches Tracking ersetzt.

Weil zu jener Zeit allerdings nur das ARToolKit als Open Source Software verfügbar war, griff das 4T Entwickler Team zunächst darauf zurück. Mittlerweile ist nun auch reacTIVision implementiert und der Benutzer kann zwischen den beiden Verfahren wählen. Eine zukünftige Kombination aus RFID und optischem Tracking ist durchaus vorstellbar und würde das gesamte 4T Framework optimieren sowie neue technische Umsetzungsmöglichkeiten eröffnen.

Um 4T auch ohne Hardware Unterstützung ausführen zu können, ist eine Simulator Applikation ähnlich der des reacTables integriert. Mit ihrer Hilfe sollen Debugging Vorgänge und Testumgebungen analysiert werden. Der Simulator arbeitet dabei mit dem Konfigurator zusammen, ein Tool, das die Anzahl der Steuerungselemente und das Setup festlegt. Das Framework ist in C++, die beiden GUI Werkzeuge sind in Java programmiert.

Das 4T Projekt erfüllt somit alle Qualitätsanforderungen, die für den reibungslosen Betrieb für Designer und Entwickler zwingend notwendig sind:

- *Anwenderfreundlichkeit*: Wie bereits erwähnt sind keinerlei Programmier-kenntnisse oder zeitintensive Einschulungen erforderlich, um mit 4T zu arbeiten. Allerdings wird ein gewisses Maß an Fachwissen im Bereich Entwicklung und

in der Handhabung mit dem Authoring-Werkzeug *Adobe® Director®*[16] vorausgesetzt.

- *Korrektheit*: Programminterne Berechnungen liefern immer exakte Ergebnisse. Positionsbestimmung und Objekterfassung können ohne korrekte Arbeitsschritte in der Datenverarbeitung sonst keine relevanten Auswertungen liefern.
- *Robustheit*: Falsche Benutzereingaben werden von der Software hinreichend erkannt und abgearbeitet. Sie stellen daher kein Stabilitätsrisiko dar. Des Weiteren schließen Benutzerhilfestellungen eventuelle Anwendungsfehler bereits im Vorfeld weitestgehend aus.
- *Skalierbarkeit*: Um laufend am System Verbesserungen und neue Features installieren zu können, ist das Framework modular zusammengesetzt und erweiterbar. Der objektorientierte Aufbau stellt dies sicher.
- *Feedback*: Das System stellt jederzeit aktuelle Informationen und wichtige Daten für die Applikationsentwicklung rasch zur Verfügung, um dem Benutzer einen kurzen aber exakten Überblick zu verschaffen.

4.2 Bestandteile des 4T Frameworks

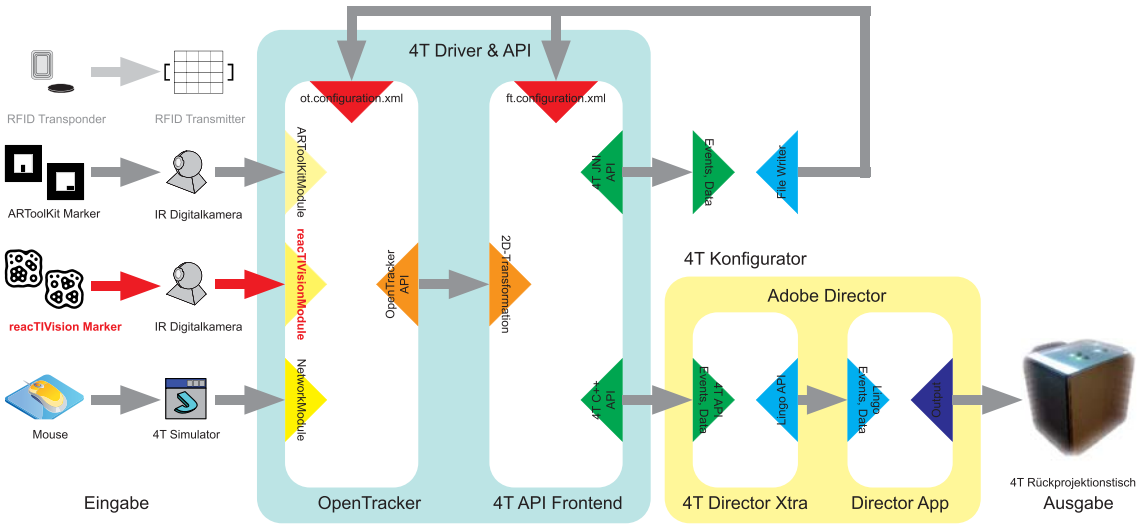


Abbildung 4.6: Überblick der (ursprünglich mit RFID geplanten) 4T Systemarchitektur

Das 4T Framework setzt sich aus vielen, unterschiedlichen Frameworks und Softwarebibliotheken zusammen. Das Grundgerüst, welches 4T dabei unterstützt, ist OpenTracker. Das Rahmensystem liefert die notwendigen Schnittstellen bzw. Module, die vor allem für optisches Tracking (z.B. ARToolKitModule, reacTIVisionModule), aber auch für Erfassungstechnologien aller Art, unentbehrlich sind. 4T bindet erst durch diese Module alle benötigten Erfassungsgeräte (Digitalkameras, Maus, etc.) in sein Framework ein (siehe Abbildung 4.6).

[16] Vormals Macromedia® Director®, mehr unter http://www.adobe.com/de/products/director/.

4.2.1 Object Tracking – Radio Frequency Identification (RFID)

Die Technik zur Erkennung von physisch etikettiert Gegenständen mittels elektromagnetischer Funkwellen nahm bereits in den 40er Jahren des 20. Jahrhunderts seinen Anfang und wird heute in vielen Bereichen des Alltags verwendet. Reisepässe, Mautkontrollsysteme oder logistische Warenabwicklungen sind nur einige wenige Anwendungsgebiete, wo die *Radio Frequency Identification (RFID)* bzw. sein RF-Verfahren zum Einsatz kommen (Finkenzeller, 2006).

Im Gegensatz zu optischen Trackingverfahren ist RFID nicht auf markante Referenzmarkierungen, Kamerasetups oder Lichtverhältnisse angewiesen. Neben der hohen Datendichte und möglichen Lesegeschwindigkeit besitzt ein solches System noch weitere Merkmale, welche das RF-Verfahren für den Gebrauch in Tracking Systemen besonders qualifizieren (Want et al., 1999):

- *Unaufdringlichkeit vs. Erkennung der physisch etikettierten Objekte*: Da die Transponder sehr klein und vor allem flexibel einsetzbar sind, wirken sie unauffällig und sind oft gar nicht für den Betrachter wahrnehmbar. Sie können nahezu unsichtbar an jedes beliebige Objekt angebracht werden, da sie sich nicht unbedingt an der Oberfläche befinden müssen, um von einem Transmitter erfasst zu werden. Somit erfüllt RFID die TUI Prämisse, dass sich tangible Objekte nicht künstlich in das Zentrum des Benutzerinteresses drängen sollen, obwohl sie in ihrer Bedeutung erweitert und ihnen digitale Informationen hinzugefügt wurden. Weil aber der Benutzer oft nicht auf Anhieb erkennt, welches Objekt nun mittels RF-Label ausgestattet ist und welches nicht, muss er aber auch zwangsläufig über Gegenstand, Funktion und Interaktionsmöglichkeiten informiert werden.
- *Post-Hoc Erweiterung vs. Funktionalität (assoziieren)*: Der Gebrauchszweck eines physisch etikettierten Gegenstandes ist für das RFID System völlig irrelevant. Die Labels sind entworfen worden, um nachträglich angebracht zu werden. Das bringt allerdings auch Nachteile mit sich. Der RFID Administrator bzw. Benutzer muss die Funktionen der betroffenen Objekte erst in einem Computersystem registrieren und die Daten anschließend warten. Erst dann können die erweiterten Artefakte Verwendung finden. Die Bindung des Gegenstandes an bestimmte Aufgaben und Eigenschaften bzw. dieser Vorgang sollte möglichst einfach gehalten werden (siehe Kapitel 2.2.3 – Track Manager).
- *Robustheit*: Es gibt keine Verschleißerscheinung. RF-Tags sind sowohl staub- als auch dreckundurchlässig und von daher bevorzugt genutzte Technologie bei widrigen Einsatzbedingungen (wie z.B. bei der Tieridentifikation in freier Wildbahn oder Viehzucht).
- *Kontaktloser Datenaustausch*: Transmitter und Transponder brauchen keine physische Berührung, um Daten auszutauschen. Ausrichtung und Position sind für RFID Objekte im Normalfall gegenstandslos, solange sie sich in der Reich-

weite der Leseeinheit befinden. Diese Tatsache trägt als zusätzliches Merkmal zur bereits oben erwähnten Unaufdringlichkeit bei.

- *Ästhetik*: Verborgene RF-Etikettierungen sind voll funktionstüchtig und beeinflussen in keinster Weise das Erscheinungsbild des erweiterten Gegenstands.

RFID Grundlagen – Transponder und Transmitter

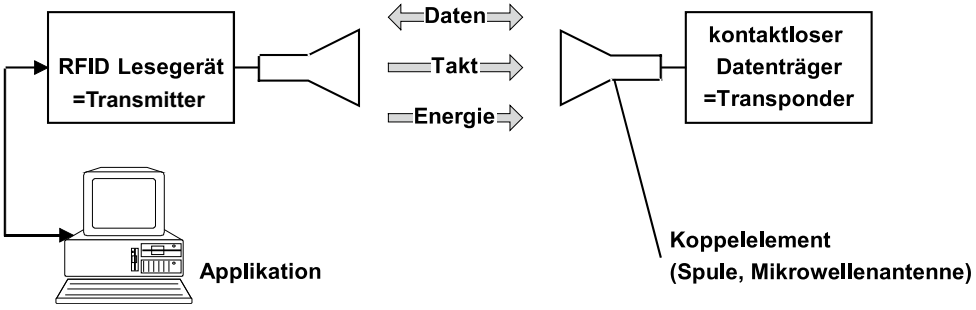


Abbildung 4.7: grundsätzliche Funktionen und Bestandteile eines RFID Systems

Ein RFID System nutzt Radiofrequenzen (RF), um Objekte anhand ihrer elektronischen Kennzeichnungen zu identifizieren. Es besteht aus zwei Komponenten (siehe Abbildung 4.7).

Der *Transponder* ist die RFID Sendeeinheit bzw. der kontaktlose Datenträger (RF-Label, -Etikett, -Tag, etc.). Er besteht je nach Bauart aus Mikrochip und Koppelelement (Spule oder Antenne) sowie Kondensator oder Batterie zur Energieversorgung. Der Datenträger wird an den zu identifizierenden Objekten angebracht und tritt erst aktiv in Erscheinung, wenn er in den Wirkungsbereich des Lesegerätes tritt.

Der *Transmitter* ist Erfassungs- bzw. Lesegerät für den Datenaustausch. Dabei kann er entweder Lese- oder Schreib/Lese-Einheit eines RFID Systems sein. Er setzt sich aus einem Hochfrequenzmodul, einer Kontrolleinheit und einem Koppelelement (Antenne) zusammen. Der Transmitter ist in den meisten Fällen via spezielle Schnittstelle auch mit einer computerunterstützten Infrastruktur (Applikation, Datenbank, Dienst, etc.) verbunden, wovon er seinen Energiebedarf bezieht und welche die Daten weiter verarbeitet.

Radiofrequenz-Verfahren

Das *Radiofrequenz (RF-)Verfahren* wird bei der elektronischen Artikelsicherung (EAS) eingesetzt und ist auch bei sensorgesteuerten Trackingsystemen, wie Audiopad (siehe Kapitel 2.2.3) oder Sensetable, implementiert. Der Datenträger ist hierbei ein L-C-Schwingkreis, der erst bei einer vordefinierten Resonanzfrequenz f_R reagiert. Bei aktuellen EAS Systemen verwendet man dazu eine Spule, die zwischen eine Folie geätzt ist, in Kombination mit einem entsprechenden Folienkondensator. Der L-C-Tag

wird dann als Etikett auf den zu sichernden Artikel geklebt und reagiert je nach Konfiguration, wenn er in den Nahfeldbereich des Transmitters gerät (Finkenzeller, 2006).

Bei tangiblen Objekten werden die ursprünglichen Induktivitäten benutzt, die jeweils aus gewickeltem, lackiertem Kupferdraht mit angelötetem Kondensator bestehen. Die Bauelemente sind üblicherweise noch durch ein Hartplastikgehäuse geschützt. In der Regel benötigen TUI Steuerungselemente keinen Mikrochip auf ihrem RFID Datenträger, da sie neben Lage und Position keine weiteren Informationen der TUI Applikation übergeben müssen.

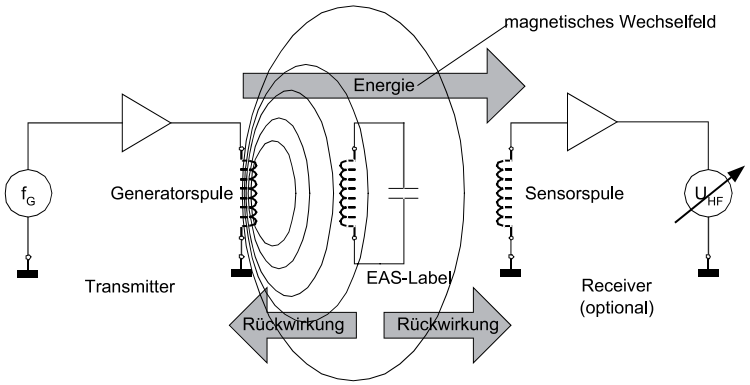


Abbildung 4.8: Prinzip des RF-Verfahrens bei EAS Systemen

Die Generatorspule (Antenne) des Transmitters erzeugt ein magnetisches Wechselfeld (mit der Generatorfrequenz f_G), welches im RF-Bereich nahe dem des Transponders bzw. L-C-Tags operiert. Kommen sich nun beide näher, wird Energie aus dem Wechselfeld in den Schwingkreis des Tags (EAS-Labels) nach dem Induktionsgesetz eingekoppelt, Strom wird induziert. Wenn die Generatorfrequenz f_G nun mit der Resonanzfrequenz f_R des Transponders übereinstimmt, kommt es zu einer Resonanzschwingung des L-C-Schwingkreises und er nimmt am Stromkreis teil. Diese *induktive Kopplung* zwischen Transponder und Transmitter gleicht der Kopplung von Primär- und Sekundärspule in einem Transformator, weshalb die Verbindung auch als *transformatorische Kopplung* bezeichnet wird. Sie bleibt intakt, solange sich die Antennenspule des Transponders im Nahfeld der Generatorspule des Transmitters befindet (siehe Abbildung 4.8).

Durch die induktive Kopplung werden in komplexeren RFID Systemen die Mikrochips passiver RF-Tags mit Energie versorgt. Da es sich beim RF-Verfahren bzw. im TUI Einsatz allerdings um einfach konstruierte L-C-Tags ohne Chip handelt, kommt dieser Effekt nicht zum Tragen. Sie müssen lediglich über ihre individuelle Resonanzfrequenz aufgespürt werden (Patten et al., 2006).

Der Strom im L-C-Schwingkreis wirkt allerdings dem magnetischen Wechselfeld entgegen. Dadurch kommt es an der Generatorspule des Transmitters zu einem leichten Anstieg des Spannungsabfalles, der wiederum zu einer messbaren Abschwächung der magnetischen Feldstärke führt.

Manche EAS Systeme verfügen noch über eine optionale Sensorspule bzw. Antenne, welche die induzierten Spannungsänderungen ebenfalls wahrnimmt. Die relative Stärke dieser Änderungen ist vom Abstand der Generator- zur Sensorspule sowie der Güte des angeregten L-C-Schwingkreises abhängig. Sie ist meistens sehr schwach und infolgedessen schwer erkennbar. Deshalb wird mit einem simplen, physikalischen Trick gearbeitet, um die L-C-Tags im RFID System eindeutig identifizieren zu können.

Die Generatorfrequenz f_G des magnetischen Wechselfeldes variiert zyklisch durch einen Wobbelgenerator (engl. Wobbler) zwischen zwei festgelegten Eckwerten. Immer wenn sie die Resonanzfrequenz f_R genau trifft, wird der L-C-Tag zur Schwingung angeregt und erzeugt die messbaren Spannungsfluktuationen an Generator- bzw. Sensorspule. Damit stellen auch Fehlertoleranzen in tangiblen Objekten (z.B. metallische Gehäuse) keine Hürde mehr bei der Erkennung ihrer Tags dar.

Transponder und Reichweiten

Beim RF-Verfahren kommen L-C-Tags zum Einsatz, die erst durch induktive Kopplung aktiviert und deshalb auch *passive Transponder* genannt werden. Sie verfügen über keine eigene Stromversorgung und beziehen den Bedarf über ihre Koppelelemente aus dem (elektro-)magnetischen Feld des Transmitters. Dabei wird im jeweiligen Transponder eine Spannung erzeugt, welche anschließend gleichgerichtet wird. Erst dann besitzt der (passive) L-C-Tag die notwendige Energie, um den Betrieb seines Mikrochips zu gewährleisten. Die Datenübertragung zwischen Tag und Lesegerät wird ebenfalls über den Energieaustausch vollzogen. Wenn sich der Transmitter außerhalb der Reichweite des Transponders befindet, reagiert er nicht. Die Passivität verhindert dann jede Art der Signalübermittlung. Passive Transponder eignen sich besonders gut als Steuerungselemente bei TUI Applikationen mit RF-Verfahren, da sie besonders robust und kostengünstig sind (Finkenzeller, 2006).

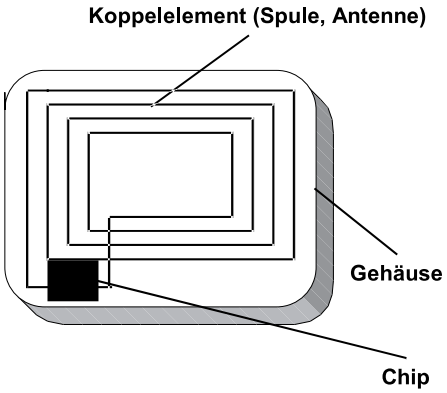


Abbildung 4.9: induktiv gekoppelter Transponder mit Antennenspule

Im Gegensatz dazu besitzen *aktive Transponder* eine Stützbatterie oder Solarzelle als Energiequelle, um ihren Chip teilweise oder komplett eigenständig mit Spannung zu versorgen und eine höhere Leistungsaufnahme auch bei größeren Entfernungen zu gewährleisten. Da nun der RF-Tag seine Energiebereitstellung selbst organisiert, kann er weit mehr Information speichern und das RFID System benötigt kein so starkes Magnetfeld wie mit passiven Transpondern. Trotzdem zweigen auch aktive Transponder aus dem magnetischen Wechselfeld des Transmitters weiterhin Energie für die Datenübertragung ab, da ihr Energiespeicher einzig und allein dem eigenen Mikrochip dient.

Die Batterie kann wahlweise in einen inaktiven „Power down"-Mode wechseln, wenn der Transponder das (elektro-)magnetische Feld bzw. den Wirkungsbereich des Transmitters verlässt. Wenn das Signal wieder stärker wird, wacht der RF-Tag aus dem stromsparenden Modus auf und läuft im Normalbetrieb weiter.

Die *Lesereichweite* eines Transponders muss für den jeweiligen Einsatzzweck entsprechend gewählt sein und hängt von mehreren Einflussfaktoren ab, um nur einige zu nennen:

- Orientierung/Positionierung des Transponders zum Transmitter
- Sendeleistung und Frequenzbereich des Transmitters
- Technischer Aufbau (Größe, Form, etc.) der Antenne des Transponders/Transmitters
- Lesegeschwindigkeit des Transmitters
- Art des physikalischen Kopplungsverfahrens (Induktive Kopplung, Backscatter-Verfahren)
- Umgebungsbedingungen (Störquellen)

Die Unterteilung der verwendeten Frequenzbereiche erfolgt länderspezifisch und lautet wie folgt:

- *Niederfrequenz (LF)-Bereich*: 100 bis 135 KHz
- *Hochfrequenz (HF)-Bereich*: 13,56 MHz
- *Ultra-Hochfrequenz (UHF)-Bereich*: 868 MHz (EU) und 915 MHz (USA)
- *Mikrowellen (MW)-Bereich*: 2,45 GHz und 5,8 GHz

Tischbasierte TUIs, welche die RFID Technologie nutzen, benötigen nur geringe Reichweiten zur Informationsübertragung. Die tangiblen Objekte treten erst bei direktem Kontakt mit der Tischfläche in Aktion und es werden ausschließlich 2D Informationen zur Lagebestimmung verarbeitet. Somit fallen tangible Tabletops in die RFID Kategorie der *Close-Coupling-Systeme*. Sie arbeiten mit induktiver Kopplung, Transponder und Transmitter operieren innerhalb eines Zentimeters bzw. bei direktem Kontakt miteinander (Reichweite ≤ 1 cm, Sendefrequenzen bis 30 MHz, LF-Bereich). Daneben gibt es noch zwei weitere Klassen.

Remote-Coupling-Systeme, welche ebenfalls induktive Kopplung einsetzen, können Entfernung von bis zu einem Meter bei der Datenübertragung überbrücken, was noch zum Nahfeldbereich gezählt wird (Reichweite < 1 m, zwischen LF- und HF-Bereich).

Alle weiteren, die deutlich darüber hinaus gehen und dem Fernfeld zugeordnet sind, fallen in die letzte Klasse der *Long-Range-Systeme* (Reichweite > 1 m, UHF-Bereich). Hier ersetzt das *Backscatter-Verfahren* (elektromagnetische Kopplung) die induktive Kopplung als Übertragungsart, weil mit weit höheren Frequenzen im Mikrowellenbereich gearbeitet werden muss, um die maximal möglichen Reichweiten von 15 m zu erzielen. Die Antenne des Transmitters erzeugt nunmehr eine elektromagnetische Welle, die sich weiträumig ausbreitet und in der Antenne des Transponders eine Wechselspannung generiert. Die Spannung wird abermals gleichgerichtet und dient passiven RF-Tags wiederum als Energiequelle ihres Chips. Aufgrund ihrer besseren technischen Voraussetzungen verwenden Long-Range-Systeme allerdings vermehrt aktive Transponder als Datenträger.

Vollduplex-, Halbduplex- und sequentielles Verfahren

Transponder, die über einen zusätzlichen elektronischen Mikrochip verfügen, können Informationen von bis zu einigen Kilobytes speichern. Die wesentlichen Bestandteile eines solchen Chips sind Prozessor, ROM (Betriebssystem), RAM (Arbeitsspeicher) und EEPROM[17] (für anwendungsrelevante Datenverarbeitung). Es gibt drei Verfahren, die bei der Datenübertragung zwischen Transponder und Transmitter (Lesen bzw. Schreiben) zum Einsatz kommen (Finkenzeller, 2006).

[17] Electrically Eraseable Programmable Read Only Memory (EEPROM)

Beim *Vollduplex (FDX)-Verfahren* erfolgt der Lese- und Schreibzugriff zeitgleich und kontinuierlich, d.h. wenn der Transmitter vom Transponder liest, überträgt er gleichzeitig Daten auf dessen Mikrochip. Diese Informationsübertragung erfolgt auf Teilfrequenzen des Transmitters, man spricht auch von *subharmonischen bzw. anharmonischen Frequenzen*.

Im Gegensatz dazu verläuft die gegenseitige Datenübertragung beim *Halbduplex (HDX)-Verfahren* zeitversetzt. Während der Transmitter den Transponder ausliest, schreibt er nicht auf dessen Mikrochip und umgekehrt. Die Übermittlung der Informationen erfolgt abwechselnd.

Sowohl bei FDX- als auch HDX-Verfahren erfolgt die Energieversorgung der RF-Tags permanent und unabhängig von der Art der Datenübertragung. Das Gegenstück dazu findet man beim *sequentiellen Verfahren (SEQ)*. Hier wird die Informationsübermittlung im Pulsbetrieb realisiert, d.h. nur zu bestimmten Zeitabfolgen wird vom Datenträger gelesen. Währenddessen ist die Energieversorgung des Transponders eingestellt. Sie wird erst beim Schreibvorgang des Transmitters wieder aufgenommen. Ähnlich dem HDX-Verfahren erfolgt die Datenübertragung also folgegebunden.

Da es sich bei FDX-Verfahren um Close-Coupling-Systeme mit Frequenzen unter 30 MHz handelt, wird dabei *Lastmodulation mit und ohne Hilfsträger* zur Übermittlung im Nahfeldbereich eingesetzt. Die Energieversorgung des Mikrochips stammt bekanntermaßen aus dem magnetischen Wechselfeld des Transmitters. Die Datenübertragung wird vom RF-Tag über das modulierte Ein- und Ausschalten seines Lastwiderstandes gesteuert. Außerdem reguliert der Widerstand den Energieverbrauch des Transmitters, was zur Veränderung des magnetischen Wechselfeldes und zum messbar steigenden Spannungsabfall an der Generatorspule führt, welche das Lesegerät wahrnimmt. Optional wird die Taktfrequenz eines zusätzlichen Lastwiderstandes im Transponder als Hilfsträger bezeichnet.

Daneben findet auch die *Methode des modulierten Rückstrahlquerschnitts* (vor allem beim Backscatter-Verfahren) zur Kommunikation im Fernfeldbereich ihre Anwendung. Sie beeinflusst wie die Lastmodulation das (elektro-)magnetische Feld des Transmitters, daher werden beide auch als *harmonische Verfahren* bezeichnet.

Der Rückstrahlquerschnitt ist ein Maß für die Wirksamkeit, mit der ein Objekt von geeigneter Materie (z.B. ein Transponder) elektromagnetische Wellen reflektiert. Er ist am größten, wenn der RF-Tag zur eintreffenden Wellenfront in Resonanz ist. Der Transponder verändert den Rückstrahlquerschnitt des Transmitters über einen Lastwiderstand. Das Lesegerät nimmt die Veränderungen am Magnetfeld auch in diesem Fall wahr. Es besitzt meistens jeweils zwei separate HF-Module zum Senden

und Empfangen. Die Antennen von Transponder und Transmitter besitzen Dipolcharakter. Im Gegensatz dazu sind Antennen im Nahfeldbereich spulenförmig.

Die Entwickler von 4T hatten für den Einsatz des RF-Verfahrens passive L-C-Tags vorgesehen. Da die Abmessungen des Rückprojektionstisches keine großen Entfernungen zwischen Transponder und Transmitter vorschreiben, wäre es eine kostengünstige Methode gewesen, um die Steuerungsobjekte zu erkennen und zu verfolgen. Andere Systeme, wie die bereits vorgestellte Augmented Knight's Castle, besitzen aktive Tags, die zusätzlich mit Mikrochips ausgestattet sind, um Daten an die Benutzer weiterzugeben.

4.2.2 Optical Tracking – ARToolKit

Das *ARToolKit* Framework[18] ist eine Softwarebibliothek für die Entwicklung von Applikationen in AR Umgebungen. Über ein dazugehöriges optisches Tracking System werden quadratische Referenzmarkierungen (meistens auf Karten angebracht) von einer Digitalkamera erfasst. Die Marker sind Platzhalter für virtuelle Objektüberlagerungen. Durch geeignete Ausgabegeräte (HMD, Handheld[19], Bildschirm) erscheinen die künstlichen Gebilde dem Betrachter als Bestandteile der realen Umgebung (siehe Abbildung 4.10).

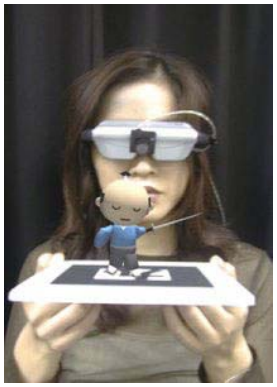


Abbildung 4.10: Shared Space – Virtuelle Figur auf einer Karte

Das Gemeinschaftsprojekt des Human Interface Technology Laboratory (HITLab)[20] der Washington University und ATR Media Integration & Communication in Kyoto kam ursprünglich bei dem interaktiven Kartenspiel *Shared Space* (Billinghurst et al., 2000), einer abgewandelten Version von Memory, als Tracking Technologie zum Einsatz.

[18] http://www.hitl.washington.edu/artoolkit/

[19] Ein Handheld (Handgerät) ist ein tragbares, handflächengroßes, elektronisches Ein- und Ausgabegerät. Beispielsweise fallen Smartphones und Blackberries bzw. PocketPCs unter diesen Sammelbegriff.

[20] http://www.hitl.washington.edu/home/

Mittlerweile ist es als erweiterte GPL[21] Softwarelizenz fester Bestandteil in unzähligen AR Applikationen.

Voraussetzungen

Um die Entwicklung und Nutzung von AR Systemen erfolgreich gestalten zu können, müssen einige Voraussetzungen erfüllt und diverse Einschränkungen berücksichtigt werden.

Wie bei allen optischen Trackingmethoden ist es notwendig, dass der zu erfassende Bereich zwischen Kamera und Referenzmarker frei ist. Einige optische Systeme (z.B. ARTag[22]) sind zwar in der Lage, kleinere Überdeckung durch spezielle Fehlerkorrekturmechanismen zu kompensieren, jedoch können auch hier noch Trackingprobleme auftreten. Um eine vollständige positive Erfassung sicherzustellen, sollte kein Störfaktor den sichtbaren Teil der AR Markierung beschneiden.

Das häufigste Problem tritt vor allem bei der physischen Manipulation durch den Benutzer auf, wenn dieser eine Karte aufnimmt und ihr Bild teilweise mit der Hand bzw. den Fingern verdeckt. Das virtuelle Objekt verschwindet dann bzw. wird nicht angezeigt. Daher muss eine dementsprechende Anweisung an alle Akteure vor der ersten Nutzung erfolgen.

Außerdem muss sich die Karte in identifizierbarer Reichweite befinden bzw. groß genug sein, um erkannt zu werden. Diese kann durch unterschiedliche Maßnahmen gewährleistet werden. Entweder vergrößert man die Referenzmarkierung oder man verringert die Entfernung zur Kamera. Dabei muss bei den Kartenabmessungen darauf geachtet werden, dass diese nicht zu groß gewählt sind, da sonst die intuitive und leichtgängige Handhabung durch den Benutzer auf der Strecke bleibt.

Ein weiterer Faktor bei der Erkennung von Referenzmarkierungen betrifft die schwarz-weißen Symbole selbst. Je einfacher ihre Struktur ist (große schwarze bzw. weiße Regionen), desto leichter werden sie auch vom Framework erkannt. Die Wahrnehmung durch die Kamera verschlechtert sich zunehmend, wenn die Marker zu weit gekippt werden. Dadurch wird die Markierungsmitte auf der Karte nicht mehr erfasst und das Tracking verläuft erfolglos.

[21] General Public License (GPL) ist eine spezielle GNU Lizenz für frei erhältliche Software. Mehr unter http://www.gnu.org/.
[22] http://www.artag.net/

Die Qualität der Videoaufnahme hängt sehr stark mit den daraus resultierenden Trackingerfolgen zusammen. Es sollte mindestens eine CCD[23] Digitalkamera mit USB- bzw. FireWire-Anschluss zum Einsatz kommen, welche über einen VFW oder WDM[24] Treiber verfügt. Die Anforderungen hierfür ähneln sehr stark denen beim reacTIVision Framework (siehe Kapitel 5.1). Weil das Tracking bei AR Systemen aber für gewöhnlich über eine direkte Verbindung zur Markierung erfolgt (bzw. aus Blickrichtung des Benutzers, da auch eine Kamera im HMD integriert sein kann), benötigt das Gerät keine Infrarot Unterstützung.

Damit die darauf folgende Bild- bzw. Videoerfassung einwandfrei abläuft, muss die eingesetzte Digitalkamera kalibriert werden. Dazu bietet das ARToolKit zwei verschiedene Möglichkeiten an, die sich nur anhand ihrer anschließenden Trackingpräzision unterscheiden: Die 2-Schritt Methode dauert länger, liefert aber genauere Ergebnisse und ist die empfehlenswertere. Es wird allerdings auch eine ungenauere 1-Schritt Methode angeboten, die etwas schneller konfiguriert ist.

Da das ARToolKit hauptsächlich für AR Konfigurationen entworfen wurde, wird als Ausgabegerät zur visuellen Darstellung der grafischen Bildüberlagerungen meistens ein HMD verwendet. Der technische Apparat unterstützt dabei vor allem im gemeinschaftlichen Einsatz (z.B. Videokonferenz) natürliche Kopfbewegungen und Verhalten (z.B. zustimmendes Nicken), wie sie auch bei Zusammentreffen in ausschließlich realen Umgebungen vorzufinden wären. Eine andere Darstellungsvariante wäre in dem speziellen Fall eher hinderlich. Nichtsdestotrotz können auch Bildschirme oder Handhelds zur virtuellen Repräsentation verwendet werden, wenn HMDs eine Benutzung des TUIs nicht entscheidend beeinflussen (siehe Music Table, Kapitel 2.2.2).

Natürlich muss auch für eine ausreichende Beleuchtung gesorgt sein. Ohne genügend Licht funktioniert die Erkennung und Verfolgung nur mäßig. Des Weiteren sollten die Referenzmarkierungen nicht auf reflektierendem Material gedruckt sein, da sich Spiegelungen ebenfalls negativ auf das optische Tracking auswirken. Da das ARToolKit keine rechenintensiven Operationen beinhaltet, wird auch nur ein handelsüblicher Arbeitsrechner vorausgesetzt.

Architektur des ARToolKit Frameworks

Zunächst werden die einzelnen Frames des von der Digitalkamera aufgezeichneten Videostreams nach den schwarzen Rahmen der Referenzmarkierungen durchsucht, die

[23] Charge-Coupled Device (CCD) ist ein lichtempfindlicher Bildsensor, der in Aufnahmegeräten eingesetzt wird und der digitalen Bilderfassung dient.

[24] Video for Windows (VFW) und Windows Driver Model (WDM) sind beides Microsoft Frameworks. VFW ermöglicht das Abspielen von digitalen Videoaufnahmen, WDM stellt Gerätetreiber für angeschlossene Hardware bereit.

genaue Identität des Markers ist zu diesem Zeitpunkt noch völlig irrelevant. Die erfassten Lagedaten werden als 4x4 Matrize geliefert und zur Verarbeitung an einen Rechner weitergeleitet. Falls ein Rahmen durch die Computersoftware gefunden wird, kann so ein spezieller Algorithmus die 3D Rotationen, Translationen und infolgedessen die relative Position der Kamera zur Karte im Raum berechnen.

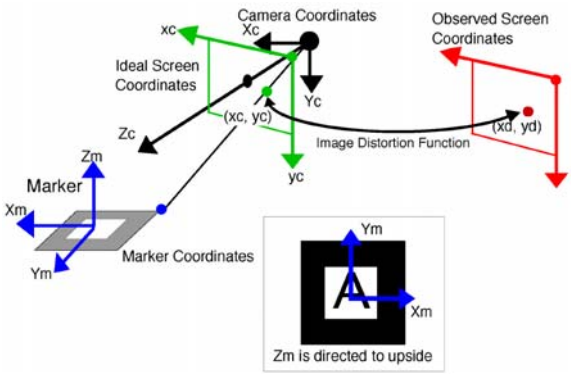


Abbildung 4.11: Die Koordinatensysteme im ARToolKit

Zur Bestimmung der Lagedaten verwendet das ARToolKit vier verschiedene Koordinatensysteme (siehe Abbildung 4.11):

- Marker-Koordinatensystem (blau)
- Kamera-Koordinatensystem (schwarz)
- Bildschirm-Koordinatensystem (grün) und
- Idealisiertes Bildschirm-Koordinatensystem (rot)

Aus dem Videostream werden die Werte des 2D *Bildschirm-Koordinatensystems* entnommen. Unter Zuhilfenahme der errechneten Parameter aus der Kamera Kalibrierung wird daraus das *idealisierte Bildschirm-Koordinatensystem* bestimmt. Als Nächstes werden von diesem ausgehend der 3D Zusammenhang zwischen *Kamera- und Marker-Koordinatensystem* hergestellt und berechnet. Das geschieht mittels iterativen Optimierungsprozess. Als Startbedingungen dienen dabei die vier Ecken der Referenzmarkierung sowie die Daten aus dem vorangegangenen Video Frame. Schlussendlich ergibt sich daraus, woher der Benutzer die Karte bzw. das entsprechende virtuelle Objekt betrachtet. Erst jetzt wird das Muster des Markers untersucht.

Nachdem die Kameraposition festgestellt ist, wird ein exaktes grafisches Computermodell an der Stelle generiert. Die Grafik überlagert den Marker. Dieser Vorgang wird erst für den Benutzer wahrnehmbar, wenn er das Szenario durch ein AR geeignetes Ausgabegerät betrachtet. Abbildung 4.12 fasst die verschiedenen Schritte noch einmal zusammen.

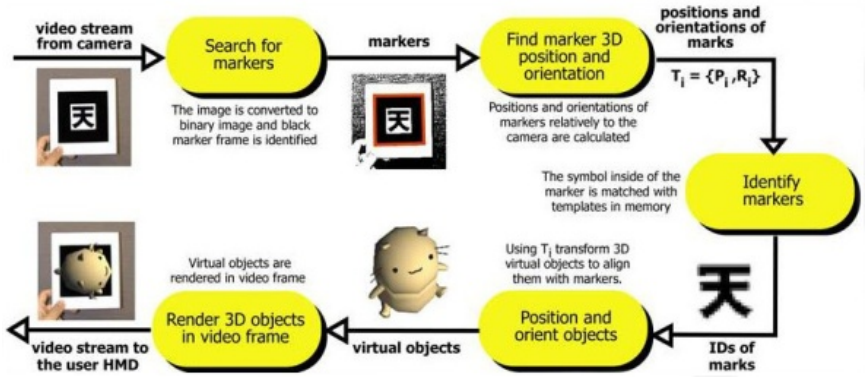


Abbildung 4.12: Funktionsweise des ARToolKit Frameworks

Das Framework ist für jedes gängige OS[25] (Linux, Mac OS X, Microsoft Windows) nur als Sourcecode (C, C++) verfügbar und muss dementsprechend kompiliert[26] werden. Hierfür sind zusätzliche Programme (in der jeweils aktuellen Version) notwendig. Als Entwicklungsumgebung wird Microsoft Visual Studio[27] unterstützt. Um die Kommunikation zwischen ARToolKit und Kameratreiber zu ermöglichen, kommt der DirectShow Wrapper DSVideoLib[28] zum Einsatz. Mittels OpenGL Utility Toolkit (GLUT)[29] lassen sich Fenster/Ereignisse verwalten (engl. Window/Event Handling). Des Weiteren werden DirectX[30] zum Rendern[31] und optional OpenVRML[32] benötigt.

Die plattformübergreifende Programmierung und die modulare Zusammensetzung des ARToolKits erleichtern die Integration von Komponenten. Die vier Module der Softwarebibliothek sind:

- *AR Modul*: die Hauptkomponente mit Tracking System, Kamerakalibrierung und Parametereinstellungen.
- *Video Modul*: beinhaltet eine Sammlung von Videobearbeitungswerkzeugen zur Erfassung der Video Frames, um nach Markern zu suchen.

---

[25] Operating System (OS), Betriebssystem.

[26] Da 4T zusammen mit ARToolKit momentan ausschließlich für den Gebrauch auf Rechnern mit Microsoft Windows XP ausgelegt ist, wird in diesem Dokument auch nur auf diese Kompilierung eingegangen. Für alle weiteren Betriebssysteme, siehe http://www.hitl.washington.edu/artoolkit/documentation/usersetup.htm.

[27] http://msdn.microsoft.com/de-de/vsts2008/products/bb933731.aspx

[28] http://sourceforge.net/projects/dsvideolib/

[29] http://www.opengl.org/resources/libraries/glut/

[30] DirectX ist eine Applikationsschnittstellen (API) Sammlung für multimediabezogene Tasks auf Windows Oberflächen. Mehr unter http://www.microsoft.com/directx.

[31] Rendering beschreibt in der Computergrafik die Bildsynthese, bei der aus virtuellen, räumlichen Modellen ein Bild erzeugt wird.

[32] OpenVRML ist eine Open-Source Software, die die Integration von Virtual Reality Model Language (VRML) Unterstützung in Programmen erlaubt. Mehr unter http://openvrml.org/.

- *Gsub Modul*: besteht aus mehreren Grafikroutinen, die auf OpenGL und GLUT Softwarebibliotheken basieren.
- *Gsub_lite Modul*: ersetzt das vorausgegangene Gsub Modul durch effizientere Grafikprogramme und operiert unabhängig von GLUT und anderen Window/Event Handling Systemen.

Abbildung 4.13 zeigt den Zusammenhang zwischen einer AR Applikation, dem Framework und seinen abhängigen Softwarebibliotheken.

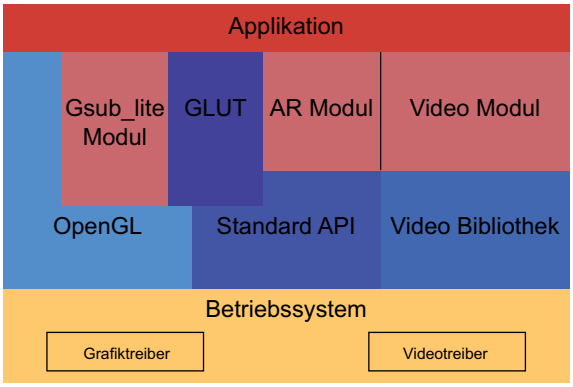


Abbildung 4.13: Architektur des ARToolKits (mit Gsub_lite Modul)

AR Marker und Kamera Kalibrierung

Für die visuellen Überlagerungen verwendet das System quadratische Referenzmarkierungen (siehe Abbildung 4.14). Anhand ihres schwarzen Randes sind sie für das Framework erkennbar. Darüber hinaus dient er zur Bestimmung der meisten Lagedaten. Das innere weiße Feld ist das eigentliche Muster (8 x 8 = 64 Felder) und dient der eindeutigen Identifizierung. Des Weiteren gibt es Aufschluss über die vier möglichen Ausrichtungen innerhalb des Rahmens. Ein simpler Algorithmus zur Mustererzeugung erlaubt eine Anzahl von maximal 4096 Kombinationen. Neben den vorgegebenen Mustern können auch eigene erstellt und verwendet werden.

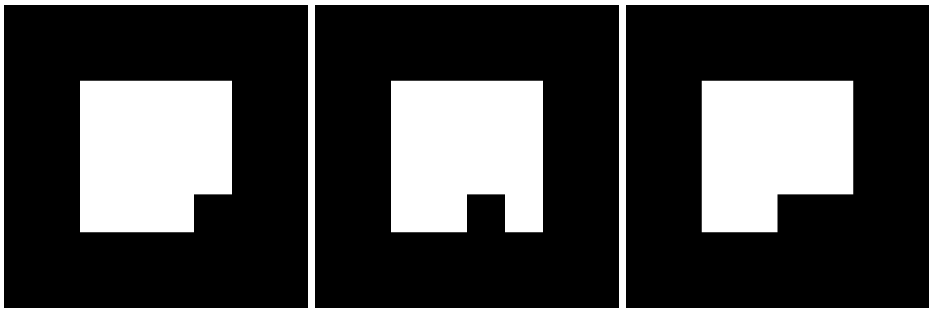


Abbildung 4.14: drei einfache Referenzmarkierungen des ARToolKits

In den meisten Fällen sind die Markierungen auf Karten angebracht, jedoch kann ohne weiteres auch jeder andere Gegenstand als Träger herangezogen werden. Über ein geeignetes Ausgabegerät kann der Benutzer nun die künstlichen Objekte von allen Seiten betrachten, er muss nur die Karte dazu entsprechend drehen und wenden.

Das ARToolKit ermöglicht auch das Entwerfen und Verwenden von eigenen Referenzmarkierungen. Dazu dient ein leerer Marker (nur schwarzer Rahmen, weißes Feld ist leer) als Grundlage. Er wird mit einem beliebigen Muster kombiniert, welches in das innere weiße Feld platziert wird. Der Marker wird anschließend von der Kamera in das System eingelesen, gespeichert und kann fortan eingesetzt werden.

Tangible AR und ARToolKitPlus

Das 4T Framework nutzte bis vor kurzem nur die Tracking Engine des ARToolKits, da reacTIVision zunächst keine Open-Source Applikation war, die verwendet werden hätte können. Der Einsatz AR basierter Technologie rechtfertigt sich aber durchaus auch aufgrund weiterer Besonderheiten, die dieses HCI Konzept beinhaltet. Sie wurden bislang zwar (noch) außer Acht gelassen, könnten aber ohne Weiteres für kommende 4T Applikationen herangezogen und implementiert werden.

Tangible Augmented Reality (TAR) kombiniert beispielsweise tangible Interaktionstechniken mit AR Darstellungsmethoden (Billinghurst & Kato, 2002). Bekanntermaßen arbeiten AR Systeme bei der Informationsdarstellung mit Überlagerungen realer Szenarien durch virtuelle Objekte. Das ermöglicht auf Kooperation beruhende Benutzergemeinschaften zu kreieren, die auf herkömmlichem Weg schwer bzw. nicht zu erzielen sind und dem realen Vorbild der Gruppenarbeit sehr nahe kommen.

AR unterstützt also die intuitive Interaktion durch den Benutzer mit der Computerschnittstelle und konzentriert sich dabei vor allem auf die gemeinschaftliche Nutzung. Ähnlich trägt dazu das TUI Konzept durch seine nahtlos gestalteten Beziehungen zwischen physischer Objektmanipulation und virtueller Datenrepräsentation bei. Die physischen Eigenschaften seiner Steuerungsobjekte (z.B. hart, weich, rund, eckig, etc.) spielen hierbei eine wesentliche Rolle, da sie es sind, welche den Grad des Benutzereinflusses bestimmen und so die möglichen Interaktionen vorgeben. Dadurch vereinfacht sich die Eingabe erheblich, obwohl sie den Systemanforderungen gewachsen ist.

Trotzdem ist es nach wie vor schwierig, aufgrund der Betrachtung eines physischen Artefakts auf seinen digitalen Informationsstand zu schließen. Es ist unmöglich diese dynamisch zu verändern, ohne dass das natürliche Benutzerverständnis zu tangiblen Handlungsabläufen verloren geht. TAR bindet nach dem TUI Prinzip virtuelle Objekte an virtuelle Informationen und ermöglicht permanente Variationsmöglichkeiten

während der Nutzung tangibler (und künstlicher) Gegenstände, sofern dies notwendig erscheint. Durch entsprechend visuelle Überlagerungen sind auch physische Objekte in der Lage, ihre Darstellungsmöglichkeiten und ihren Informationsgehalt zu erweitern.

Eine Weiterentwicklung stellt das *ARToolKitPlus* (Wagner & Schmalstieg, 2007) dar. Es wurde vor allem für den Einsatz auf Handhelds und Smartphones konzipiert. Aufgrund dieser Spezialisierung hat es nur mehr sehr wenig mit seinem Vorgänger zutun und ist daher auch nur für erfahrene Programmierer von Nutzen. Im 4T ist es für die weitere Entwicklung des Frameworks geplant, da es vielversprechende Leistungsverbesserungen in Punkto Funktionalität gegenüber der bisherigen ARToolKit Version bietet.

Aktuell wird am modularen *Studierstube Tracker* (Schmalstieg et al., 2000) gearbeitet, der momentan nur kommerziell erhältlich ist. Obwohl das Konzept vom ARToolKit sowie ARToolKitPlus stammt, wurde die Softwarebibliothek von Grund auf neu erstellt und kann sowohl auf leistungsstarken Rechnern als auch auf Handhelds verwendet werden.

4.2.3 Rahmensystem für Trackingdaten – OpenTracker

Für gewöhnlich werden Trackingdaten aus AR oder Virtual Reality (VR) Applikationen über unterschiedliche Computerschnittstellen gewonnen und in mehreren Zwischenschritten verarbeitet. Digitalkameras oder andere Aufnahmegeräte zur Bilderfassung dienen der Datenbeschaffung. Sie alle besitzen verschiedene Gerätetreiber und müssen auf die Anforderungen der Applikation abgestimmt und für den Transport über Netzwerkverbindungen korrekt adaptiert sein.

Man muss zwar verschiedene Setups für unterschiedliche Systemkonfigurationen in Kauf nehmen, aber individuelle Teilschritte, wie z.B. geometrische Transformationen oder Kalman-Filter[33], sind wiederkehrende Bestandteile vieler AR/VR Applikationen. Unter dem Leitspruch „*write once, track anywhere*“ stellt das nachfolgende skalierbare Rahmensystem eine allgemeine Lösung zu den vielen unterschiedlichen Aufgaben innerhalb eines Trackingsystems dar.

Das modular aufgebaute *OpenTracker* System der Studierstube[34] verfolgt die Implementierung von unterschiedlichen Trackingmethoden (wie z.B. ARToolKit) ohne Zwischenschritte und vereinfacht ihre Konfiguration und Datenverarbeitung mittels

[33] Der Kalman-Filter ermöglicht trotz fehlerhafter Daten (z.B. durch Messgeräte verursachte Störungen) Rückschlüsse auf den tatsächlichen Zustand eines Systems zu ziehen. Das ist nur möglich, wenn sowohl die mathematische Struktur des Systems als auch der fehlerbehafteten Messdaten bekannt ist.

[34] http://studierstube.icg.tu-graz.ac.at/opentracker/

XML[35] Unterstützung (Reitmayr & Schmalstieg, 2001). Das Ergebnis ist ein plattform- und schnittstellenübergreifendes Datenflussnetzwerk, welches die allgemeinen Voraussetzungen und Anforderungen zur Verarbeitung von Trackingdaten erfüllt und als kontrollierende architektonische Ebene zwischen der eigentlichen Applikation und den restlichen Softwarekomponenten (Gerätetreiber, Netzwerkverbindungen) agiert und vermittelt (siehe Abbildung 4.15).

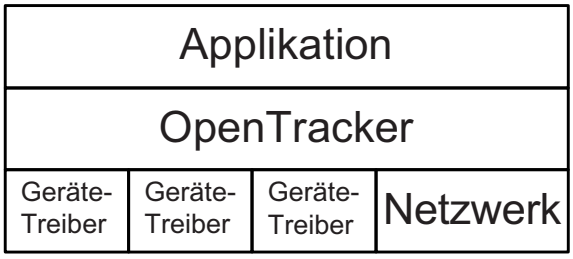


Abbildung 4.15: Die Beziehung von OpenTracker mit anderen Softwarekomponenten

Das Studierstube Projekt beruht dabei auf Beobachtungen, die anhand von AR bzw. VR Systemen im Umgang mit anderen Trackern und ihren Daten erzielt worden sind (Reitmayr & Schmalstieg, 2005):

- Die Abstraktion von sich wiederholenden Programmteilen (betreffend Trackingdaten) trennt auch gleichzeitig Applikationen von ihren Konfigurationseinstellungen und ihren angeschlossenen Geräten, was eine bessere Code Ausnutzung und Wiederverwendung gewährleistet.
- Die flexible Anordnung solcher Programmteile vereinfacht zudem Versuchsdurchführung und Entwicklung von AR und VR Applikationen.
- Eine eigens bereitgestellte Konfigurationssprache (XML) erlaubt überschaubare Entwicklungs- und Einstellungsmöglichkeiten vorzunehmen, auch wenn es sich um komplexe Setups handelt.
- Ein erweiterbares Softwaredesign ermöglicht einfache und rasche Implementierung von neuen Funktionen sowie den Austausch von Hardware.

OpenTracker wurde speziell für AR und VR Umgebungen konzipiert und ist in C++ implementiert. Alle Einstellungsmöglichkeiten sind durch XML Konfiguration möglichst flexibel und leicht veränderbar gehalten, ohne dass dabei das Originalprogramm bearbeitet werden muss. So definiert ein jedes Trackerobjekt verschiedene Schnittstellen, welche Abfragen über momentane Position und Orientierung sowie aktuelle Schaltflächenzustände (von Trackingapparaten) beantworten. In einem

[35] Extensible Markup Language (XML) ist ein simples und flexibles Auszeichnungssprachenkonzept in Textformat, welche den Datenaustausch zwischen Computersystemen ermöglicht. Mehr unter http://de.selfhtml.org/xml/index.htm.

Datenflussgraphen werden die Trackerobjekte anhand ihrer Beschreibung in einer Konfigurationsdatei zusammengefasst und in Knoten und Kanten strukturiert.

Verschiedene Computerschnittstellen werden so in der Open-Source Architektur miteinander vereint. Die Hard- und Softwareanforderungen an das Datenflussnetzwerk werden durch das modulare Softwaredesign der Bibliothek durch Verkapselung in die unmittelbare Konfiguration des OpenTracker Systems getragen. Dazu gehört sowohl die Eingabegeräte zur Datenerfassung als auch die anschließende Weiterverarbeitung der daraus gewonnenen Daten.

Voraussetzungen

Bevor OpenTracker erfolgreich zum Einsatz kommen kann, muss die Software kompiliert werden. Hierzu benötigt man noch zwei weitere Open-Source Softwarebibliotheken:

- Apache Xerces[36] ist eine XML Parser Bibliothek für C++ und Java. Sie ermöglicht es jeder Applikation XML Daten zu lesen und die Gültigkeit zu bestätigen. Die Implementierung der Knotengraphen und des ereignisbasierten Datenflusses basieren auf der DOM[37] Struktur von Xerces. OpenTracker verwendet die Software Bibliothek zum Parsen seiner XML Konfigurationsdatei.
- Adaptive Communication Environment (ACE)[38] implementiert als Wrapper Bibliothek viele essentielle Richtlinien und Muster für Software zur parallelen bzw. simultanen Kommunikation. ACE schafft plattformübergreifenden Zugang zu Systemressourcen wie z.B. Netzwerk Sockets, Event Demultiplexing, Signal Handling oder Nachrichten Routing.

Datenflussgraph sowie Knoten und Kanten

OpenTracker beschreibt und strukturiert mit Hilfe eines objektorientierten Datenflussgraphen die speziellen Konfigurationen und Transformationen erfasster Trackingdaten. Dabei werden Funktionalitäten des Frameworks durch *Knoten* (engl. Nodes) dargestellt, die durch *gerichtete Kanten* (engl. Edges) miteinander verbunden sind, welche wiederum Schnittstellen repräsentieren (siehe Abbildung 4.16). Der (gerichtete) Graph zeigt die Flussrichtung des Datenstroms an. Zwischen den übermittelnden *Kinderknoten* und den empfangenden *Elternknoten* erfolgt der einseitige Datenaustausch, indem *Ereignisse* (engl. Events) versendet bzw. von einem Knoten zum anderen „gedrückt“ werden. Ein Knoten kann im speziellen Fall sowohl Kinder- als auch Elternknoten sein.

[36] http://xerces.apache.org/

[37] Document Object Model (DOM) dient als Spezifikation von Schnittstellen, welche für den Zugriff auf HTML oder XML Dokumente zugeschnitten sind. Die DOM Implementierung erlaubt dem Benutzer eine objektorientierte Strukturierung des Dokuments in Klassen, Methoden und Attributen. Mehr unter http://de.selfhtml.org/dhtml/modelle/dom.htm.

[38] http://www.cs.wustl.edu/~schmidt/ACE.html

Das Ereignisaustauschmodell ist der grundlegende Mechanismus hinter dem Datenflusskonzept.

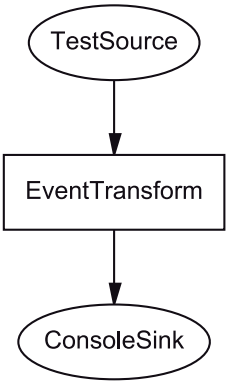


Abbildung 4.16: einfacher linearer Datenflussgraph

Startknoten (engl. Source Nodes) werden im Datenflussgraphen durch Blätter repräsentiert und erhalten ihre Daten von externen Quellen. Die Informationen fließen dann in das OpenTracker System ein. Die Mehrzahl der Knoten verkapselt Gerätetreiber, welche direkt auf einen bestimmten Trackingapparat zugreifen. Sie können auch Verbindungsbrücken zu komplexen, abgeschlossenen Systemen formen, wie z.B. dem ARToolKit. Weitere Startknoten emulieren wiederum Tracker via Keyboard, adressieren Netzwerkdaten oder antworten mit konstanten Werten (nützlich bei Debugging Prozessen). Die standardmäßige Einstellung der Knoten liefert nur die zuletzt erhaltenen Daten von einem Trackingapparat oder dem Netzwerk.

Umwandlungsknoten (engl. Filter bzw. Transformation Nodes), sind Zwischenknoten und verändern bzw. kombinieren Ausgabewerte und Parameter, die sie von ihren Kindern erhalten haben. Durch die erhaltenen, gefilterten Informationen aktualisiert der Umwandlungsknoten seinen Systemstatus. Es gibt insgesamt neun verschiedene Filter, die zum Einsatz kommen:

- Umwandlungsfilter: Die Werte der Kinderknoten werden geometrischen Transformationen unterworfen.
- Tastenfilter: Ob eine Eingabe über ein angeschlossenes Gerät erfolgt wird über boolesche Operationen bestimmt. Dieser Wert wird dann in einem neuen Datenereignis festgehalten.
- Voraussagefilter: Um Verzögerungen so gut wie möglich im Vorhinein zu verhindern, wird dieser Filter auf gefährdete Trackingdaten angewandt.
- Rausch- und Glättungsfilter: Damit werden inhärente Ungenauigkeiten bei Trackern behoben.
- Entzerrungsfilter: Bei RF unterstützten Tracking Systemen müssen Verzerrungen im Magnetfeld durch diesen Filter behoben bzw. linearisiert werden.

- Fusionsfilter: Die Daten von mehreren Kinderknoten werden vereint, um daraus neue Werte zu gewinnen.
- Übersetzungsfilter: Damit wird ein Datentyp in einen anderen umgewandelt, indem ein konstanter Wert hinzugefügt wird (z.B. 2D Positionen nach 3D).
- Abklemmfilter: Der nichtlineare Filter wird an benutzerspezifischen Extremwerten angewandt, um diese an der Stelle abzuschneiden.
- Konfidenzfilter: Informationen verschiedener Kinderknoten werden basierend auf vertraulichen Messwerten bezüglich ihrer Datengenauigkeit ausgewählt.

Endknoten (engl. Sink Nodes) ähneln im Aufbau Startknoten, verbreiten jedoch an externen Ausgängen die gesammelten Informationen aus dem Datenfluss. Auf diese Weise können Dateien oder Applikationen von Außerhalb erreicht bzw. weitere OpenTracker Graphen via Netzwerkverbindungen angesteuert werden. Dabei zeichnet ein *Protokollierungsknoten* (engl. Logging Node) die erhaltenen Daten auf und speichert sie in einer Datei. Der aufgenommene Datenfluss kann dann durch den entsprechenden Startknoten wiedergegeben werden.

Jeder Knoten hat einen oder mehrere *Eingabeports* und einen *Ausgabeport*, der wiederum mit jedem Eingabeport eines weiteren Knotens verbunden sein kann. Multiple Eingabeports sind notwendig, da Berechnungen oft mehr als einen Parameter beinhalten. Ein Port ist ein eindeutig identifizierbarer Verbindungspunkt für eine Kante. Dadurch ist es z.B. dem dazugehörigen Knoten möglich, zwischen den Ereignissen zu unterscheiden, die durch seine verschiedenen Ports geleitet werden. Demzufolge wird der Datenfluss durch gerichtete Kanten im Graphen geschaffen. Ein Knoten, der über einen seiner Eingabeports ein neues Datenereignis empfängt, aktualisiert sich selbst und damit seinen momentanen Zustand. Über seinen Ausgangsport versendet er anschließend das neue Datenereignis.

Mehrere Kinderknoten können Verbindungen zu den Eingabeports eines Elternknotens haben. So wählt z.B. ein *Verschmelzungsknoten* (engl. Merge Node) nur einen Teil der empfangenen Daten eines Ereignisses aus bzw. kombiniert diese und erzeugt infolgedessen ein eigenes Ereignis. Dabei ist jeder Eingabeport des Verschmelzungsknotens mit anderen Kinderknoten bzw. deren Ausgabeports verbunden.

Des Weiteren kann auch ein einziger Eingabeport mit mehreren Ausgabeports verbunden sein, um eine Zusammenführung der Ereignisse zu erzielen. Der Elternknoten kann hier nur zwischen den Eingabeports, nicht aber zwischen den Kinderknoten, unterscheiden, woher er die Ereignisse empfängt.

Umgekehrt dient die Kombination von einem Ausgangsport mit mehreren Eingabeports einer Ausfächerung der Ereignisse innerhalb des Graphen. Dabei verweist ein

Referenzknoten (engl. Reference Node), der keine Kinder besitzt, auf den bezugnehmenden Knoten samt Untergraphen und imitiert seine Elemente (siehe Abbildung 4.17). Der Knoten verteilt alle Ereignisse gleichmäßig unter den betreffenden Elternknoten.

Dabei werden die Elemente des ursprünglichen Knotens mit eindeutigen Identifikationen (IDs) verknüpft. Auf die IDs verweisen dann die Elemente des Referenzknotens und spezifizieren so den Knoten, der imitiert werden soll. Der Referenzknoten entspricht in allen Belangen dem Knoten, auf den er verweist. Somit können die Ausgabedaten eines Knotens im Datenflussgraphen an anderer Stelle verwendet werden.

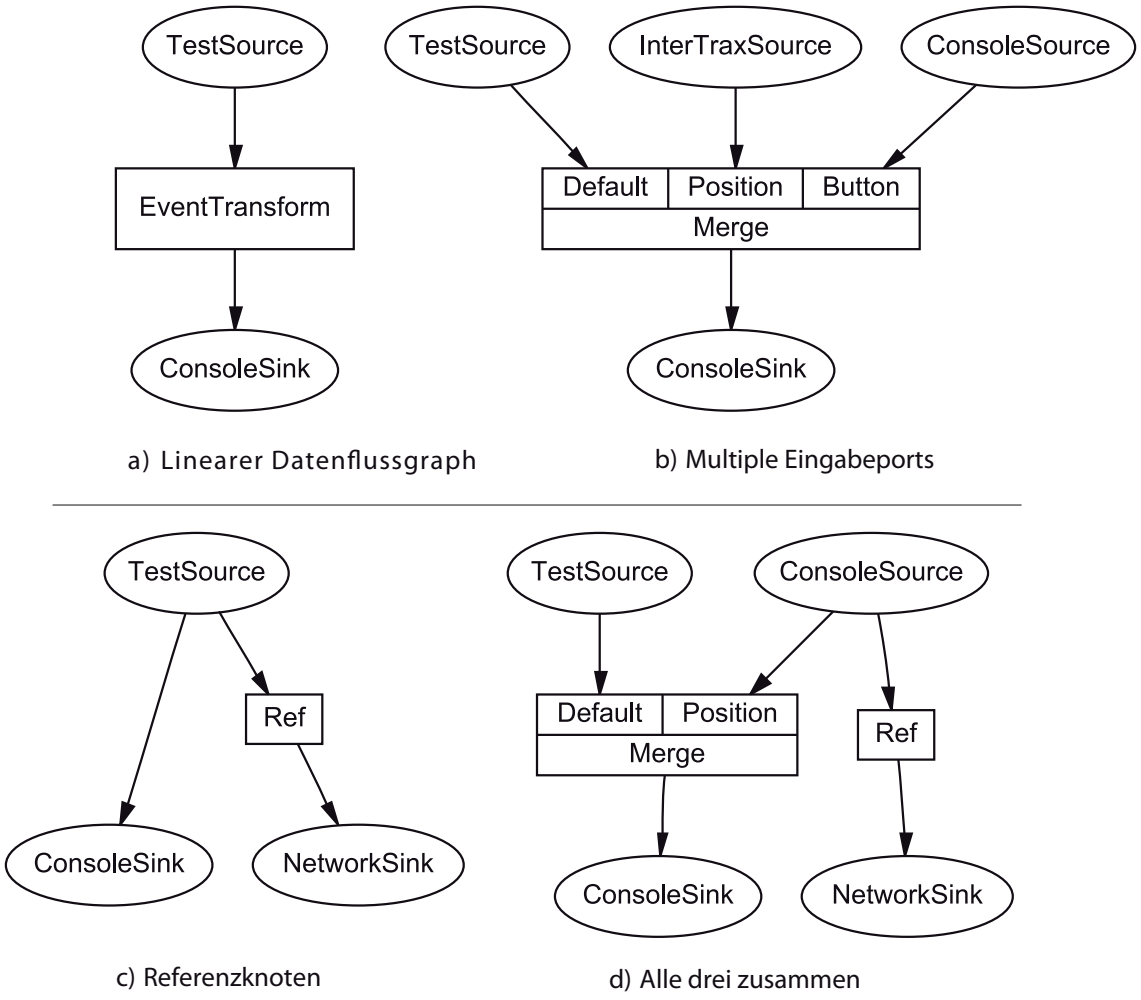


Abbildung 4.17: a) einfache und b-d) komplexe Modelle eines Datenflussgraphen

Nicht alle OpenTracker Berechnungen lassen sich in das grundlegende Modell des Ereignisaustausches via Ports zwischen Kinder- und Elternknoten einordnen. Dazu zählen z.B. Algorithmen, die den Trackerstatus zu einem gewissen Zeitpunkt berechnen

und hierfür unterschiedliche Eingabe- und Ausgabeschnittstellen nutzen. Im Datenfluss werden solche Schnittstellen, wie bereits erwähnt, durch Kanten repräsentiert.

Ereigniskanten (engl. Event Edges) implementieren über ihre Schnittstelle das grundlegende Prinzip der Ereignisverschiebung, bei dem neue Datenereignisse von Kinderknoten an ihre Elternknoten übermittelt bzw. „gedrückt" werden. Jedes abgehende Ereignis erhält dabei durch seinen Kinderknoten einen Zeitstempel. Dadurch reagieren diese Knoten auf zeitabhängige Abläufe, die Trackingdaten betreffen.

Ereignisreihungskanten (engl. Event Queue Edges) sind Ansammlungen von (in Warteschlangen) gereihten Ereignissen und stellen, ähnlich wie *zeitabhängige Kanten* (engl. time-dependent Edges), Erhebungsmechanismen zu Datenberechnungen bereit. Die Ereignisreihungsschnittstelle unterstützt die Abfrage und Identifizierung der gespeicherten Ereignisse über ihre Indizes. Eine zeitabhängige Schnittstelle/Kante führt seine Kalkulationen zu einem festgelegten Zeitpunkt aus, an welchem die dazugehörigen Daten zurückgegeben bzw. „gezogen" werden.

Hauptklassen sowie Schnittstellen und Module

Die Softwarebibliothek weist eine hierarchische Klassenstruktur seiner Objekte auf. Sie besteht im Kern aus einigen *Hauptklassen*, welche die grundlegenden Knotenschnittstellen implementieren, einem *Parser*[39], der die Struktur der Laufzeit[40] durch eine XML Konfigurationsdatei festlegt und der *Hauptschleife*, die durch das Ereignisaustauschmodell angetrieben ist. Weitere Funktionalitäten werden über drei leicht modifizier- und erweiterbare *Modulklassen* realisiert (Krispel, 2006):

- Die *NodeFactory* Schnittstelle erlaubt einem Parser neue Knoten zu erstellen, indem sie die verschiedenen Knotentypen instanziiert. Sie übernimmt für den ihr bekannten Knoten das eigentliche Parsing.
- Die *Module* Schnittstelle unterstützt Berechnungen innerhalb der Hauptschleife. Sie sollte gemeinsam mit der NodeFactory Schnittstelle in derselben Klasse implementiert sein, damit ein Modul die neu erstellten Knoten leicht wiederfindet. Module lesen bzw. „ziehen" Daten aus dem Datenflussgraphen.
- Die *Context* Schnittstelle organisiert alle Module, wendet den Parser an und implementiert die Hauptschleife. Sie fügt alle beteiligten Komponenten zusammen, indem sie mittels ConfigurationParser Objekt alle notwendigen Module und NodeFactories der Applikation speichert und die Konfigurationsdateien parst.

[39] Ein Parser (bzw. Zerteiler) wandelt Eingabedaten in ein verwertbares Format zur Weiterverarbeitung um und stellt diese dann bereit.

[40] Die Laufzeit (engl. Runtime) ist die Dauer, die ein Programm von einem Rechner ausgeführt wird.

Um so viel Flexibilität wie möglich zu gewährleisten, beinhaltet die Applikation keine fixen Schnittstellen. Entwickler greifen entweder auf einen der vielen vorhandenen Knoten zurück oder sie wenden ihre eigenen Module an (z.B. wurde ein reacTIVision-Module am INSO entwickelt, um gleichnamiges Framework am 4T Tisch einzusetzen), welche Endknoten als Schnittstellen ihrer Applikation implementieren. Stellvertretend für die Vielzahl an Modulen[41] in OpenTracker wird nun das *ARToolKitModule* kurz beschrieben.

Das ARToolKitModule ist die Schnittstelle zur ARToolKit Softwarebibliothek, um Ausrichtung und Position von optischen Referenzmarkierungen in einem Tracking System zu erfassen und Änderungen zu verfolgen. Durch *ARToolKitSource* Knoten erhält sie ihre Informationen und schleust Ereignisse in den Datenflussgraphen ein. Parametereinstellungen werden am Modul über das Element *ARToolKitConfig* vorgenommen.

XML Konfiguration und DTD Datei

Der Datenflussgraph wird durch eine *XML Konfigurationsdatei* und entsprechenden *Dokumenttypdefinitionen (DTD)*[42] beschrieben. In der DTD Datei werden alle Beziehungen, Beschränkungen und Regeln des Graphen, u.a. die Knotenstruktur, unabhängig von den eigentlichen Daten, deklariert. So entspricht die Kinder-Eltern-Knotenbeziehung exakt der Repräsentation im Datenflussgraphen. XML-fähige Software ist nun in der Lage diese DTD zu bearbeiten.

Ein DTD Editor kann die Datei verändern und dient Wartungszwecken, während ein XML Parser wie Xerces die Konfigurationsdatei ausliest, um die entsprechende Baumstruktur im Speicher abzubilden. Diese repräsentiert die Elemente und Attribute sowie ihre Beziehungen zu einander. Der Parser ist auch imstande ungültige Notationen festzustellen, indem er eine Validierung der XML Datei durchführt, auf die sich die DTD Datei bezieht.

Ein GUI-basierter XML Editor erlaubt es die Trackerkonfiguration zu gestalten bzw. zu verändern. Dabei muss der Benutzer nicht grundsätzlich über die Syntax der Konfigurationsdatei Bescheid wissen, da diese bereits in der entsprechenden DTD Datei definiert ist. Dadurch übernimmt der Editor ähnlich dem Parser auch noch die Funktion der Fehlerüberprüfung. Die DTD Datei dient ihm als Grundlage für den korrekten Aufbau der XML Datei.

---

[41] http://studierstube.icg.tu-graz.ac.at/opentracker/html/module_ref.php.

[42] In der DTD wird Struktur eines Dokumentes festgelegt. Die Deklaration umfasst Elementtypen, Attributlisten, Entitäten und Notationen. Mehr unter http://de.selfhtml.org/xml/dtd/index.htm.

OpenTracker überprüft den Baum, der durch den XML Parser generiert wurde, und erstellt für jedes Element ein neues Objekt, basierend auf den Elementnamen. Die Folgewerte der Attribute werden anhand ihrer Objektklassen geparst und die korrespondierenden Objektmitglieder gesetzt. Die Struktur des gerichteten Graphen entsteht durch Verwendung eindeutiger Elemente und ihren Referenzen. XML verstärkt die unverwechselbare Kennzeichnung der IDs und Xerces vereinfacht die Suche nach referenzierten Elementen.

Das Ereignisaustauschmodell des Datenflussgraphen arbeitet nach dem Prinzip eines *Druck/Zug-Mechanismus* (engl. push/pull). Die Ereignisschnittstelle (im Graphen durch die Ereigniskante repräsentiert) verwendet den *Druck-Mechanismus*, bei dem ein momentanes Ereignis vom Startknoten über Zwischenknoten an den Endknoten „gedrückt“ wird. Jeder Knoten führt dann die Aktualisierungsmethode seiner Elternknoten aus. Anschließend rufen diese die entsprechende Methode rekursiv bei ihren eigenen Elternknoten auf, nachdem sie das Ereignis verarbeitet haben (siehe Abbildung 4.18).

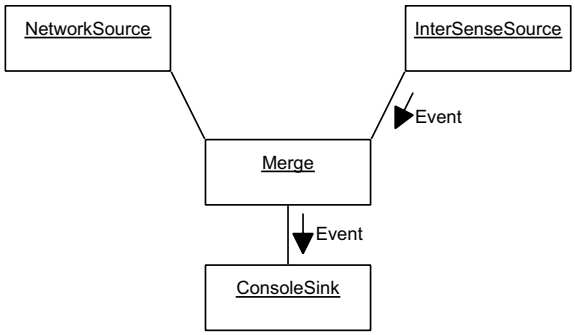


Abbildung 4.18: Startknoten „drücken“ Ereignisse (Events) durch den Graphen

Die Module, die mit den Startknoten verknüpft sind, müssen also nur die Ereignisweitergabe garantieren und aufrecht erhalten, indem sie die Aktualisierungsmethode ihrer Startknoten aufrufen, um diese auf den neuesten Stand zu bringen. Lediglich eine Referenz des Objekts mit den gespeicherten Ereignisdaten wird weitergegeben.

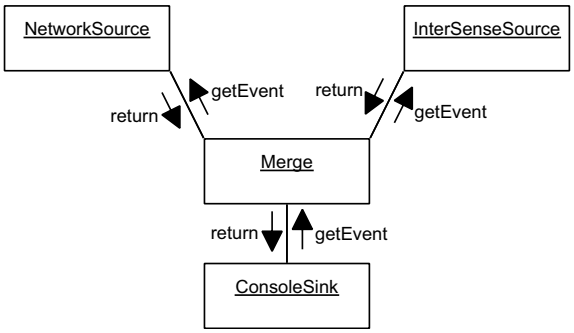


Abbildung 4.19: Endknoten „ziehen“ Ereignisse (Events) durch den Graphen

Ereignisreihungsschnittstellen sowie zeitabhängige Schnittstellen benutzen den *Zug-Mechanismus*. Sie benötigen Informationen zu vorangegangenen Kinderknoten, falls ein Knoten über eine der beiden Schnittstellen abgefragt wird. Dabei rufen sie mit den entsprechenden Parametern die Schnittstellen ihrer Kinderknoten rekursiv auf (siehe Abbildung 4.19).

4.3 Schlussfolgerungen

OpenTracker implementiert die vorgestellten Frameworks für optisches und elektromagnetisches Tracking als Module, die teilweise bereits vorhanden sind (z.B. ARToolKitModule). Andere wurden neu programmiert (z.B. reacTIVision-Module).

4T nutzt OpenTracker bzw. die Module als Schnittstelle für die Informationsgewinnung am Rückprojektionstisch. Digitalkamera und RFID Antennen können so ihre Steuerungselemente erfassen und die gesammelten Daten an das Framework übermitteln. Dabei setzt man ebenfalls auf die Einfachheit und die Stärken der XML Konfiguration. Nur so wird gewährleistet, dass 4T alle Rahmenbedingungen einer intuitiven Computerschnittstelle zur effizienten Bedienung von TUI Applikationen aller Art erfüllt.

5 Erweiterung und Architektur des 4T Frameworks

Im nun folgenden Kapitel wird reacTIVision als alternatives optisches Trackingverfahren für das 4T Framework vorgestellt. Obwohl das ARToolKit im Tangible Tabletop Bereich und bei 4T respektable Tracking Ergebnisse liefert, wird es doch hauptsächlich für VR Applikationen verwendet.

Experimente am 4T Rückprojektionstisch haben außerdem gezeigt, dass die Digitalkamera oberhalb seiner Oberfläche montiert sein muss, wenn das ARToolKit für das Object Tracking verwendet wird. Anders ist kein effizientes Verfolgen und Erfassen der optischen Markierungen an den Steuerungsobjekten möglich.

Gleichzeitig führt aber die Position der Kamera (über Kopf) auch zu gewissen Behinderungen bei der Bedienung des TUIs, da das (teilweise) Abdecken der optischen Markierungen durch die Hand des Benutzers die Erfassung der tangiblen Steuerungselemente deutlich erschwert.

Leider konnte bis dato keine Konfiguration oder Kameraposition für das 4T Projekt gefunden werden, die ähnlich erfolgversprechende Ergebnisse für das Tracking von unterhalb der sandgestrahlten Tischfläche erzielt. Diese Einschränkungen des ARToolKits ließen schließlich Überlegungen nach neuen optischen Trackingverfahren laut werden.

Da OpenTracker und 4T nicht nur eine gemeinsame XML Konfiguration besitzen, sondern auch beide modular aufgebaut sind, ist die Erweiterung des 4T Frameworks durch das stabile und robuste reacTIVision Framework nur eine logische Konsequenz nach den bisherigen Trackingerfahrungen mit ARToolKit. Da es primär für tischbasierte TUI Applikationen (siehe reacTable, Kapitel 2.2.1) gedacht ist, sollte es die Qualität des Trackings (nicht nur von unten) im 4T System dementsprechend verbessern können.

Ferner versteht sich das reacTIVision Framework auch in erster Linie als Standalone Tracking Anwendung für den gemeinschaftlichen Einsatz, engl. *Distributed Application Framework*, und nicht als reine Softwarebibliothek wie das ARToolKit.

Neben der Integration von reacTIVision wird in den kommenden Seiten ein kurzer Einblick in die Entwicklungsumgebung von 4T gegeben und eine fertige TUI Applikation präsentiert. Den Abschluss machen diverse Designvorschläge für mögliche weitere Änderungen am 4T Framework.

5.1 Optical Tracking – reacTIVision

Das *reacTIVision* Framework[43] (Kaltenbrunner & Bencina, 2007) ist ein zweidimensionales optisches Tracking System für Tangible Tabletops. Es wurde seinerzeit primär für den reacTable entwickelt und ist seit 2005 als plattformübergreifende Open-Source Software verfügbar. Das Programm erkennt und verfolgt in Echtzeit Steuerungsobjekte einer TUI Applikation anhand von Referenzmarkierungen (engl. Fiducials).

Um die optischen Markierungen auf den Objekten schnell und präzise auf einem geeigneten Computersystem verarbeiten zu können, benötigt das Framework eine hochauflösende Infrarot (IR) Digitalkamera zur Datenerfassung und einen Projektor zur visuellen Rückmeldung. Zusätzlich unterstützt es Multi-Touch Finger Tracking auf interaktiven Oberflächen.

5.1.1 Voraussetzungen

Bei der Entwicklung bzw. Installation eines Tracking Systems, welches reacTIVision als Framework einsetzt, müssen je nach Verwendungszweck einige Vorkehrungen getroffen werden.

Da die Hardware bzw. die technische Umsetzung in den meisten Fällen vom Benutzer nicht wahrgenommen werden soll, sind alle Komponenten (Kamera, Beleuchtung, Projektor, Hardware) unter der Tischfläche anzubringen. Das hat zum einem den Vorteil, dass der User in seiner intuitiven Handhabung am TUI nicht abgelenkt wird, zum anderen wirkt das Gesamtbild des Tangible Tabletops professioneller und ästhetischer. Außerdem beeinflusst der Verwendungszweck die Tischgestalt. Für den mobilen Einsatz muss die Konstruktion rasch zusammen gebaut werden können (und umgekehrt). Trotzdem muss sie robust gebaut sein und unempfindlich auf Erschütterungen reagieren.

Die Oberfläche des Tisches muss lichtdurchlässig sein. Im Idealfall verwendet man einseitig sandgestrahltes Plexiglas (die behandelte Fläche zeigt nach oben), aber aus Kostengründen kann auch eine Glasfläche mit Transparentzeichenpapier verwendet werden. Es muss nur sichergestellt sein, dass die Tischoberfläche nicht komplett durchsichtig ist, da sonst ein Projektor optische Informationen nicht zufriedenstellend darauf darstellen könnte.

[43] Die aktuelle reacTIVision Version steht unter http://www.sourceforge.net/projects/reactivision/ zum Download bereit.

Abbildung 5.1: Die richtige Wahl der Tischgröße

Eine runde Tischfläche ermöglicht die gemeinschaftliche Nutzung des TUIs etwas besser auszureizen als eine eckige, da ein Kreis den Benutzerzugang von allen Seiten zulässt und jedem den gleichen interaktiven Spielraum verschafft. Der Aktionsradius jüngerer Benutzer ist (z.B. beim Music Table aufgrund der Tatsache, da es sich um Kinder handelte) teilweise stark eingeschränkt (Berry et al., 2006). Daher muss die Größe der interaktiven Tischfläche entsprechend der physischen Reichweite der bevorzugten Benutzergruppe passend gewählt sein (siehe Abbildung 5.1).

Je nachdem wie groß der Tisch (bzw. die Tischfläche) ist, muss auch die Digitalkamera den entsprechenden Bedingungen und technischen Anforderungen genügen. Handelsübliche USB 2.0 bzw. FireWire Kameras mit einer Auflösung von mindestens 640 x 480 Pixel und 30 fps sind im Normalfall geeignet. Da die Referenzmarkierungen schwarz/weiß sind, ist keine Farbunterstützung notwendig. Beim reacTable kommt eine AVT Guppy F-080B/C Digitalkamera zum Einsatz. Für großflächige Systeme bzw. einen professionellen Einsatz sind allerdings spezielle Industriekameras, ausgestattet mit hochempfindlicher Sensortechnik (CCD, CMOS[44]), notwendig, um verwertbare Resultate zu erzielen. Bei Videokameras dürfen keine Videosignalmodi mit Zeilensprungverfahren (Interlaced Video Signal) verwendet werden, da das Bild der Fiducials zerstört und nicht korrekt erfasst werden würde. Nur Aufnahmen im Full-Frame Mode sind geeignet.

Lichtverhältnisse spielen bei optischen Trackingverfahren eine entscheidende Rolle. Kamera und Projektor müssen in unterschiedlichen Lichtspektren operieren, um gegenseitige Konflikte (z.B. Projektion überlagert Fiducial ID Marker) zu vermeiden. Da der Projektor augenscheinlich im sichtbaren Lichtbereich darstellt, darf die

[44] Complementary Metal Oxide Semiconductor (CMOS) ist wie CCD ein Bildsensor, der aber mehr Rauscheffekte als CCD aufweist und daher meist nur in günstigeren Kamerasystemen verbaut wird.

Digitalkamera nur im IR[45] Bereich arbeiten. Des Weiteren ist eine modifizierbare Kamera (austauschbare Filter und Linsen) von Vorteil. CCD Kameras verrichten die Bilderkennung einzig und allein im IR Lichtspektrum und sind daher zu bevorzugen. Es ist jedoch darauf zu achten, dass ein möglicher IR-Sperrfilter (blockt IR Licht) entfernt und durch einen IR Pass Filter (blockt sichtbares Licht) ersetzt wird. So ist gewährleistet, dass die Kamera korrekt erfasst und sichtbares Licht vom Projektor keinen Störfaktor bildet. Zusätzlich sollte das System durch geeignete IR Beleuchtung zwecks ausreichender Lichtverhältnisse versorgt sein.

Der Projektor muss entsprechend gekühlt werden (z.B. Lüfter), wenn er unterhalb der Tischfläche verbaut ist, da von ihm beträchtliche Wärmestrahlung ausgeht. Gleiches gilt für das Computersystem, auf dem reacTIVision und die jeweilige Tangible Applikation ausgeführt werden. Auch beim Projektor ist aufgrund des flächendeckenden Einsatzes ein Weitwinkelobjektiv zusätzlich zu einer ausreichend starken Lampe zu bevorzugen.

Wird eine Tischbauweise gewählt, die das optische Tracking von unten voraussetzt, können je nach Konstruktion auch Spiegel bzw. Kameras mit Weitwinkelobjektiv eingesetzt werden. Beide Varianten besitzen den Vorteil, dass die Bilderfassung großer Flächen über eine möglichst kurze Entfernung erfolgen kann, wobei die notwendigen Brennweiten (Fokus) trotzdem erzielt werden, vorausgesetzt, dass die Einstellungen an der Kamera auch korrekt durchgeführt sind. Zusätzlich bietet reacTIVision die Möglichkeit, dass auf diese Weise entstandene Bildverzerrungen durch die eingebaute Kalibrierungsfunktion behoben werden bzw. gar nicht erst auftreten können.

5.1.2 Architektur des reacTIVision Frameworks

Jede Komponente des Frameworks kann einzeln und daher bei Bedarf auch auf unterschiedlichen Computersystemen ausgeführt werden (siehe Abbildung 5.2). Aufgrund dieses baukastenartigen Aufbaus ist auch die reacTIVision Software um weitere Bilderfassungs- und Frame-Processing-Module erweiterbar.

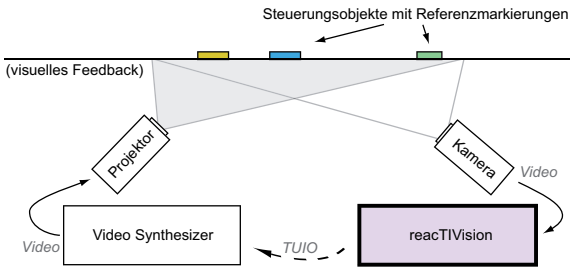


Abbildung 5.2: Diagramm des reacTIVision Frameworks

[45] Als Infrarot wird der Spektralbereich des Lichts mit Wellenlängen von 780 nm bis 1 mm bezeichnet. IR Licht ist für das menschliche Auge nicht wahrnehmbar.

Für die Datenübertragung zwischen den Programmteilen ist das *TUIO Protokoll* (Kaltenbrunner et al., 2005) verantwortlich, das dafür eine Reihe von OSC[46] Befehlen (Wright et al., 2003) bereitstellt. Darin definiert TUIO sowohl allgemeine Eigenschaften der Steuerungselemente (Präsenz, Position, Ausrichtung), die sich auf der Tischfläche befinden, als auch mögliche Finger- und Handbewegungen durch den Benutzer.

reacTIVision sendet permanent die (via Oscpack[47]) verschlüsselten OSC Mitteilungen an jede (lokal oder global) vernetzte TUI Applikation via UDP Port 3333. Der Client dekodiert sie und setzt die OSC Befehle (*set*, *alive*, *fseq*) entsprechend den erfolgten Aktionen am Tangible Tabletop um (siehe Tabelle 5.1). Damit reacTIVision auch mit älteren Musikprogrammen kompatibel einsetzbar ist, unterstützt es auch das langsamere MIDI[48] Protokoll zur Datenübertragung.

Parameter	Bedeutung
s	sessionID, temporäre Object ID, int32
i	classID, Fiducial ID Nummer, int32
x, y, z	Position, float32, range 0...1
a, b, c	Winkel, float32, range 0..2PI
X, Y, Z	Bewegungsvektor, float32
A, B, C	Rotationsvektor, float32
m	Bewegungsbeschleunigung, float32
r	Rotationsbeschleunigung, float32
p	Freier Parameter, Typ durch OSC Packet Header definiert

Tabelle 5.1: Die Parameter des *set* Befehls

[46] Open Sound Control (OSC) ist ein frei zugängiges, transportunabhängiges, nachrichtenbasiertes Protokoll, das Kommunikation zwischen Computern, Sound Synthesizern und allen weiteren Multimedia Geräten in einem Netzwerk unterstützt. Mehr unter http://opensoundcontrol.org/.

[47] Die Softwarebibliothek Oscpack ver- und entpackt OSC Befehle. Mehr unter http://www.audiomulch.com/~rossb/code/oscpack/.

[48] Musical Instrument Digital Interface

TUIO Profile	
2D Oberfläche	`/tuio/2Dobj set s i x y a X Y A m r` `/tuio/2Dcur set s x y m r`
2.5D Oberfläche	`/tuio/25Dobj set s i x y z a X Y A m r` `/tuio/25Dcur set s x y z m r`
3D Oberflächen	`/tuio/3Dobj set s i x y z a X Y Z A m r` `/tuio/3Dcur set s x y z m r`
Rohdaten Profil	`/tuio/raw_[profileName]` `/tuio/raw_dtouch set i x y a`
Benutzerdefiniertes Profil	`/tuio/_[formatString]` `/tuio/_sixyP set s i x y 0.57`

Tabelle 5.2: vordefinierte TUIO Profile für Tangible Tabletop Anwendungen

Des Weiteren stellt TUIO vordefinierte Profile (siehe Tabelle 5.2) für die gängigsten Tangible Tabletop Anwendungen zur Verfügung (Kaltenbrunner et al., 2005). Falls keine der vorgegebenen Einstellungen passend sein sollte, gibt es auch Rohdaten und benutzerdefinierte Profile, die entsprechend angepasst werden müssen.

Das Framework besitzt eine Reihe von Client Beispielen für verschiedene Programmiersprachen (z.B. C++, C#, Java, Pure Data[49]), die als Grundlage bei der Entwicklung von TUIs dienen und eine einfache Integration erlauben. Designer erhalten bei der Entwicklung zusätzliche Unterstützung durch den *TUIO Simulator*, der das gleichnamige Protokoll implementiert und eine komplettes Tabletop System ohne die dafür notwendigen Hardwarevoraussetzungen virtuell nachahmt. Dadurch wird Software Debugging (vor allem in der Eingangsphase) erleichtert.

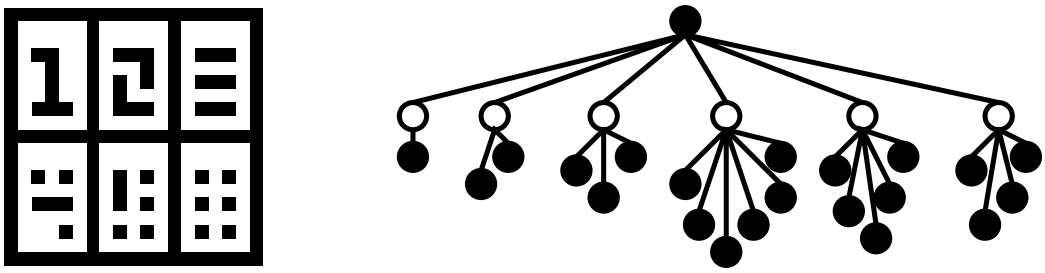


Abbildung 5.3: klassische Referenzmarkierung (l.) und ihr Adjazenzgraph (r.)

[49] Pure Data ist eine datenstromorientierte Programmiersprache sowie Entwicklungsumgebung, die vor allem zur Erstellung interaktiver Multimedia Applikationen (z.B. Audio Synthesizer) eingesetzt wird. Dabei werden die Resultate als Grafiken visualisiert. Mehr unter http://puredata.info/.

5.1.3 Fiducial Recognition Engines

Eine Kamera übermittelt dem System Bilder der Tischfläche. Dabei wird jeder Frame via PortVideo[50] einzeln abgetastet und auf die speziellen Referenzmarkierungen untersucht. Erfolgt eine positive Erkennung, wird der Frame in ein monochromes Bild umgewandelt. Dann wird das Bild anhand seiner schwarzen und weißen Anteilen durch einen Adjazenzgraphen[51] in Regionen unterteilt (siehe Abbildung 5.3).

Dieser Graph wird anschließend nach bestimmten Baumstrukturen, die in den Fiducials verschlüsselt sind, durchsucht. Erfolgreiche Treffer werden im Musterabgleich mit Referenzmustern aus einem Katalog verglichen, um so jeder Markierung ihre eindeutige ID Nummer zuzuweisen. Die effiziente Gestaltung der Fiducial IDs erlaubt somit eine exakte Bestimmung der Steuerungsobjekte, ihrer Position und ihrer Ausrichtung auf der Tischfläche. Die Daten werden dann durch OSC Befehle an alle reacTIVision Programmteile via TUIO Protokoll übermittelt.

Die Resultate der Bildsegmentierung werden auch beim Finger Tracking eingesetzt, um Fingerabdrücke und Bewegungen auf der Tischfläche als weiße Punkte zu erkennen. Dabei werden mathematische Verfahren angewandt, die sowohl die Fingerpunkte als auch nicht erkannte Fiducials erfassen. Die genauen Algorithmen zur Erzeugung (Design, Evolution, genetischer Algorithmus) der Referenzmarkierungen und zum Ablauf ihrer Bildsegmentierung (topologisches Tracking) werden in den wissenschaftlichen Arbeiten von Bencina & Kaltenbrunner (2005) und Bencina et al. (2005) im Detail beschrieben.

reacTIVision verfügt über drei Fiducial Recognition Engines (siehe Abbildung 5.4). Dabei dient die Softwarebibliothek *Libfidtrack* der Erkennung und Erfassung der Referenzmarkierungen (Bencina et al., 2005). Für das Finger Tracking werden eigene Symbole verwendet.

[50] PortVideo ist ein Open-Source Framework zur Bild- und Videoerfassung. Es entstand im Zuge des reacTable Projekts. Mehr unter http://mtg.upf.es/reactable/?portvideo.

[51] Adjazenzgraphen erzeugen einen Graphen, dessen Kanten benachbarte Regionen zusammenfügen. Seine Knoten sind den Regionen zugeordnet. Diese Konstellation entspricht einer Kombination aus Breiten- und Tiefensuche.

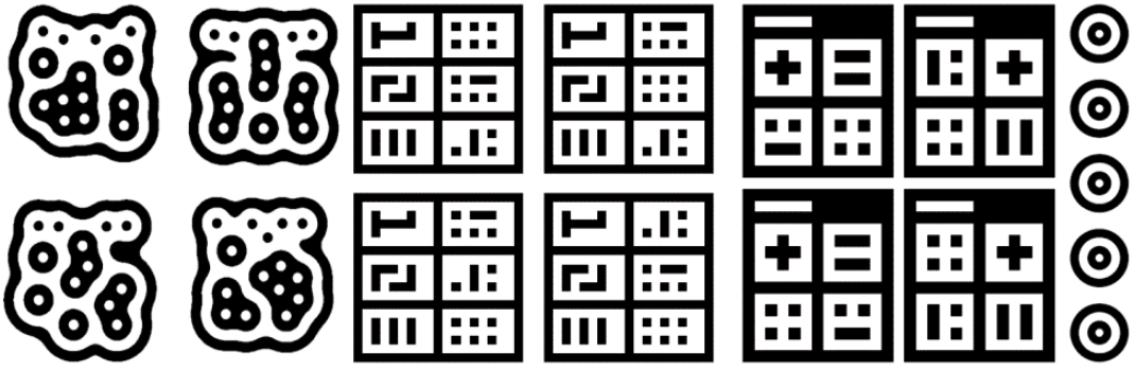


Abbildung 5.4: Amoeba, klassische, D-Touch sowie Finger Tracking Symbole (v.l.n.r.)

Die Referenzmarkierungen der *Amoeba Engine* sind durch den genetischen Algorithmus bei ihrer Erzeugung besonders auf Größe, Form, Mittelpunkt und Rotationswinkel optimiert worden. Alle 90 verschiedenen Symbole besitzen eine Baumstruktur mit jeweils 19 Blattknoten und einer maximalen Baumtiefe von 2 (siehe Abbildung 5.5).

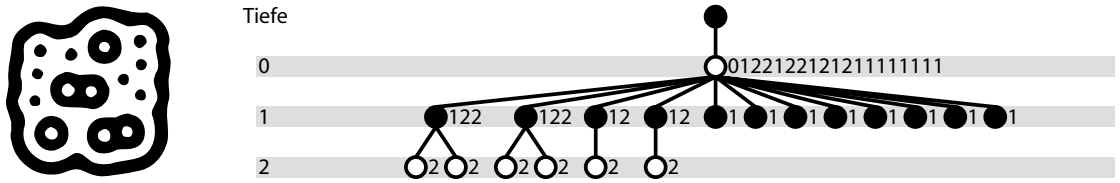


Abbildung 5.5: Amoeba Referenzmarkierung (l.) und ihr Adjazenzgraph (r.)

Die spezielle Baumstruktur der Amoeba Symbole ist aufgrund ihrer topologischen Eigenheiten ausschlaggebend für die Robustheit und Genauigkeit der Bilderkennung, da beim optischen Tracking weniger Störfaktoren bzw. falsch positive Strukturen wahrgenommen werden. Die Standard Engine bildet per Grundeinstellung Amoeba.

Die *D-Touch Engine* war zunächst zur freien Distribution GPL lizensiert und ist seitdem Bestandteil von reacTIVision. Im Framework sind ihre Symbole erweitert und würfelähnlich angeordnet worden (24 Permutationen der Regionen, jeweils 1-4 Subregionen). Die leere zusätzliche Region (am Symbol oben links angeordnet, siehe Abbildung 5.4) dient Winkelberechnungen. D-Touch verwendet eine Vielzahl von topologisch basierten Symbolen, darunter auch die klassischen Referenzmarkierungen, welche die ursprünglichen D-Touch Markierungen implementieren.

Die *klassische Engine* beruht zwar auf D-Touch (bzw. die originalen D-Touch Referenzmarkierungen), ist aber komplett neu programmiert und umgesetzt worden. Ihre würfelförmigen Symbole repräsentieren 120 unterschiedliche geometrische Anordnungen. Sie setzen sich aus den möglichen Permutationen von schwarzen Blättern

(bzw. ihrer Anzahl) zusammen, die innerhalb der 2-6 weißen Regionen angeordnet sind. In Abbildung 5.3 wären das im Uhrzeigersinn gelesen: 1, 2, 3, 6, 5, 4. Das obere linke Kästchen des klassischen (D-Touch) Symbols repräsentiert den Rotationswinkel. Im Vergleich zur Amoeba Engine verlaufen Berechnungen des Mittelpunkts (der Referenzmarkierung) und des Winkels ungenauer.

Extrem simpel aber robust gestaltet sich die Markierung für das Multi-Touch Finger Tracking. Seine Baumstruktur besitzt gerade mal einen einzigen Ast (siehe Abbildung 5.4) und wird durch die Amoeba Engine erkannt. Die Referenzmarkierung stellt keinen Platz für Winkelberechnungen bereit, da diese auch nicht benötigt werden. Das Symbol liefert nur Informationen zur Lokalisierung des Fingerabdrucks, welche völlig ausreichend sind.

Lediglich die verstärkt auftretende Erkennung von falsch positiven Markierungen aufgrund erhöhten Bildrauschens stellt ein Problem dar. Diese Fehlerquelle kann durch Berechnungen vorangegangener Fingerabdrücke und voraussichtlichen Bewegungskurven gemindert bzw. teilweise behoben werden.

Mittlerweile wird bereits an einem verbesserten Finger Tracking System gearbeitet, das voraussichtlich beim reacTIVision Framework der nächsten Generation zum Einsatz kommen wird. Diese Engine wird ohne Finger Symbole auskommen und erhöhte Fehlertoleranz durch zweifache Kontrolle gegenüber falsch positiven Identifikationen aufweisen. Des Weiteren sollen neue Referenzmarkierungen von anderen Softwarebibliotheken wie ARToolKit sowie Strichcode und Semacode[52] Dekodierungstechnologie die reacTIVision Engine in naher Zukunft bereichern.

5.2 Rahmenbedingungen für Entwickler (Developer)

Die beiden folgenden Kapitel (5.2 und 5.3) dienen den Hauptbenutzern des Frameworks, dem Entwickler und dem Anwender, als didaktischer Leitfaden für die schrittweise Installation und Inbetriebnahme sowie für die anschließende Gestaltung von TUI Applikationen.

5.2.1 Entwicklungsumgebung

Die nachstehenden Absätze befassen sich mit der 4T Entwicklungsumgebung. Des Weiteren erfährt man mehr über die eingesetzten APIs und Konfigurationsdateien.

Wie bereits zuvor beschrieben baut das 4T Framework auf OpenTracker auf. So ist gewährleistet, dass neue Module bzw. Schnittstellen von OpenTracker stets auch

[52] http://semacode.com/.

kompatibel mit 4T sind. Auf diese Weise stehen dem Entwickler zahlreiche Möglichkeiten der Toolkit Erweiterung zur Verfügung.

Daneben setzt man auch auf eine eigene C++ sowie eine JNI[53] API[54]. Erstere liefert die Anbindung für die 4T Konsole und die 4T.x32 Xtra Datei, welche die Verbindung zur Skriptsprache Lingo herstellt. Die JNI API wiederum dient als Schnittstelle für den 4T-Konfigurator und kann natürlich bei Bedarf um weitere Funktionen ergänzt werden.

Weil das 4T Projekt auf so vielen Softwarebibliotheken und weiteren Frameworks beruht, müssen beinahe ebenso viele Abhängigkeitsverhältnisse unter den beteiligten Komponenten berücksichtigt werden, damit sie als Einheit korrekt funktionieren. Um nicht von Grund auf neu beginnen zu müssen, steht eine Microsoft Visual Studio Solution (erkennbar an der Extension .sln) bereit.

In der Datei *4T.sln* sind die wichtigsten Projekte, die auch als Binärdateien bereits kompiliert vorliegen, integriert. Ursprünglich unter Windows XP (Service Pack 2) mit Microsoft Visual Studio 2005 sowie JDK[55] 1.5.0 (Version 7) entwickelt, sollte 4T auch auf neuen Systemen (wie z.B. Microsoft Vista oder Windows 7) und aktuellem Visual Studio 2008 funktionieren. Möglicherweise müssen dahingehend ein paar Einstellungen angepasst bzw. dynamische Bibliotheken aktualisiert werden.

Die erneute Kompilierung des 4T Frameworks macht vor allem dann einen Sinn, wenn neue Programmversionen der inkludierten Softwarekomponenten zum Einsatz kommen sollen. Nur dann kann die Funktionstüchtigkeit der aktualisierten Programmbibliotheken und zusätzlichen Frameworks innerhalb des 4T Projekts gewährleistet werden.

[53] Java Native Interface (JNI) ist eine standardisierte API. Damit können plattformspezifische Funktionen bzw. Methoden aus der Programmiersprache Java heraus aufgerufen werden.

[54] Application Programming Interface (API) oder Programmierschnittstelle, welche externen Programmen eine Anbindungsmöglichkeit an das eigene System zur Verfügung stellt.

[55] Das Java Development Kit (JDK) beinhaltet neben dem Java Runtime Environment (JRE) zahlreiche Java-Entwicklungswerkzeuge. Mehr unter http://java.sun.com/javase/6/docs/.

_SetEnvironment.cmd

```
@echo off

set ROOT=%~dp0
set ROOT=%ROOT:~0,-1%

echo Setting environment variables to 4T root of "%ROOT%"

reg add "HKCU\Environment" /f /v "4TROOT" /d "%ROOT%"
reg add "HKCU\Environment" /f /v "FOURTROOT" /d "%ROOT%"
reg add "HKCU\Environment" /f /v "4TTABLE" /d "%~dp04TTable"
reg add "HKCU\Environment" /f /v "FOURTTABLE" /d "%~dp04TTable"
reg add "HKCU\Environment" /f /v "ACEROOT" /d "%~dp0ACE_Wrappers"
reg add "HKCU\Environment" /f /v "ARTOOLKITPLUSROOT" /d "%~dp0ARToolkitPlus"
reg add "HKCU\Environment" /f /v "ARTOOLKITROOT" /d "%~dp0ARToolkit"
reg add "HKCU\Environment" /f /v "OPENVIDEOROOT" /d "%~dp0openvideo"
reg add "HKCU\Environment" /f /v "OTROOT" /d "%~dp0opentracker"
reg add "HKCU\Environment" /f /v "RFDEVICE" /d "%~dp0RS232\RFDevice"
reg add "HKCU\Environment" /f /v "XERCESCROOT" /d "%~dp0xerces-c-bin_2_7_0"
reg add "HKCU\Environment" /f /v "XDKPATH" /d "%~dp0XDK"
reg add "HKCU\Environment" /f /v "JNIROOT" /d "%~dp0JNI"

rem Read current PATH env variable from registry

FOR /F "tokens=3 delims=" %%i IN ('reg query "HKCU\Environment" /v "PATH"') DO
set CURRENTPATHVAR=%%i

reg add "HKCU\Environment" /f /t REG_EXPAND_SZ /v "PATH" /d "%CURRENTPATH-
VAR%;%%4TTABLE%%\Debug;%%4TTABLE%%\Release;%%ACEROOT%%\lib;%%ARTOOLKITPLUSROOT%
%\build\VS.NET\Release;%%ARTOOLKITPLUSROOT%%\build\VS.NET\Debug;%%ARTOOLKITROOT
%%\bin;%%ARTOOLKITROOT%%\DSVL\bin;%%OPENVIDEOROOT%%\bin;%%OTROOT%%\bin\win32;%%
RFDEVICE%%\Build;%%RFDEVICE%%\Debug;%%XERCESCROOT%%\bin"

rem reg add "HKCU\Environment" /f /t REG_SZ /v "PATH" /d
"%%4TTABLE%%\Debug;%%4TTABLE%%\Release"

echo.
echo Log off and log on to refresh environment variables!
echo.
pause
```

Tabelle 5.3: _SetEnvironment.cmd im Stammverzeichnis von 4T

Bevor 4T.sln ausgeführt werden kann, muss man mit Hilfe der Datei *_SetEnvironment.cmd* sämtliche notwendigen Variablen und Pfade im Stammverzeichnis /4TTableAPI korrekt setzen (siehe Tabelle 5.3). Nachdem man sich am Betriebssystem ab- und erneut angemeldet hat, kann die Solution gestartet werden. Sie beinhaltet folgende Projekte:

- *4T/4TConsole*: Hier befindet sich die 4TConsole.exe. Mit der Konsole wird das Eingabegerät initialisiert (Digitalkamera) und die Koordinatendaten bei Bewegungen der Steuerungsobjekte ausgegeben. Die Datei ist von 4TTable abhängig.
- *4T/4TTable*: Bildet den Kern des 4T Frameworks und legt die Datei 4TTable.dll an. 4TTable ist abhängig von OpenTracker und Xerces. Von letzterem gibt es nur die Binärdatei.
- *4T/4TXtra*: Erstellt die Datei 4T.x32 für den Adobe® Director®. Das Xtra ist von 4TTable, Xerces und dem Xtra SDK[56] abhängig.
- *ARToolKit/libAR* und *ARToolKit/libARvideo*: Die beiden ARToolKit Bibliotheken libAR.dll und libARvideo.dll werden bei OpenTracker benötigt, falls man das Framework mit AR Unterstützung einsetzen möchte. Die Projekte sind von DSVL abhängig, was sich im Verzeichnis /ARToolKit/DSVL befindet.
- *ARToolKitPlus*: Die Erweiterung von ARToolKit kommt bei 4TTable momentan nicht zum Einsatz, die Unterstützung durch OpenTracker ist jedoch aktiviert. ARToolKitPlus (Version 2.1.0) ist in 4T integriert.
- *JNI*: Erstellt 4TJNI.dll für das Framework und ARJNI.dll für ARToolKit. Um eine Rekompilierung beider Projekte durchzuführen, ist es notwendig, dass JNI bereits installiert ist. Danach muss die Umgebungsvariable %JNIROOT% dementsprechend gesetzt werden.
- *reacTIVision*: Das reacTIVision Toolkit (Version 1.3) wird von OpenTracker Modulen genutzt. Die Unterstützung durch 4T ist in der DTD Datei gewährleistet. Änderungen erfolgen direkt in den XML Konfigurationsdateien über einen Texteditor, da momentan keine 4T-Konfigurator Unterstützung implementiert ist.
- *OpenTracker*: Beinhaltet das OpenTracker Projekt in der Version 1.1.1. Es ist von der Softwarebibliothek ACE, Xerces, ARToolKit, ARToolKitPlus, 4TTable, reacTIVision und OpenVideo abhängig. OpenTracker steht zu Testzwecken für verschiedene Konfigurationseinstellungen als ausführbare Programmdatei zur Verfügung.

Jedes Projekt liegt bereits als Binärdatei vor und kann natürlich auch einzeln rekompiliert werden. Das geschieht über den Aufruf "*Erstellen*" – "*Erstellen*" in Visual Studio, anschließend einfach das gewünschte Projekt auswählen. Alle Frameworks und Softwarebibliotheken sind zum 4T Package zusammengefasst. So hat der Entwickler sämtliche Werkzeuge in der Hand, die für die konstruktive Arbeit mit dem Framework notwendig sind.

[56] Ein Software Development Kit (SDK) besteht aus Programmen und Dokumentationen zu einer bestimmten Software (in dem Fall Adobe® Director®). Es handelt sich um eine integrierte Entwicklungsumgebung, die es Entwicklern erleichtert, eigene darauf basierende Applikationen zu erstellen.

Prototyp

Mit Hilfe der Datei *CreateStandaloneSnapshot.cmd* werden alle notwendigen dynamischen Bibliotheken (erkennbar an der Extension .dll) und die Datei 4TConsole.exe in das Verzeichnis */Standalone_Build_XXXX* kopiert (siehe Anhang). Ein weiterer Ordner, der für das Logging zuständig ist, wird ebenfalls angelegt.

Daneben gibt es noch eine weitere Variante, die mittels *CreateXtraSnapshot.cmd* das Verzeichnis */Xtra_Build_XXXX* mit denselben DLLs erstellt. Statt der Datei 4TConsole.exe wird jedoch 4T.x32 kopiert. Die Logging Dateien sind ebenfalls wieder enthalten.

Die Prototypen werden fortlaufend anhand ihrer aktuellen Version durchnummeriert und nochmals in den Textdateien *StandaloneBuildNumber.txt* und *XtraBuildNumber.txt* festgehalten. Im Ordner */ConfigExamples* sind sowohl für ARToolKit als auch 4T-Simulator die entsprechenden OpenTracker und 4T Konfigurationsdateien enthalten. Um die beiden Beispiele zu testen, muss man lediglich die jeweiligen Dateien (XML und DTD) in das Standalone Verzeichnis kopieren und gegebenenfalls die standardmäßig gesetzten Einstellungen anpassen.

5.2.2 Installation und Inbetriebnahme des 4T Toolkits

Die Middleware stellt die notwendigen APIs[57] bereit, liest die Daten von der USB Schnittstelle (z.B. Digitalkamera) und ist für ihre Übersetzung in Objekt-Positionskorrelationen verantwortlich. Außerdem werden alle Programmereignisse der 4T API in einer Log Datei aufgezeichnet.

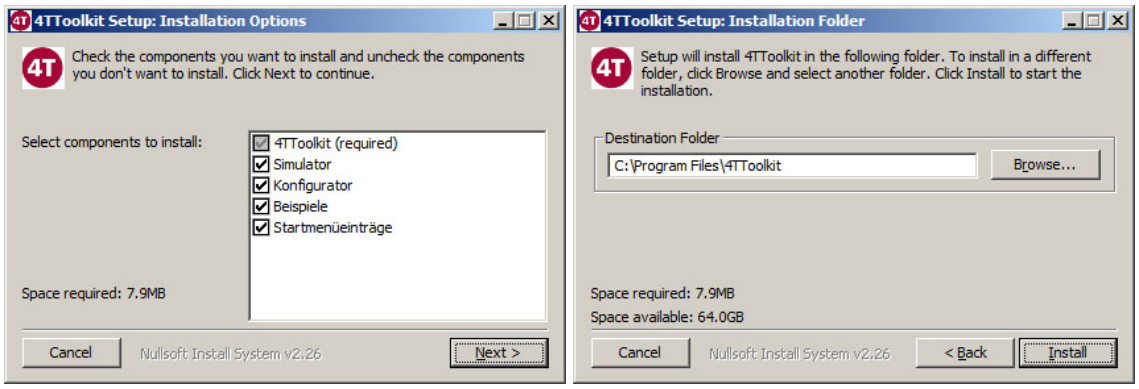


Abbildung 5.6: 4T Toolkit Setup

Das 4T Toolkit (*4TToolkit.exe*) installiert mit dem *4T-Konfigurator* und dem *TTTTSimulator* (bzw. 4T-Simulator) die beiden Hauptwerkzeuge für Setup und

[57] Application Interface (API) stellt die Programmierschnittstelle, die von einem Softwaresystem anderen Programmen einen Systemzugang zur Verfügung stellt.

Debugging des Frameworks. Die ursprüngliche Version lässt nur die Konfiguration des Simulators zu, erst durch die AR-Erweiterung (*4TToolkit_addon.exe*) werden nachträglich auch Einstellungen für das ARToolKit ermöglicht (siehe Abbildung 5.6). Eine reacTIVision Unterstützung ist in der aktuellen Umsetzung (Version 2.0) nicht implementiert.

Entwicklung von 4T Applikationen

Bevor jedoch mit den gewonnenen Informationen des 4T Toolkits konstruktiv gearbeitet werden kann, muss eine geeignete 4T Applikation vorhanden sein bzw. erstellt werden. Zur technischen Umsetzung dient, neben der herkömmlichen Programmierung in Java, auch die aktuelle Version von Adobe® Director®. Mit Hilfe des Multimedia-Authoring-Werkzeugs kann man auch ohne Programmierkenntnisse relativ einfach interaktive Spiele, Webapplikationen oder eLearning Kurse für Websites gestalten und in weiterer Folge publizieren. Neben seiner eigenen Skriptsprache *Lingo*[58] unterstützt die Applikation auch JavaScript[59].

Erfahrene Programmierer können bei der Entwicklung auf Java zurückgreifen. Dabei müssen sie lediglich darauf achten, dass die Datei *4TJNI.jar* im Java Classpath korrekt eingebunden ist. Das Java Archiv wird mit dem 4T Toolkit installiert und befindet sich in dessen Unterverzeichnis /examples/*Java*.

Xtra

Um Designern absolute Handlungsfreiheit zu gewähren, stellt Adobe® seine Entwicklungsumgebung XDK[60] kostenlos zur Verfügung. So lässt sich u.a. Adobe® Director® um fehlende bzw. benötigte Funktionen durch sogenannte *Xtra* Dateien (erkennbar an der Extension .x32) erweitern. Sie ermöglichen dem Authoring-Werkzeug neue Aufgaben über seine ursprüngliche Programmierung hinaus zu erfüllen.

Die notwendigen Erweiterungen für das 4T Framework sind in der Datei *4T.x32* enthalten. Wie alle anderen Xtras muss sie sich bei Programmstart von Adobe® Director® in dessen Unterverzeichnis /Configuration/Xtras/*Scripting* befinden, um als Plugin korrekt erkannt zu werden. Des Weiteren ist die Datei auch bei der Ausführung eines 4T Programms notwendig. Sie muss sich im Ordner */Xtras* befinden, der wiederum im Programmverzeichnis zu liegen hat.

Zusätzlich ist es erforderlich, dass der Ordner */log* (hier werden 4T API Logs gespeichert) zusammen mit der Director-Film Datei (erkennbar an der Extension .dir)

---

[58] Mehr unter http://www.adobe.com/support/director/lingo.html bzw. http://www.lingoworkshop.com/.

[59] JavaScript ist eine Skriptsprache und wird u.a. für das DOM-Scripting in Webbrowsern eingesetzt. Mehr unter http://de.selfhtml.org/javascript/.

[60] Xtra Development Kit (XDK), mehr unter http://www.adobe.com/support/xtras/.

im aktuellen Arbeitsverzeichnis angelegt ist, da das Programm sonst nicht korrekt startet und einfriert. Erst jetzt kann mit der Entwicklung einer neuen 4T Applikation begonnen werden.

4T-Konfigurator – ARToolKit & TTTTSimulator

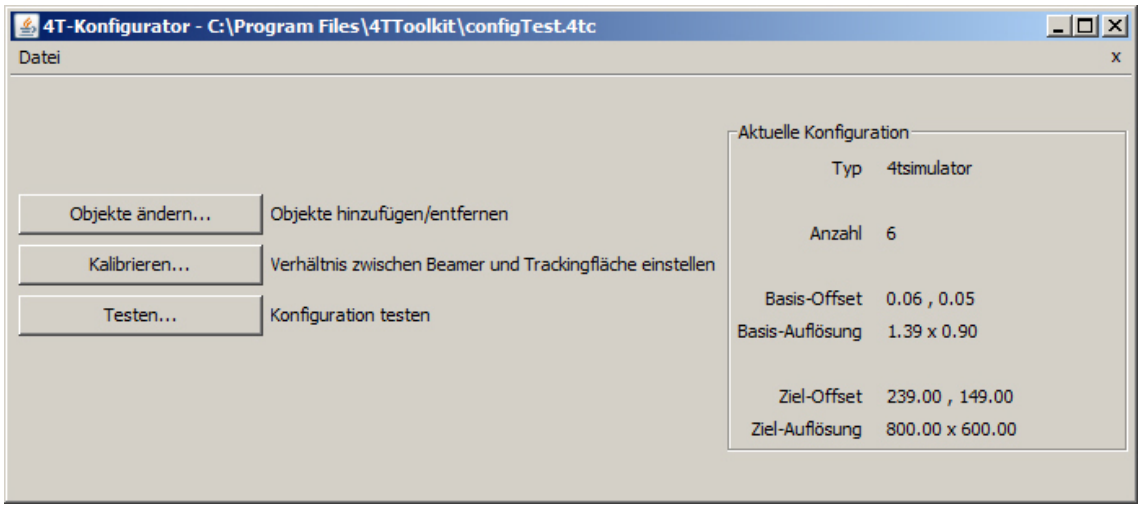


Abbildung 5.7: Hauptfenster des 4T-Konfigurators

Der 4T-Konfigurator legt die Anzahl der Steuerungselemente fest, stellt das Verhältnis zwischen Projektor und Trackingfläche ein und überprüft letztendlich alle vorgenommenen Einstellungen (siehe Abbildung 5.7). Er wird über die Batch-Datei *konfigurator.bat* gestartet.

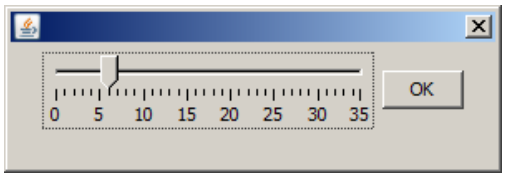


Abbildung 5.8: „Objekt ändern…“

Um nun eine neue Konfiguration zu erstellen, muss der Menüpunkt „*Datei*“ – „*Neu*“ gewählt werden. Hier kann sich der Benutzer zwischen einer ARToolKit- und einer Simulator-Konfiguration entscheiden. Anschließend sind folgende Punkte im Programmfenster nacheinander abzuarbeiten, damit für die jeweilige 4T Anwendung die optimale Konfiguration gefunden wird:

- „*Objekte ändern…*“: Die Anzahl der eingesetzten Steuerungselemente wird durch Verschieben eines Reglers bestimmt. Maximal können 35 Objekte verwendet werden (siehe Abbildung 5.8).
- „*Kalibrieren…*“: Die Bildschirmansicht wechselt zu einem grünen Rechteck mit schwarzem Hintergrund. Dieses GUI repräsentiert die Trackingfläche und ihre

Steuerungsobjekte. Es ist darauf zu achten, dass während der Kalibrierung einer Simulator-Konfiguration der TTTTSimulator bereits zuvor gestartet wurde, da man sonst keine korrekten Resultate erzielt. Des Weiteren sind die Objekte immer an den Trackingflächenrand zu bewegen, um die Begrenzungen des 4T Rückprojektionstisches exakt zu bemessen. Die errechneten Koordinaten können dann noch skaliert werden, indem das grüne Rechteck verschoben wird. Alle Konfigurationseinstellungen lassen sich durch die Taste „*R*“ auf den Ausgangszustand zurücksetzen, mit „*ENTER*“ bestätigen und mittels „*ESC*“ verwerfen.

- „*Testen…*“: Nachdem die 4T Applikation kalibriert ist, kann das Endergebnis anhand eines einfachen Adobe® Director® Programmes überprüft werden.

Der 4T-Konfigurator erlaubt dem Benutzer die fertige Konfiguration über „*Datei*“ – „*Speichern unter…*“ zu speichern (erkennbar an der Extension .4tc), um sie später erneut zu bearbeiten. Durch den darunterliegenden Menüpunkt „*Datei*“ – „*Anwenden…*“ werden die Einstellungen für die Anwendung entsprechend als XML Konfigurationsdateien exportiert. Sie sollten üblicherweise gleich in das Verzeichnis der 4T Applikation zusammen mit der entsprechenden DTD Datei gespielt werden. Da derzeit keine reacTIVision-Konfigurationsmöglichkeit implementiert ist, müssen die entsprechenden Einstellungen (Basis-Offset, Ziel-Offset, etc.) in den jeweiligen XML Dateien per Hand vorgenommen werden. Der genaue Inhalt dieser Dateien wird in Kapitel 5.2.3 ausführlich beschrieben.

Die Einstellungen aus dem 4T-Konfigurator haben im Adobe® Director® keinen Einfluss auf die tatsächlichen Koordinaten. Sie werden vom Authoring-Werkzeug auf die aktuelle Bühnengröße skaliert. Beim Aufruf der Methode *tttt.scaleStage()* wird die Bühne auf ihre ursprüngliche Ausgangsgröße gesetzt, was jedoch keine Rolle für die Positionierung in der 4T Applikation hat. Es ist darauf zu achten, dass man sich im Fullscreen-Mode (an einem Projektor) befindet, da die Bühneneinstellungen im Authoring-Mode verstellt werden.

Neben der Möglichkeit, mit dem Adobe® Director® eine 4T Applikation zu erstellen, gibt es noch eine weitere: die Programmierung in Java. Dabei sollte man sich nicht auf die Koordinatendaten des 4T-Konfigurators verlassen, sondern mit der Methode *getTargetRect()* die maximalen bzw. minimalen Werte abfragen, da sie sich im späteren Verlauf sehr wahrscheinlich ändern (z.B. durch andere Konfigurationen). Die Anwendung funktioniert unabhängig von der gewählten Ausgabegröße und Position. Es empfiehlt sich ein abschließender Test, um eine Rekonfiguration für weitere Trackingmethoden zu erleichtern.

TTTTSimulator (4T-Simulator)

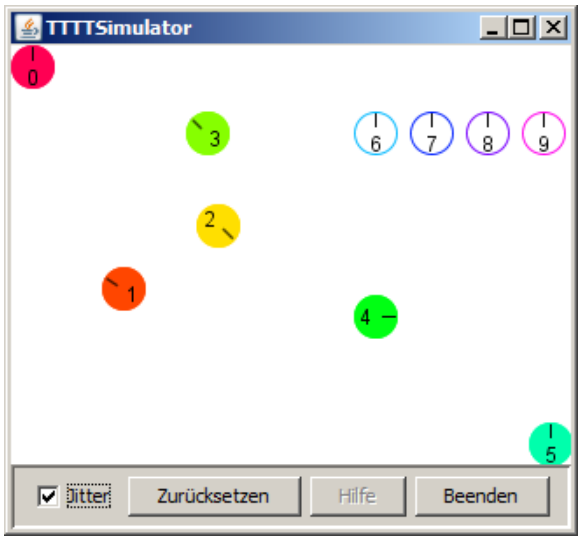


Abbildung 5.9: TTTTSimulator GUI

Der TTTTSimulator (bzw. 4T-Simulator) vervollständigt das 4T Toolkit. Sein GUI ermöglicht eine Benutzung des 4T-Konfigurators bzw. des 4T Toolkits ohne weitere Hardware oder Rückprojektionstisch (siehe Abbildung 5.9).

Per Definition ist die Standard-Anzahl der zu simulierenden Steuerungselemente auf 10 Stück festgelegt. Die Objekte unterscheiden sich farblich und sind durchnummeriert (beginnend mit Null). Theoretisch kann dieser Wert (*objects = 10*) jedoch in der Datei *4Tsimulator.conf* im Ordner /4TToolkit/*simulator* beliebig gewählt werden. Falls die Datei fehlt, wird der ursprüngliche Standardwert für die Simulation herangezogen.

Um den realen Gegebenheiten beim optischen Tracking weitestgehend zu entsprechen, kann der Benutzer auftretende Bildschwankungen während der Bildschirmdarstellung mittels *Jitter*[61] simulieren. Die Steuerungselemente schwingen allerdings nicht sichtbar im Programmfenster des 4T-Simulators.

Mit der linken Maustaste lassen sich die Objekte aktivieren bzw. bewegen. Ein aktiviertes Steuerungselement wird mittels Mausrad gedreht (nach unten -> Drehung im Uhrzeigersinn, nach oben -> Drehung gegen Uhrzeigersinn). Die rechte Maustaste entfernt ein Objekt von der Trackingfläche und es wird farblos. Erneuter Rechtsklick und das Objekt besitzt wieder seine ursprüngliche Farbe und Funktion. Die mittlere Maustaste schaltet die automatische Objektbewegung ein und aus.

[61] Als Jitter (Fluktuation, Schwankung) bezeichnet man ein Taktzittern bei der Übertragung von Digitalsignalen, eine leichte Genauigkeitsschwankung im Übertragungstakt. Allgemeiner ist Jitter in der Übertragungstechnik ein abrupter und unerwünschter Wechsel der Signalcharakteristik. Dies kann sowohl Amplitude als auch Frequenz und Phasenlage betreffen.

5.2.3 XML Konfiguration

4T verwendet zwei XML Konfigurationsdateien. Die Datei *ot.configuration.xml* ist für OpenTracker Einstellungen zuständig, für 4T ist es *ft.configuration.xml*. Beide müssen gemeinsam mit der korrespondierenden DTD Datei *opentracker.dtd* im selben Verzeichnis sein, wo sich auch das auszuführende Programm befindet. Bei einem Standalone Prototypen wäre das z.B. 4Tconsole.exe, bei 4T Xtra die Datei 4TXtra.dll.

Es besteht auch die Möglichkeit, dass man bestimmte Verzeichnisse über die 4T API als Speicherort definiert. Diese Methode ist besonders empfehlenswert, wenn der Benutzer zwischen mehreren verschiedenen Konfigurationen wechseln möchte, wie das z.B. beim 4T-Konfigurator der Fall ist.

Die Datei ot.configuration.xml verwendet dieselbe Syntax wie alle anderen XML Konfigurationsdateien von OpenTracker. Sie wird direkt an die OpenTracker API weitergeleitet. Damit die Knotenstruktur des Rahmensystems (siehe Kapitel 4.2.3) von 4T erkannt und genutzt werden kann, muss jeder *Rückrufknoten* (engl. Callback Node) über "*name*" eine eindeutige Bezeichnung erhalten.

Ein Rückrufknoten entnimmt Daten aus dem Datenflussgraphen einer Applikation. Er wird vor allem von externen Programmen außerhalb des OpenTracker Frameworks genutzt, um Informationen eines bestimmten Knotens zu extrahieren. Der Rückruf-knoten ist ein einfacher Ereignisüberwacher/-erzeuger (engl. EventObserver/ EventGenerator). Er gibt Ereignisse weiter und ruft dabei immer dann eine registrierte Rückruffunktion auf, wenn durch diese Aktion ein neues Ereignis empfangen wird.

Sein Identifikationsattribut „name" ist nun die Verbindung von OpenTracker zu 4T. Es bezieht sich auf das entsprechende "*identifier*" Attribut in der Datei ft.configuration.xml.

4T-Simulator Konfigurationsbeispiel

ot.configuration.xml (4T-Simulator)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE OpenTracker SYSTEM "opentracker.dtd">
<OpenTracker>
	<configuration>
	</configuration>
	<NetworkSource DEF="netobjekt1" number="0"
	multicast-address="224.100.200.101" port="1289"/>
	<NetworkSource DEF="netobjekt2" number="1"
	multicast-address="224.100.200.101" port="1289"/>
	<Callback name="objekt1" DEF="objekt1">
		<EventTransform scale="533.33 600 0">
			<Ref USE="netobjekt1"/>
		</EventTransform>
	</Callback>
	<Callback name="objekt2" DEF="objekt2">
		<EventTransform scale="533.33 600 0">
			<Ref USE="netobjekt2"/>
		</EventTransform>
	</Callback>
</OpenTracker>
```

Tabelle 5.4: OpenTracker Konfigurationsdatei für den 4T-Simulator

In Tabelle 5.4 wird eine OpenTracker XML Konfigurationsdatei für die Verwendung mit dem 4T-Simulator gezeigt. Das Beispiel beinhaltet zwei *Netzwerk-Startknoten* (engl. NetworkSource Node), welche mit dem 4T-Simulator interagieren. Um die ungenutzten Werte der Z-Achse aus den Koordinatendaten zu streichen, werden sie skaliert und nur die entsprechenden X- und Y-Werte umgewandelt. Zum Schluss definiert der Rückrufknoten das „name“- und „DEF“ Attribut, welche die Referenz zum „identifier“ Wert der 4T Konfigurationsdatei herstellen.

ft.configuration.xml (4T-Simulator)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<ft:configuration version="1.2" loglevel="0"
xmlns:ft="http://deco.inso.tuwien.ac.at/TangibleTableToolkit">
	<ft:grid bbxpos="0" bbypos="0" bbwidth="800" bbheight="600"
	tbxpos="0" tbypos="0" tbwidth="800" tbheight="600"
	vendor="INSO" version="1.0" type="4tsimulator">
		<ft:object identifier="objekt1" number="0"
		jitter-suppression="0.0 0.0 0.0">
			<ft:shape type="circle" color="#000000"/>
		</ft:object>
		<ft:object identifier="objekt2" number="1"
		jitter-suppression="0.0 0.0 0.0">
			<ft:shape type="circle" color="#000000"/>
		</ft:object>
	</ft:grid>
</ft:configuration>
```

Tabelle 5.5: 4T Konfigurationsdatei für den 4T-Simulator

Die Tabelle 5.5 beinhaltet die korrespondierende 4T Konfigurationsdatei. Seine Elemente sind wie folgt beschrieben:

- <*ft:configuration*>
  - *version*: Entspricht der aktuellen Versionsnummer der Konfigurationsdatei. Im Moment wird nur Version 1.2 unterstützt, jede ältere verursacht einen Ausnahmefehler im Programm.
  - *loglevel*: Definiert die Protokollierungsebene.
    - *0*: Es wird nichts protokolliert.
    - *1*: Es werden Positionsinformationen protokolliert.
    - *2*: Es werden Positions- und Programmfehlerinformationen protokolliert.
    - *3*: Es wird alles protokolliert (Verbose).
- <*ft:grid*> repräsentiert das 4T Koordinatennetz des Frameworks.
  - *bbxpos, bbypos, bbwidth, bbheight*: Definiert die *Basis-Auflösung* (engl. Base Box). In dieser Einstellung werden die Maximum- und Minimumeingabewerte festgehalten, welche vom Eingabegerät (z.B. Digitalkamera bei optischem Tracking) erfasst wurden. Die Daten werden über den 4T-Konfigurator berechnet und vom Xtra Plugin 4T.x32 verwendet.
  - *tbxpos, tbypos, tbwidth, tbheight*: Definiert die *Ziel-Auflösung* (engl. Target Box). Seine Werte entsprechen denen der Base Box.
  - *vendor*: Entspricht dem Hersteller/Anbieter des Eingabetypen (z.B. „INSO“ für den 4T-Simulator).
  - *version*: Beinhaltet die Versionsnummer des Eingabegerätes.
  - *type*: Beschreibt den Eingabetypen.
    - *4tsimulator*: Die XML Konfiguration ist für den 4T-Simulator Einsatz bestimmt.
    - *artoolkit*: Die XML Konfiguration ist für den ARToolKit Einsatz bestimmt.
    - *reactivision*: Die XML Konfiguration ist für den reacTIVision Einsatz bestimmt.
- <*ft:object*> entspricht einem Objekt im 4T Koordinatennetz.
  - *identifier*: Diese Attribut stellt die Verbindung zu OpenTracker (bzw. seiner XML Konfigurationsdatei ot.configuration.xml) her. Dabei verweist „identifier“ auf die Attribute „name“ und „DEF“ des Rückrufknotens.
  - *number*: Bezeichnet die Nummer zur Kennzeichnung eines 4T Objekts. Jeder positive Zahlenwert inkl. der Null ist gültig.
  - *jitter-suppression*: Erlaubt die Unterdrückung von Bildschwankungen (engl. Jitter-Suppression) bei Steuerungselementen. Die Einstellung macht vor allem bei Verwendung von ARToolKit oder reacTIVision

einen Sinn, wenn vermeintlich stillstehende Objekte keine konstanten Koordinatenwerte über das Eingabegerät zurückliefern. Wird ein zuvor definierter Schwellwert (Gleitkommazahlenwerte für X-, Y- und Z-Achse) überschritten, setzt die Jitter-Suppression ein. Bei Objektänderungen wird er jedoch ignoriert.

- *<ft:shape>* Ein Objekt kann über einen *Formknoten* (engl. Shape Node) verfügen und demzufolge geometrische Figuren darstellen. Diese Erweiterung wird momentan zwar nicht von der 4T Xtra Datei angesteuert, könnte allerdings für zukünftige Veröffentlichungen nützlich sein.
  - *type*: Definiert die verschiedenen geometrischen (2D und 3D) Formen, die für Objekte möglich sind: *circle, rectangle, square, triangle, rhombus, trapeze, polygon, unknown*.
  - *color*: Entspricht der Farbgebung des Objekts in Hexadezimal RGB Farbcodes.

ARToolKit Konfigurationsbeispiel

Nachdem zuvor Konfigurationsbeispiele für den Einsatz am 4T-Simulator präsentiert wurden, werden nun die entsprechenden Einstellungen auch für die unterstützten optischen Trackingverfahren gezeigt. Den Anfang macht das ARToolKit (siehe Tabelle 5.6 und Tabelle 5.7).

ot.configuration.xml (ARToolKit)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE OpenTracker SYSTEM "opentracker.dtd">
<OpenTracker>
	<configuration>
		<ARToolKitConfig videomode="ardata/WDM_camera.xml"
		camera-parameter="ardata/camera_para.dat" treshhold="50"
		framerate="10" pattern-dir="ardata/patterns/"/>
	</configuration>
	<ARToolKitSource DEF="nobjekt1" tag-file="patt.arrow"
	center="0,0" size="50"/>
	<ARToolKitSource DEF="nobjekt2" tag-file="patt.L"
	center="0,0" size="50"/>
	<Callback name="objekt1" DEF="objekt1">
		<EventTransform rotation="1 0 0 0" scale="1 1 0">
			<Ref USE="nobjekt1"/>
		</EventTransform>
	</Callback>
	<Callback name="objekt2" DEF="objekt2">
		<EventTransform rotation="1 0 0 0" scale="1 1 0">
			<Ref USE="nobjekt2"/>
		</EventTransform>
	</Callback>
</OpenTracker>
```

Tabelle 5.6: OpenTracker Konfigurationsdatei für ARToolKit

Im Grunde unterscheiden sich die ot.configuration.xml Dateien für Simulator und optisches Tracking nicht wesentlich von einander. Beim ARToolKit übernimmt der *ARToolKit Startknoten* (engl. ARToolKitSource Node) die Aufgaben des Netzwerk-Startknotens (4T-Simulator). Sein Attribut „*tag-file*“ legt die Referenzmarkierung fest, die dem definierten Steuerungselement zugrunde liegt. Erneut ist es der Rückrufknoten, der für die Kommunikation zwischen den XML Konfigurationsdateien zuständig ist.

ft.configuration.xml (ARToolKit)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<ft:configuration version="1.2" loglevel="0"
xmlns:ft="http://deco.inso.tuwien.ac.at/TangibleTableToolkit">
        <ft:grid bbxpos="0" bbypos="0" bbwidth="800" bbheight="600"
        tbxpos="0" tbypos="0" tbwidth="800" tbheight="600"
        vendor="TUG" version="1.0" type="artoolkit">
                <ft:object identifier="objekt1" number="0"
                jitter-suppression="5.0 5.0 0.0">
                        <ft:shape type="circle" color="#000000"/>
                </ft:object>
                <ft:object identifier="objekt2" number="1"
                jitter-suppression="5.0 5.0 0.0">
                        <ft:shape type="circle" color="#000000"/>
                </ft:object>
        </ft:grid>
</ft:configuration>
```

Tabelle 5.7: 4T Konfigurationsdatei für ARToolKit

Für die ARToolKit- bzw. seine Kamera-Konfiguration ist es zunächst besonders wichtig, dass sowohl die Konfigurationsdatei *WDM_camera.xml* als auch die Kalibrierungsdatei *camera_para.dat* im Ordner 4TToolkit/konfigurator/*ardata* korrekt sind und ihre Pfade in der Datei ot.configuration.xml vorliegen. Für mehr Information dazu ist die OpenTracker Dokumentation heranzuziehen.

reacTIVision Konfigurationsbeispiel

ot.configuration.xml (reacTIVision)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE OpenTracker SYSTEM "opentracker.dtd">
<OpenTracker>
	<configuration>
		<reacTIVisionConfig displaymode="target"/>
	</configuration>
	<reacTIVisionSource DEF="nobjekt1" fiducialId="0"/>
	<reacTIVisionSource DEF="nobjekt2" fiducialId="1"/>
	<Callback name="objekt1" DEF="objekt1">
		<EventTransform scale="533.33 600 0">
			<Ref USE="nobjekt1"/>
		</EventTransform>
	</Callback>
	<Callback name="objekt2" DEF="objekt2">
		<EventTransform scale="533.33 600 0">
			<Ref USE="nobjekt2"/>
		</EventTransform>
	</Callback>
</OpenTracker>
```

Tabelle 5.8: OpenTracker Konfigurationsdatei für reacTIVision

Letztendlich sind in Tabelle 5.8 und Tabelle 5.9 die Konfigurationseinstellungen für das reacTIVision Modul und seinen *reacTIVision-Startknoten* (engl. reacTIVisionSource Node) beschrieben. Ähnlich dem ARToolKit Attribut „tag-file“ übernimmt hier nun „*fiducialId*“ den Verweis auf die Referenzmarkierungen der tangiblen Objekte.

ft.configuration.xml (reacTIVision)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<ft:configuration version="1.2" loglevel="0"
xmlns:ft=" http://deco.inso.tuwien.ac.at/TangibleTableToolkit">
	<ft:grid bbxpos="0" bbypos="0" bbwidth="800" bbheight="600"
	tbxpos="0" tbypos="0" tbwidth="800" tbheight="600"
	vendor="TUG" version="1.0" type="reactivision">
		<ft:object identifier="objekt1" number="0"
		jitter-suppression="5.0 5.0 0.0">
			<ft:shape type="circle" color="#000000"/>
		</ft:object>
		<ft:object identifier="objekt2" number="1"
		jitter-suppression="5.0 5.0 0.0">
			<ft:shape type="circle" color="#000000"/>
		</ft:object>
	</ft:grid>
</ft:configuration>
```

Tabelle 5.9: 4T Konfigurationsdatei für reacTIVision

Den Abschluss dieses Kapitels mach ein kurzer Auszug aus der DTD Datei für das 4T Framework. Hier kann man im Detail lesen, wie die jeweiligen Knoten und ihre Parameter für die jeweiligen Konfigurationen definiert sind (siehe Tabelle 5.10)

opentracker.dtd

```
<?xml version="1.0" encoding="UTF-8"?>
…
>
<!--network receiver elements, acting as sources-->
<!ELEMENT NetworkSourceConfig EMPTY>
<!ELEMENT NetworkSource EMPTY>
<!ATTLIST NetworkSource
  DEF ID #IMPLIED
  number CDATA #REQUIRED
  multicast-address CDATA #REQUIRED
  port CDATA #REQUIRED
>
<!-- video source elements for ARToolKit library-->
<!ELEMENT ARToolKitConfig EMPTY>
<!ATTLIST ARToolKitConfig
  camera-parameter CDATA #REQUIRED
  treshhold CDATA "100"
  framerate CDATA "10"
  videomode CDATA #REQUIRED
  videomode2 CDATA #IMPLIED
  videolinuxmode CDATA #IMPLIED
  videolinuxmode2 CDATA #IMPLIED
  pattern-dir CDATA ""
>
<!ELEMENT ARToolKitSource EMPTY>
<!ATTLIST ARToolKitSource
  DEF ID #IMPLIED
  tag-file CDATA #REQUIRED
  center CDATA #REQUIRED
  size CDATA #REQUIRED
>
<!-- video source elements for ReacTIVision -->
<!ELEMENT ReactivisionConfig EMPTY>
<!ATTLIST ReactivisionConfig
  port CDATA #IMPLIED
  invert CDATA #IMPLIED
  gridfile CDATA #IMPLIED
  treefile CDATA #IMPLIED
  displaymode (source|target|nodisplay|nowindow) #IMPLIED
  fiducialengine (classic|dtouch|standard) #IMPLIED
>
<!ELEMENT ReactivisionSource EMPTY>
<!ATTLIST ReactivisionSource
  DEF ID #IMPLIED
  fiducialId CDATA #REQUIRED
>
…
>
```

Tabelle 5.10: Dokumenttypdefinition für 4T

5.3 Applikation für Endbenutzer (Designer)

Im Folgenden werden die Voraussetzungen für die erfolgreiche Ausführung einer 4T Applikation erklärt. Anschließend wird ein Beispielprogramm kurz vorgestellt und beschrieben.

5.3.1 Voraussetzungen

Bevor man eine Applikation unter 4T starten kann, müssen alle Voraussetzungen stimmen. Im vorangegangenen Kapitel 5.2.1 sind Standalone Prototypen beschrieben worden, deren Programmbibliotheken auch als Grundlage für die eigenständige Applikation dienen. Tabelle 5.11 beschreibt für jede der drei Varianten alle notwendigen Schritte, um eine 4T Applikation erfolgreich zu starten:

4T Simulator	ARToolKit	reacTIVision
4T Applikation erstellen (Adobe® Director® oder Java)		
4T-Simulator starten	-	-
4T-Konfigurator starten		-
Datei – Neu – Simulator-Konfiguration	Datei – Neu – ARToolKit-Konfiguration	-
Objekt ändern… – Kalibrieren… – Testen…		-
-	-	XML Konfigurationsdateien in Editor anpassen
XML Konfigurationsdateien und opentracker.dtd in Programmverzeichnis kopieren		
Standalone Prototyp Programmbibliotheken in Programmverzeichnis kopieren		
-	-	reacTIVision Framework[62] in Programmverzeichnis kopieren
4T Applikation starten		

Tabelle 5.11: Die drei Ausführungsvarianten einer 4T Applikation

Entscheidend dabei ist, ob die Applikation am 4T-Simulator oder unter einem der beiden optischen Trackingverfahren gestartet werden soll. Für das ARToolKit und den

[62] Folgende Dateien müssen auf alle Fälle inkludiert sein: reactivision.exe, DSVL.dll, SDL.dll, reactivision.xml und camera.xml.

Simulator legt der 4T-Konfigurator alle relevanten Details fest, für reacTIVision müssen die XML Konfigurationsdateien (z.B. von einer ARToolKit-Konfiguration) in einem Editor umgeschrieben bzw. optimiert werden.

Der Ordner /log sowie die XML Konfigurationsdateien (ot.configuration.xml, ft.configuration.xml) und die Datei opentracker.dtd müssen sich immer im Verzeichnis der 4T Applikation befinden.

Außerdem müssen die X- und Y-Achse des reacTIVision Frameworks bei jedem Programmstart neu invertiert (Taste „I“) werden. Das liegt an der Anordnung der IR-Kamera und der Spiegel innerhalb des Rückprojektionstisches.

Im Anhang befindet sich als Beispiel für eine korrekt funktionierende 4T Applikation die vollständige Dateienauflistung zu ChemTable, der im folgenden Unterkapitel vorgestellt wird.

5.3.2 Beispiel: Der ChemTable

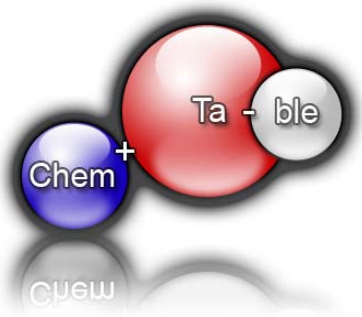


Abbildung 5.10: ChemTable Logo

Im Zuge des HCI Seminars wurden einige Applikationen für das damals noch im Aufbau befindliche 4T Projekt im Sommersemester 2007 entwickelt. So auch der *ChemTable* der Gruppe 1, deren Mitglieder die Studenten Reinhard J. Kurz, Stefan Kuschnigg und Andreas Walch waren. Die 4T Applikation ist primär zur interaktiven Veranschaulichung im Chemieunterricht (HTL, AHS, etc.) konzipiert.

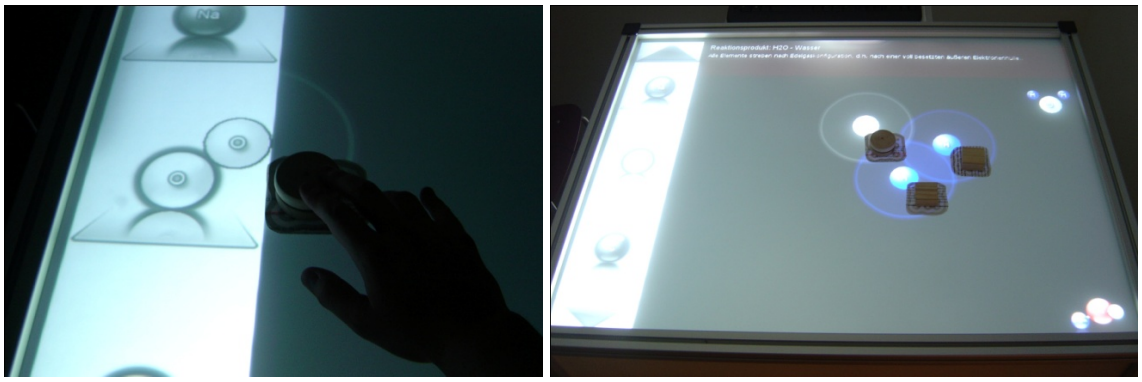


Abbildung 5.11: links: interaktives Auswahlmenu, rechts: Pucks im Reaktionsraum (reacTIVision)

Der ChemTable ist ein spezielles Lernprogramm, das durch Kombination chemischer Elemente alle möglichen Reaktionen nachstellt und gleichzeitig wissenswerte Informationen visuell darstellt. Das Programm unterteilt die Oberfläche des 4T Rückprojektionstisches in zwei Bereiche (siehe Abbildung 5.11): Links befindet sich das *interaktive Auswahlmenu*, rechts der *Reaktionsraum*. Dabei erfolgt die komplette Steuerung über interaktive Steuerungsobjekte, sogenannte *Pucks* (aus Holz, siehe Abbildung 5.12).

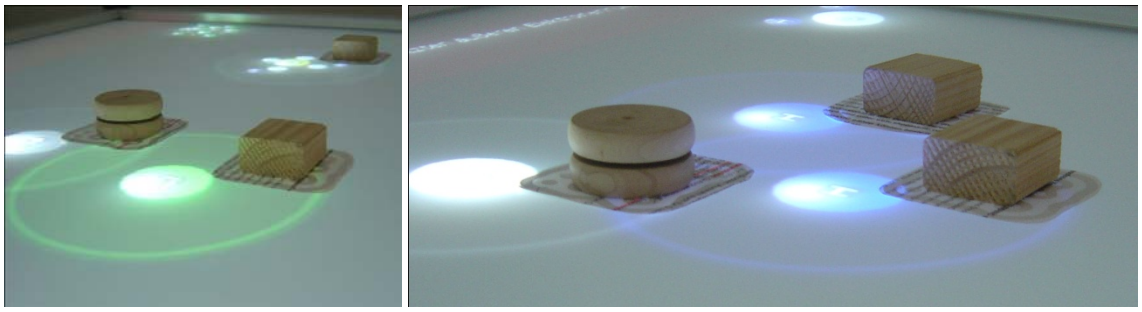


Abbildung 5.12: interaktive Pucks und ihr Aktionsradius (reacTIVision)

Um einem Steuerungsobjekt einen chemischen Stoff zuzuweisen, bewegt der Benutzer den Puck einfach in das Auswahlmenu, welches sich über den Puck mittels Pfeilen vertikal scrollen lässt. Anschließend wird das gewünschte Element anhand seiner Summenformel aktiviert, indem der Benutzer das Steuerungsobjekt auf dessen Summenformel bewegt. Falls man ein anderes chemisches Element für ein virtuelles Experiment heranziehen möchte, muss das Auswahlverfahren wiederholt werden. Der kreisförmige Aktionsradius des Pucks übernimmt das Element, gleichzeitig erscheint im oberen Bereich des Reaktionsraums eine kurze Beschreibung des selbigen.

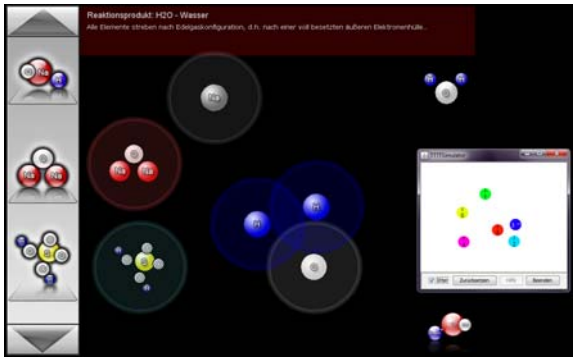


Abbildung 5.13: chemische Reaktionen am ChemTable (4T-Simulator)

Die Aktionsradien mehrerer kompatibler chemischer Stoffe müssen sich überschneiden, um ein Reaktionsprodukt hervorzurufen (siehe Abbildung 5.13). Erneut erfolgt eine Beschreibung des Vorgangs und seines Resultats am oberen Bildrand des Reaktionsraums. Infolgedessen erhält der Benutzer (Schüler) Hintergrundinformationen zum Versuch.

In ersten Tests des ChemTables am 4T Rückprojektionstisch wurde noch mit dem ARToolKit gearbeitet (siehe Abbildung 5.14). Dabei war die Digitalkamera, die das Tracking der Steuerungselemente übernahm, stets über dem Tisch montiert. Der Projektor war bereits unterhalb des Tangible Tabletops befestigt und lieferte das Ergebnis von unten auf die TUI Oberfläche. Das Tracking war allerdings durch die Tischplatte nicht durchführbar, weil die optischen Marker nicht erkannt werden konnten, um ein korrektes Erfassen der ChemTable Pucks zu ermöglichen.

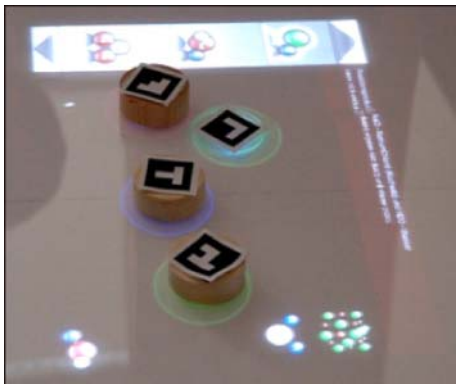


Abbildung 5.14: ChemTable am 4T Rückprojektionstisch (ARToolKit)

Der ChemTable besticht durch seine simple und intuitive Bedienung. Er ist eine gelungene Abwechslung zum trockenen Unterricht aus dem Chemie-Lehrbuch und regt Schüler zu mehr Eigeninitiative bei der Erforschung chemischer Phänomene an. Gleichzeitig ermöglicht der ChemTable wie alle anderen Tangible Tabletops kooperatives Arbeiten in der Gruppe, um Problemstellungen gemeinsam zu lösen. Alles

in allem wurde eine 4T Applikation geschaffen, die durchaus ihre Berechtigung als alternatives Unterrichtsmittel besitzt.

5.4 Erweiterungen und Schlussfolgerungen

In seiner derzeitigen Form entspricht das 4T Rückprojektionstisch den wesentlichen Eigenschaften eines Tangible Tabletops. Die interaktive Oberfläche ermöglicht ein intuitives Zusammenspiel zwischen Benutzer und 4T Applikation. Die geringen Anschaffungskosten, die innovative Bedienerführung sowie die flexible Erstellung und Konfiguration von TUI Applikationen über das 4T Framework sprechen für eine gelungene Umsetzung. Der Einsatz des reacTIVision Frameworks hat im Vergleich zum ARToolKit auch zu deutlich besseren Trackingergebnissen geführt.

Die nun folgenden Unterkapitel zeigen einen kurzen Einblick in mögliche Erweiterungen bzw. liefern Denkanstöße zu eventuellen Verbesserungen des 4T Frameworks und seines Rückprojektionstisches.

5.4.1 Beschaffenheit der TUI Oberfläche

Im Fall von 4T ist die interaktive Oberfläche aus Plexiglas beidseitig sandgestrahlt, damit es zu keinen unerwünschten Lichtreflexionen kommt. Das führt aber leider auch unweigerlich dazu, dass die Referenzmarkierungen von der Kamera leicht verschwommen wahrgenommen werden. Dieser Umstand resultiert nicht nur bei kleineren Mustern in deutlichen Tracking Schwierigkeiten.

Beim reacTable wird nur eine einseitig sandgestrahlte Arbeitsfläche verwendet, daher fällt die Diffusion nicht so stark aus. Außerdem ist der Tisch rund geformt. Die derzeitige 4T Oberfläche und ihre Abmessungen (105 x 80 cm) sind bei den momentanen Hardwarekomponenten zu überdenken, da sie zu keinen optimalen Trackingergebnissen führen.

Der Zugang sollte gleichmäßig möglich sein, ein eckiger Tisch limitiert Interaktion an den Ecken und bevorzugt Benutzer nach ihrer Position am TUI. Die Tendenz in Richtung einer runden (einseitig sandgestrahlten) Tischplatte macht nicht nur hinsichtlich der Ergonomie sondern auch in Bezug auf kooperative Gruppenarbeiten einen Sinn (erleichterter Zugang, keine individuelle Benachteiligung).

5.4.2 Referenzmarkierungen – Größen und Optimierung

Anhand mehrerer Experimente und Erfahrungsberichte benötigen Referenzmarkierungen (Fiducial IDs, reacTIVision Framework) eine Punktdichte von *mindestens 60 Pixel/cm*, um anhand ihrer Muster korrekt erkannt zu werden, da ansonsten vermehrt Tracking Probleme auftreten. Daraus lassen sich die notwendigen Parameter für Referenzmarkierungen am 4T Rückprojektionstisch berechnen:

- 4T Rückprojektionstisch: 105 x 80 cm
- Kameraauflösung: 640 x 480 Pixel
- Verhältnis Kamera/Tisch: 640/105 und 480/80 ergeben ca. 6 Pixel/cm
- Daraus folgt: 60/6 = 10 cm

Die Referenzmarkierungen müssen mindestens 10 cm groß sein, um am 4T Rückprojektionstisch korrekt erkannt zu werden. Je schlechter die Kameraauflösung, desto größer die Fiducial IDs. Eine Verkleinerung der optischen Marker bei gleichbleibender Trackingqualität kann demzufolge nur erreicht werden, wenn eine Digitalkamera mit höherer Auflösung Verwendung findet.

Unabhängig von ihrer Größe verlieren Referenzmarkierungen schnell ihre Lesbarkeit (bzw. Farbe), wenn sie an der Unterseite der Steuerungsobjekte durch Translations- und Rotationsbewegungen abgenutzt werden. Die herkömmliche Variante ist ein einfacher Schwarz-Weiß-Ausdruck auf gewöhnlichem Papier. Zusätzlichen Schutz kann eine Folie über der Referenzmarkierung gewährleisten, um Gebrauchsspuren zu minimieren bzw. gänzlich zu vermeiden. In weiterer Folge könnte man die Strukturen mit IR-reflektierender Farbe drucken. Durch den Einsatz von IR-Strahlern und einer geeigneten IR-Kamera werden hier bereits entscheidende Tracking Vorteile erzielt, die möglicherweise noch verstärkt werden könnten.

5.4.3 Digitalkamera

Erst als eine Digitalkamera mit einer Auflösung von 1024 x 768 Pixel am reacTable zum Einsatz kam, sind auch Fiducial IDs mit einem Durchmesser unter 8 cm korrekt erkannt worden. Zusammen mit dem bereits genannten Rechenmodell für Referenzmarkierungen kann aus dieser Feststellung die Schlussfolgerung gezogen werden, dass neben der Oberflächenbeschaffung der 4T Tischplatte auch die richtige Wahl der Kamera(s) eine große Rolle bei optischen Trackingmethoden spielt.

Die Verwendung mehrerer Digitalkameras, die von unterschiedlichen Positionen gleichzeitig die Oberfläche auf Referenzmarkierungen abtasten, könnte auch bei größeren Tischflächen zu akzeptablen Trackingergebnissen führen. Ein Multi-Kamera Treiber müsste dafür sorgen, dass mehrere Digitalkameras initialisiert, synchronisiert und ihre einzelnen Aufnahmen zu einem Gesamtbild zusammenfügt werden. Doch leider unterstützt reacTIVision 1.4 diese Funktion nicht. Womöglich wird sie in einer der kommenden Versionen des Frameworks umgesetzt.

5.4.4 Multi-Touch Finger Tracking

reacTIVision wurde grundsätzlich für das optische Tracking von Referenzmarkierungen entwickelt. Erst im Nachhinein ist auch eine Multi-Touch Funktionalität implementiert worden.

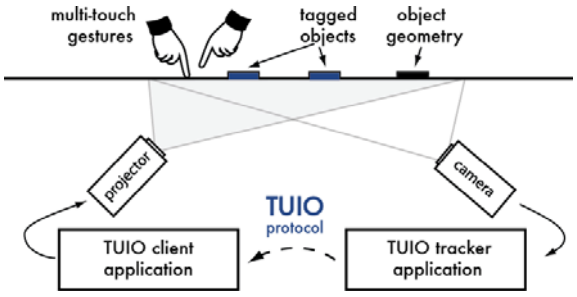


Abbildung 5.15: Fiducial ID und Multi-Touch Finger Tracking

Ein Grund, warum am derzeitigen 4T Rückprojektionstisch das reacTIVision Framework ohne Finger Tracking genutzt wird, liegt in der Tatsache, dass das System bislang noch zu wenig ausgereift ist. Falsch positive Fingerabdrücke werden verstärkt erkannt, doch ihre Anzahl liegt noch im toleranten Bereich. Zukünftige reacTIVision Frameworks und vor allem der Einsatz von TUIO 2.0[63] sollen diesen Makel beim Finger Tracking beheben (siehe Abbildung 5.15).

Außerdem birgt der gleichzeitige Einsatz von Referenzmarkierungen und Multi-Touch Finger Tracking Probleme. Während die Fiducial IDs über eine IR-Kamera im unsichtbaren Bereich des Lichts erkannt werden, muss für die Wahrnehmung der Fingerabrücke bei reacTIVision für ausreichend gleichmäßige Beleuchtung gesorgt sein, um das erforderliche Kontrastverhältnis einzuhalten. Auch weitere Faktoren wie z.B. Helligkeit, Belichtungszeit und Verstärkung des CCD-Ausgangs-Signals (Kontrast-Anhebung) haben wesentlichen Einfluss auf die Tracking Qualität. Somit stellt die Verwendung von Referenzmarkierungen in Kombination mit Multi-Touch Finger Tracking eine besondere Herausforderung an das Setup dar.

5.4.5 Middleware Erweiterungen

Selbst wenn die Entwicklung einer 4T Applikation für einen Designer mit Adobe® Director® kein Hindernis darstellt, so ist ein Verständnis von OpenTracker und dem 4T Framework immanent, wenn das reacTIVision Framework für das Tracking verwendet werden soll. In dem Fall erfolgt die XML Konfiguration nachwievor über einen Texteditor, indem der XML Code einer bereits existierenden Konfigurationsdatei überarbeitet und angepasst wird.

Daher muss als nächstes die Integration des reacTIVision Frameworks in den 4T-Konfigurator oberste Priorität haben. Nicht nur, da es die Einstellungen für reacTIVision enorm vereinfacht, sondern auch, weil im kürzlich vorgestellten TUIO 2.0 Protokoll neue Funktionen integriert sind (wie z.B. Fingerdruck Stärkenmessung, RFID Unterstützung, etc.). Obwohl TUIO 2.0 wohl erst in einer der kommenden reacTIVision

[63] http://www.tuio.org/

Versionen vollständig inkludiert sein wird, sprechen schon jetzt die Vorteile deutlich für eine vollständige Eingliederung des Frameworks und seiner Konfigurations-möglichkeiten. So könnten die neuen Funktionen durchaus bei der Implementierung des elektromagnetischen Trackings in das 4T Projekt hilfreich sein und somit die Basis für eine wirkliche Alternative zu GUIs festigen.

6 Zusammenfassung

Die vorliegende Magisterarbeit befasst sich mit den technischen Details der reacTIVision Erweiterung des *Tangible Table ToolkiT (4T) Frameworks* und der Fertigstellung des dazugehörigen Rückprojektionstisches. Die 4T Anordnung ergibt ein kostengünstiges Tangible User Interface (TUI), das eine konkurrenzfähige Alternative zu kommerziellen Tangible Tabletop Systemen darstellt und der Gestaltung von interaktiven TUI Applikationen dient.

Zum damaligen Projektstart von 4T war das auf 2D Tracking spezialisierte *reacTIVision* Framework noch nicht als Open Source Software verfügbar, daher verwendete man *ARToolKit* als primäre optische Trackingkomponente. Obwohl seine Stärken ganz klar bei der Realisierung von Augmented Reality (AR) Projekten liegen, liefert die Softwarebibliothek auch bei Applikationen mit 2D Trackingfläche zufriedenstellende Ergebnisse.

Im Zuge dieser Magisterarbeit wurde nun reacTIVision 1.4 in das 4T Framework erfolgreich integriert. Es kann vom Entwickler/Benutzer als Alternative zu ARToolKit verwendet werden.

Mit seiner Unterstützung werden weit bessere Resultate beim Tracking erzielt, da reacTIVision extrem robust und wenig fehleranfällig arbeitet. Außerdem ist das Framework Multi-Touch fähig. Dieses Feature kann für spätere 4T Applikationen eingesetzt werden. Am Aufbau der Hardware mussten keine markanten Änderungen vorgenommen werden. Lediglich die Digitalkamera wurde nachjustiert. Des Weiteren war es notwendig, zusätzliche IR-Lampen am 4T Rückprojektionstisch zu installieren, um auch kleinere Referenzmarkierungen (Durchmesser mindestens 10 cm) deutlich zu erkennen.

Normalerweise werden alle notwendigen Daten durch den *4T-Konfigurator* bestimmt und in XML Konfigurationsdateien gespeichert. Dieser legt die Anzahl der Steuerungselemente fest, stellt das Verhältnis von Projektor zu Trackingfläche ein und ermöglicht eine Überprüfung aller vorgenommenen Einstellungen. Zusammen mit dem *4T-Simulator* (Debugging, Test ohne Hardware) bildet er als sogenanntes *4T Toolkit* die grundlegenden Softwarevoraussetzungen, um mittels ARToolKit am 4T Rückprojektionstisch neue Applikationen konfigurieren und testen zu können.

Das 4T Framework setzt bei der Transformation der Trackingdaten auf das skalierbare Rahmensystem *OpenTracker*. Es ist als plattform- und schnittstellenübergreifendes Datenflussnetzwerk implementiert. Der modulare Aufbau des Systems gewährleistet eine einfache Erweiterung und seine flexible XML Konfiguration ist auch bei 4T im

Einsatz. Dadurch konnte z.B. das reacTIVisionModule integriert werden, welches bereits programmiert, aber noch keinen vollständigen Bestandteil von 4T bildete. Infolgedessen sind neue Module bzw. Schnittstellen von OpenTracker auch stets mit 4T kompatibel. Auf diese Weise stehen dem Entwickler zahlreiche Möglichkeiten der Toolkit Erweiterung zur Verfügung. Daneben stellt OpenTracker die gängigsten Treiber für verschiedene Trackingmethoden bereit.

Da es jedoch für das reacTIVision Framework nachwievor keine schrittweise Konfigurationsmöglichkeit gibt, müssen die Parameter in den XML Konfigurationsdateien von Hand geändert bzw. aus einer bestehenden Datei extrahiert und angepasst werden.

Das 4T Toolkit ist primär dazu gedacht, um Designern und Entwicklern den simplen Einstieg in die Welt des Tangible Computing zu ermöglichen. Insofern finden seine Benutzer Unterstützung bei der Entwicklung von sowohl kommerziell als auch nichtkommerziell genutzter Software.

In erster Linie muss man nicht über die hard- und softwaretechnische Zusammensetzung Bescheid wissen (z.B. keine signifikanten Programmierkenntnisse besitzen) und kann sich ausschließlich auf sein kreatives Schaffen konzentrieren. Die Middleware unterstützt den Benutzer dahingehend, indem sie neben der programmierorientierten Gestaltung (durch Java) auch Schnittstellen zu Multimedia-Anwendungen zur Verfügung stellt, die vor allem ihren Fokus verstärkt auf den Authoring-Prozess richten.

Das 4T Projekt erfüllt alle Qualitätsanforderungen, die für engagierte Designer und Entwickler unerlässlich sind:

- *Anwenderfreundlichkeit*: Um mit 4T zu arbeiten sind keine Programmierkenntnisse oder Einschulungen notwendig. Allerdings wird ein gewisses Maß an Fachwissen im Bereich Entwicklung und in der Handhabung mit dem Authoring-Werkzeug Adobe® Director® vorausgesetzt.
- *Korrektheit*: Programminterne Berechnungen liefern immer exakte Resultate. Positionsbestimmung und Objekterfassung könnten sonst keine relevanten Auswertungen liefern.
- *Robustheit*: Falsche Benutzereingaben werden von der Software erkannt bzw. abgearbeitet und stellen kein Stabilitätsrisiko dar. Benutzerhilfestellungen schließen eventuelle Anwendungsfehler bereits im Vorhinein aus.
- *Skalierbarkeit*: Um am System laufend Verbesserungen und neue Features installieren bzw. tauschen zu können, ist das Framework modular zusammengesetzt und erweiterbar. Der objektorientierte Aufbau stellt dies sicher.

- *Feedback*: Das System stellt jederzeit aktuelle Informationen und wichtige Daten für die Applikationsentwicklung zur Verfügung, um dem Benutzer einen kurzen aber exakten Überblick zu verschaffen.

Die Vorteile von 4T gegenüber kommerziellen TUIs liegen auf der Hand: Die Anschaffungskosten für die Hardwarekomponenten sind relativ gering. Im Gegensatz zum klassischen GUI bieten sich vielseitige alternative Möglichkeiten der Benutzereingabe, die nicht nur für Kinder attraktiv und anziehend sein können. TUI Applikationen sind auch für programmierunerfahrene Designer rasch zu entwickeln und zu konfigurieren.

Intuitive TUIs wie 4T zeichnen sich vor allem durch einfache Bedienung, Übersichtlichkeit sowie schnelles Verständnis aller Programmabläufe aus. Diese positiven Eigenschaften beruhen auf dem Prinzip der verkörperten Wahrnehmung und ermöglichen die vollständige Nutzung unserer haptischen Fähigkeiten in solchen Systemen. Zusammen mit der Erkenntnis, dass die Verwendung einer realen Schnittstelle mit physischen Elementen einer reinen Simulation immer vorzuziehen ist, bietet das 4T Projekt die idealen Voraussetzungen und technischen Mittel für die rasche und flexible Gestaltung von tischbasierten TUI Applikationen.

Physische Verkörperungen digitaler Informationen helfen vor allem Kindern, abstrakte Konzepte wortwörtlich „in den Griff“ zu bekommen und so zu verstehen. Gleichzeitig verbirgt sich dahinter auch der wesentliche Grundgedanke von 4T sowie aller tangiblen Computerschnittstellen. Die didaktische Relevanz physischer Manifestationen und Manipulationen wird bei der Verwendung intuitiver Computerschnittstellen sichtbar.

Nichtsdestotrotz gibt es allerdings auch bei 4T noch einige offene Punkte, die Stoff für weitere Entwicklung und Evaluierung bieten, wie z.B. die vollständige Integration des reacTIVision Frameworks in den 4T-Konfigurator. Die Einstellungsoptionen für reacTIVision wären denen bei der Verwendung des ARToolKits endlich ebenbürtig und enorm vereinfacht. Schon jetzt spricht vieles deutlich für eine vollständige Eingliederung des Frameworks und seiner Konfigurationsmöglichkeiten. So könnten die dadurch neu gewonnenen Funktionen durchaus auch bei der Implementierung des elektromagnetischen Trackings in das 4T Projekt hilfreich sein und somit zur Vervollständigung einer äußerst interessanten Alternative zu GUIs beitragen.

Quellenverzeichnis zu Abbildungen und Tabellen

Abbildung 2.1	http://mtg.upf.edu/reactable
Abbildung 2.2	Ishii & Ulmer, 1997
Abbildung 2.3	Ulmer & Ishii, 2001
Abbildung 2.4	Ulmer & Ishii, 2001
Abbildung 2.5	Ishii & Ulmer, 1997
Abbildung 2.6	Ishii & Ulmer, 1997
Abbildung 2.7	Kato et al., 2000
Abbildung 2.8	Ulmer & Ishii, 2001
Abbildung 2.9	http://mtg.upf.edu/reactable
Abbildung 2.10	Bencina et al., 2005
Abbildung 2.11	http://mtg.upf.edu/reactable
Abbildung 2.12	http://mtg.upf.edu/reactable
Abbildung 2.13	http://mtg.upf.edu/reactable
Abbildung 2.14	http://mtg.upf.edu/reactable
Abbildung 2.15	http://mtg.upf.edu/reactable
Abbildung 2.16	http://mtg.upf.edu/reactable
Abbildung 2.17	Kaltenbrunner et al., 2006
Abbildung 2.18	Berry et al., 2006
Abbildung 2.19	http://www.mis.atr.jp/past/kdb/music_table.html
Abbildung 2.20	Berry et al., 2006
Abbildung 2.21	Berry et al., 2006
Abbildung 2.22	Berry et al., 2006
Abbildung 2.23	Berry et al., 2006
Abbildung 2.24	Berry et al., 2006
Abbildung 2.25	http://www.mis.atr.jp/past/kdb/bush_teleg.html
Abbildung 2.26	http://www.jamespatten.com/audiopad
Abbildung 2.27	Patten et al., 2002
Abbildung 2.28	Patten et al., 2001
Abbildung 2.29	Patten et al., 2002
Abbildung 2.30	http://www.jamespatten.com/audiopad
Abbildung 2.31	Patten et al., 2006
Abbildung 2.32	Patten et al., 2006
Abbildung 2.33	Patten et al., 2006
Abbildung 2.34	Patten et al., 2006
Abbildung 3.1	Klemmer et. al, 2006
Abbildung 3.2	Igoe & O'Sullivan, 2004
Abbildung 3.3	http://www.jamespatten.com/audiopad
Abbildung 3.4	Antle, 2009

Abbildung 4.1	http://deco.inso.tuwien.ac.at
Abbildung 4.2	http://deco.inso.tuwien.ac.at
Abbildung 4.3	http://www.microsoft.com/surface/
Abbildung 4.4	DECO/INSO
Abbildung 4.5	DECO/INSO
Abbildung 4.6	http://deco.inso.tuwien.ac.at
Abbildung 4.7	Finkenzeller, 2006
Abbildung 4.8	Finkenzeller, 2006
Abbildung 4.9	Finkenzeller, 2006
Abbildung 4.10	Kato et al., 2000
Abbildung 4.11	http://www.hitl.washington.edu/artoolkit/
Abbildung 4.12	http://www.hitl.washington.edu/artoolkit/
Abbildung 4.13	http://www.hitl.washington.edu/artoolkit/
Abbildung 4.14	http://www.cs.utah.edu/gdc/projects/augmentedreality
Abbildung 4.15	Reitmayr & Schmalstieg, 2005
Abbildung 4.16	Reitmayr & Schmalstieg, 2005
Abbildung 4.17	Reitmayr & Schmalstieg, 2005
Abbildung 4.18	Reitmayr & Schmalstieg, 2005
Abbildung 4.19	Reitmayr & Schmalstieg, 2005
Abbildung 5.1	http://www.mis.atr.jp/past/kdb/music_table.html
Abbildung 5.2	Kaltenbrunner & Bencina, 2007
Abbildung 5.3	Bencina et al., 2005
Abbildung 5.4	Kaltenbrunner & Bencina, 2007
Abbildung 5.5	Bencina et al., 2005
Abbildung 5.6	http://deco.inso.tuwien.ac.at
Abbildung 5.7	http://deco.inso.tuwien.ac.at
Abbildung 5.8	http://deco.inso.tuwien.ac.at
Abbildung 5.9	http://deco.inso.tuwien.ac.at
Abbildung 5.10	Gruppe 1 ChemTable, HCI Seminar 2007
Abbildung 5.11	Gruppe 1 ChemTable, HCI Seminar 2007
Abbildung 5.12	Gruppe 1 ChemTable, HCI Seminar 2007
Abbildung 5.13	Gruppe 1 ChemTable, HCI Seminar 2007
Abbildung 5.14	Gruppe 1 ChemTable, HCI Seminar 2007
Abbildung 5.15	http://www.tuio.org/
Abbildung A	Välkkynen et al., 2003
Abbildung B	Välkkynen et al., 2003
Abbildung C	Lampe & Hinske, 2007

Tabelle 2.1	Jordà et al., 2007
Tabelle 2.2	Patten et al., 2006
Tabelle 5.1	Kaltenbrunner et al., 2005
Tabelle 5.2	Kaltenbrunner et al., 2005
Tabelle 5.3	4T Toolkit
Tabelle 5.4	4T Toolkit
Tabelle 5.5	4T Toolkit
Tabelle 5.6	4T Toolkit
Tabelle 5.7	4T Toolkit
Tabelle 5.8	4T Toolkit
Tabelle 5.9	4T Toolkit
Tabelle 5.10	4T Toolkit
Tabelle 5.11	4T Toolkit

Literaturverzeichnis

Antle, A.N. (2009). *Embodied child computer interaction - Why embodiment matters*, ACM Interactions, March+April Issue (2009), pp. 27-30.

Bencina, R. & Kaltenbrunner, M. (2005). *The Design and Evolution of Fiducials for the reacTIVision System.* Proceedings of the 3rd International Conference on Generative Systems in the Electronic Arts (3rd Iteration 2005). Melbourne, Australia.

Bencina, R., Kaltenbrunner, M. & Jordà, S. (2005). *Improved Topological Fiducial Tracking in the reacTIVision System*. Proceedings of the IEEE International Workshop on Projector-Camera Systems (Procams 2005), San Diego, CA, USA, 2005.

Berry, R., Makino, M., Hikawa, N. & Suzuki, M. (2003). *The Augmented Composer Project: The Music Table*. Proceedings of the Second IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR '03), © 2003 IEEE.

Berry, R., Makino, M., Hikawa, N. & Suzuki, M. (2004). *The Bush Telegraph: Networked Cooperating Music-Making*. Entertainment Computing – ICEC 2004 (ISBN 3-540-22947-7). Proceedings of the Third International Conference, Eindhoven, The Netherlands, September 1-3, 2004.

Berry, R., Makino, M., Hikawa, N., Suzuki, M. & Inoue, N. (2006). *Tunes on the Table*. CM Multimedia Systems Journal, March 2006, Vol. 11, No. 3, Springer Berlin/Heidelberg.

Billinghurst, M. & Kato, H. (2002). *Collaborative Augmented Reality.* Communications of the ACM, July 2002, Vol. 45, No. 7, pp. 64-70.

Billinghurst, M., Poupyrev, I., Kato, H. & May, R. (2000). *Mixed Realities in Shared Space: An Augmented Reality Interface for Collaborative Computing*. Proceedings of ICME 2000, 2000, IEEE. pp. 1641-1644.

Crampton Smith, G. (1995). *The Hand that rocks the Cradle*. I.D., May/June 1995, pp. 60-65.

Djajadiningrat T., Wensveen S., Frens J., Overbeeke K. (2004). *Tangible Products: Redressing the Balance between Appearance and Action*. In: Personal and Ubiquitous Computing, Volume 8 Issue 5, pp. 294-309, 2004.

Dreyfus, H. (2001). *Disembodied Telepresence and the Remoteness of the Real*. In On the Internet, pp. 50-72, Routledge: London, UK, 2001.

Druin, A. & Solomon, C. (1996). *Designing multimedia environments for children: Computers, creativity and kids*. John Wiley & Sons, New York, USA.

Finkenzeller, K. (2006). *RFID Handbuch - Grundlagen und praktische Anwendungen induktiver Funkanlagen, Transponder und kontaktloser Chipkarten*. 4. Auflage, Carl Hanser Verlag München, Deutschland, ISBN 3-446-40398-1, August 2006.

Hornecker, E. (2005). *A Design Theme for Tangible Interaction: Embodied Facilitation*. Proceedings of ECSCW '05, Springer (2005), pp. 23–43.

Igoe, T. & O'Sullivan, D. (2004). *Physical Computing*. Thomson Course Technology, Boston, MA, USA, 2004.

Ishii, H. & Ullmer, B. (1997). *Tangible Bits: Towards seamless Interfaces between People, Bits and Atoms.* Proceedings of Conference on Human Factors in Computing Systems CHI '97, March 1997, ACM Press, pp. 234-241. Atlanta, USA.

Jordà, S., Geiger, G., Alonso, M. & Kaltenbrunner, M. (2007). *The reacTable: Exploring the Synergy between Live Music Performance and Tabletop Tangible Interfaces*. Proceedings of the first international conference on „Tangible and Embedded Interaction" (TEI07). Baton Rouge, Louisiana, USA.

Jordà, S., Kaltenbrunner, M., Geiger, G. & Bencina, R. (2005). *The reacTable.* Proceedings of the International Computer Music Conference (ICMC2005). Barcelona, Spain.

Kaltenbrunner, M. & Bencina, R. (2007). *reacTIVision: A Computer-Vision Framework for table-based Tangible Interaction.* Proceedings of the first international conference on „Tangible and Embedded Interaction" (TEI07). Baton Rouge, Louisiana, USA.

Kaltenbrunner, M., Bovermann, T., Bencina, R. & Costanza, E. (2005). *TUIO – A Protocol for table-based Tangible User Interfaces*. Proceedings of the 6th International Workshop on Gesture in Human-Computer Interaction and Simulation (GW 2005), Vannes (France).

Kaltenbrunner, M., Jordà, S., Geiger, G. & Alonso, M. (2006). *The reacTable: A collaborative Musical Instrument.* Proceedings of the Workshop on „Tangible Interaction in Collaborative Environments“ (TICE), at the 15th International IEEE Workshops on Enabling Technologies (WETICE 2006). Manchester, UK.

Kato, H. & Billinghurst, M. (1999). *Marker Tracking and HMD Calibration for a video-based Augmented Reality conferencing System*. Proceedings of the second International Workshop on Augmented Reality (1999), pp. 85–94. San Francisco, California, USA.

Kato, H., Billinghurst, M., Poupyrev, I., Imamoto, K., Tachibana, K. (2000). *Virtual Object Manipulation on a Table-Top AR Environment.* Proceedings of the International Symposium on Augmented Reality, pp. 111-119, (ISAR 2000). Munich, Germany.

Klemmer, S., Hartmann, B. & Takayama, L. (2006). *How bodies matter: five themes for interaction design*. Proceedings of the 6th conference on Designing Interactive systems, June 26-28, 2006, University Park, PA, USA.

Krispel, U. (2006). *HOWTO use OpenTracker.* Introduction and usage guide to OpenTracker. Seminar paper for Scene Graph Programming, Graz University of Technology, 2006.

Lampe, M. & Hinske, S. (2007). *The Augmented Knight’s Castle – Integrating Pervasive and Mobile Computing Technologies into Traditional Toy Environments*. In: Magerkurth, C., Röcker, C. (eds.): A Reader for Pervasive Gaming Research. Vol. 1, Shaker Verlag, 2007.

Lave, J. & Wenger, E. (1991) *Situated Learning: Legitimate Peripheral Participation*. 1st ed. Learning in Doing: Social, Cognitive, and Computational Perspectives, J.S. Brown, et al. Cambridge: Cambridge Press. 138 pp., 1991.

Markopoulos, P., Read, J., Hoÿsniemi, J. & MacFarlane, S. (2008). *Child computer interaction: advances in methodological research*. Cognition, Technology and Work, v.10 n.2, p.79-81, March 2008.

McLuhan, M. (1994). *Understanding Media: The Extensions of Man*. Cambridge, MA: MIT Presspp. 1994.

Montessori, M. (1949). *Childhood Education*. Henry Regency Company, Illinois, USA, 1949.

Montessori, M. (1964). *The Montessori Method*. New York: Frederick A. Stokes Co. pp. 376, New York, USA, 1964.

Olson, G. M. & Olson, J. S. (2000). *Distance matters*. Human-Computer Interaction 15. pp. 139-78, 2000.

Papert, S. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*. Basic Books, pp. 125, New York, USA, 1980.

Paradiso, J., Hsiao, K., Strickon, J., Lifton, J. & Adler, A. (2000). *Sensor Systems for Interactive Surfaces*. IBM Systems Journal, Volume 39, Nos. 3 & 4, October 2000, pp. 892-914.

Patten, J., Ishii, H., Hines, J. & Pangaro, G. (2001). *Sensetable: A wireless Object Tracking Platform for Tangible User Interfaces*. Proceedings of CHI 2001, March 31 – April 5 2001, ACM Press, ©2001 ACM.

Patten, J., Recht, B. & Ishii, H. (2002). *Audiopad: A tag-based Interface for Musical Performance.* Proceedings of the Conference on New Interface for Musical Expression (NIME 2002). Dublin, Ireland.

Patten, J., Recht, B. & Ishii, H. (2006). *Interaction Techniques for Musical Performance with Tabletop Tangible Interfaces*. Proceedings of the 2006 ACM SIGCHI international Conference on Advances in Computer Entertainment Technology. Hollywood, California, USA.

Pecher, D. & Zwaan, R. A. (2005). *Grounding Cognition: The Role of Perception and Action in Memory, Language and Thinking*. Cambridge University Press, pp. 334, Cambridge, UK, 2005.

Pesce, M. (2000). The Playful World – *How Technology is Transforming our Imagination*, pp. 186–187. Ballantine, New York, 2000.

Reitmayr, G. & Schmalstieg, D. (2001). *OpenTracker – An Open Software Architecture for reconfigurable Tracking based on XML.* Proceedings of IEEE Virtual Reality Conference 2001 (VR 2001).

Reitmayr, G. & Schmalstieg, D. (2005). *OpenTracker – A flexible Software Design for three-dimensional Interaction*. Virtual Reality (2006) 9: pp. 79–92, Springer-Verlag London Limited 2005.

Schmalstieg, D., Fuhrmann, A., Hesina, G., Szalavári, Z., Miguel Encarnação, L., Gervautz, M. & Purgathofer, W. (2000). *The Studierstube Augmented Reality Project.* Technical report, TU Wien, 2000.

Strubel, S. & Zimmermann, S. (2005). *Intuitive Interaktion durch Tangible User Interfaces*. Journal Article, TU Dresden, Fakultät Informatik, Lehrstuhl Multimedia-technik, 2005.

Suchman, L. (1994). *Do Categories Have Politics? The Language/Action Perspective Reconsidered*. Computer Supported Collaborative Work (2). pp. 177-90, 1994.

Ullmer, B. & Ishii, H. (2001). *Emerging Frameworks for Tangible User Interfaces*. IBM Systems Journal, v.39 n.3-4, pp. 915-931, July 2000.

Underkoffler, J. & Ishii, H. (1998). *Illuminating light: an optical design tool with a luminous-tangible interface*. Proceedings of the SIGCHI conference on Human factors in computing systems (1998). ACM Press/Addison-Wesley Publishing Co.: Los Angeles, California, USA, 1998.

Välkkynen, P., Korhonen, I., Plomp, J., Tuomisto, T., Cluitmans, L., Ailisto, H. & Seppä, H. (2003). *A user interaction paradigm for physical browsing and nearobject control based on tags*. Proceedings of Physical Interaction Workshop on Realworld User Interfaces, September 2003.

Wade, N. (1995). *Psychologists in Word and Image*. pp. 173. MIT Press, Cambridge, UK, 1995.

Wagner, D. & Schmalstieg, D. (2007). *ARToolKitPlus for Pose Tracking on Mobile Devices.* Proceedings of the 12th Computer Vision Winter Workshop (CVWW‘07).

Want, R., Fishkin, K. P., Gujar, A. & Harrison, B.L. (1999). *Bridging physical and virtual worlds with electronic tags*. Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit, pp. 370-377, May 15-20, 1999, Pittsburgh, Pennsylvania, USA.

Weiser, M. (1991). *The Computer for the 21st Century*. Scientific American 9, (1991), pp. 933-940.

Wellner, P., Mackay, W. & Gold, R. (1993). *Computer Augmented Environments: Back to the Real World*. Commun. ACM, Vol. 36, No. 7, July 1993.

Wright, M., Freed, A. & Momeni, A. (2003). *OpenSound Control: State of the Art 2003*. Proceedings of the 3rd Conference on New Instruments for Musical Expression (NIME 03), Montreal, Canada, 2003.

Xu, D. (2005). *Tangible User Interface for Children – An Overview* at UCLAN Department of Computing Conference, Preston, UK, 2006.

Anhang

Datenkennzeichnung in RFID Systemen

Välkkynen et al. (2003) legen in ihrer Arbeit über *Physical Browsing* drei Paradigmen für die Datenkennzeichnung durch Tags in RFID Systemen fest: ScanMe, PointMe und TouchMe. Sie sollen bei der Umgebungsentwicklung berücksichtigt werden bzw. dem Designer einer RFID gestützten tangiblen Applikation als unterstützende Beispiele dienen, um die intuitive Benutzerinteraktion mit erweiterten physischen Gegenständen so natürlich wie möglich zu erhalten bzw. zu gestalten.

Abbildung A: ScanMe

Hinter *ScanMe* versteckt sich das Prinzip, dass alle Objekte in einer Umgebung durch Tags gekennzeichnet sind (siehe Abbildung A). Betritt nun der Benutzer das Szenario, so ist es ihm durch ein entsprechendes Lesegerät möglich, jeden physisch etikettierten Gegenstand von Interesse aufzuspüren und auszuwählen, um weitere Informationen, die auf seinem RF-Tag gespeichert sind, abzurufen. Dabei kann es durchaus vorkommen, dass ein RFID gekennzeichnetes Objekt nicht unmittelbar in Sichtweite ist, weil es unter Umständen durch einen weiteren Gegenstand verdeckt wird. Die Abtastung erfasst es trotzdem, solange es sich in Reichweite des Transmitters befindet.

PointMe ist dem vorangegangenen Paradigma recht ähnlich, nur wird hierbei mittels Lesegerät, das über eine Laser- oder IR-Technologie verfügt (z.B. Smartphone), auf den gewünschten erweiterten Gegenstand gezeigt, um ihn direkt auszuwählen (siehe Abbildung B). Die Reaktion mit dem RF-Tag erfolgt sofort und zeigt das Resultat umgehend am Lesegerät an. Die Umsetzung in eine entsprechende Applikation bedarf dabei einer exakten Definition des Einsatzzwecks, um sinnvoll implementiert zu werden.

Abbildung B: PointMe

Dazu zählt z.B. das Projekt *Augmented Knight's Castle*, eine erweiterte Spielzeug-Ritterburg, deren Umgebung (Boden, Bäume, etc.), Szenarien und Spielfiguren mit Antennen und RF-Tags ausgestattet sind (Lampe & Hinske, 2007). Mit einem PDA als Lesegerät lernen Kinder auf spielerische Art und Weise das Mittelalter kennen, wenn sie damit in Reichweite einer Spielfigur oder eines Objekts mit RF-Tag kommen und den Transmitter darauf richten (siehe Abbildung C). Die Spielelemente können aber auch mit phantastischen Geschichten und Märchen verknüpft werden, die in unterschiedlichen Szenen zusammengestellt sind. Auf den Mikrochips sind die jeweiligen Zusatzinformationen gespeichert, die nach dem PointMe oder ScanMe Beispiel gelesen werden.

Abbildung C: Augmented Knight's Castle für Kinder nutzt PointMe/ScanMe

Das letzte Beispielszenario *TouchMe* beschreibt die Objekt- bzw. Tagauswahl durch das Lesegerät anhand (virtueller) Berührung. Hierzu benötigt der Benutzer Standortinformationen zum RF-Tag, welches wieder nicht sichtbar sein muss. Das Objekt selbst kann erst innerhalb einer gewissen Reichweite als physisch etikettierter Gegenstand wahrgenommen werden. Im Gegensatz zu PointMe kann das gewünschte Artefakt exakt ausgewählt werden, da es der Benutzer einfach „berührt". TouchMe tritt bei tisch-

basierten TUIs wohl am ehesten in Erscheinung, weil hier die Steuerungsobjekte stets in Kontakt mit der interaktiven Oberfläche treten.

Obwohl die Paradigmen bei der Konzipierung eines RFID Systems mit TUI äußerst hilfreich sind, haben auch sie ihre Makel. Die Erkennung wird bei ScanMe problematisch, wenn die Assoziierung von virtuellen Objekten und ihren realen Pendants aufgrund der universellen Namensgebung nicht reibungslos verläuft. Die Tags müssen daher über eine Benennung verfügen, die für die Benutzer auch nachvollziehbar ist.

PointMe funktioniert nur, wenn die RF-Etikettierung auch sichtbar und der Bereich zwischen Abtaststrahl und zu überprüfendem Tag frei von blockierenden Störfaktoren ist. Abhängig von der Breite des Strahls erfasst das Lesegerät womöglich mehr als einen Tag bzw. Gegenstand, wenn diese nah beisammen angeordnet sind. Dieses Problem könnte durch ein weiteres Menü am Lesegerät überbrückt werden.

Das dritte Paradigma, TouchMe, wird lediglich durch seine geringe Reichweite eingeschränkt, was jedoch bei TUIs mit tischbasierter Eingabe nicht so stark ins Gewicht fällt.

Programmverzeichnis Standalone Prototyp

Der Standalone[64] Prototyp beinhaltet folgende Dateien:

- 4TConsole.exe
- 4TJNI.dll
- 4TTable.dll
- ACE.dll
- ARJNI.dll
- DSVL.dll
- glut32.dll
- jpeg62.dll
- libARvideo.dll
- libtiff3.dll
- msvcp71d.dll
- msvcr71d.dll
- opentracker.dll
- openVideo.dll
- VideoWrapperd.dll
- xerces-c_2_7.dll
- zlib1.dll

Zusätzlich benötigt man neben den Dateien des Standalone Prototypens noch zwei weitere Bibliotheken von Microsoft Visual Studio 2005, die sich womöglich nicht im Systemordner befinden: *msvcp71.dll* und *msvcr71.dll*.

[64] Hard- bzw. Software, die eigenständig und ohne zusätzliche Elemente funktioniert, bezeichnet man als Standalone.

Programmverzeichnis ChemTable

Im Falle des ChemTables, der im, sieht das Programmverzeichnis (optisches Tracking erfolgt über das reacTIVision Framework) wie folgt aus:

- /Xtras/4T.x32
- /log
- /midi (nicht unbedingt erforderlich)
- 4TConsole.exe (nicht unbedingt erforderlich)
- reactivision.exe
- chemtable.exe
- 4TJNI.dll
- 4TTable.dll
- ACE.dll
- ARJNI.dll
- DSVL.dll
- glut32.dll
- jpeg62.dll
- libARvideo.dll
- libtiff3.dll
- msvcp71d.dll
- msvcr71d.dll
- opentracker.dll
- openVideo.dll
- SDL.dll
- VideoWrapperd.dll
- xerces-c_2_7.dll
- zlib1.dll
- opentracker.dtd
- reactivision.xml
- camera.xml
- ot.configuration.xml
- ft.configuration.xml