# INTERAKTIVE TABLETOP-ANWENDUNGEN ZUM LERNEN PHYSIKALISCHER GRUNDLAGEN

**STUDIENARBEIT**


*von* Ricardo Langner

Student der Computervisualistik, Matrikel: 169038

*Betreuer* Jun.-Prof. Dr.-Ing. Raimund Dachselt

(Otto-von-Guericke Universität Magdeburg)

Prof. Dr. Sheelagh Carpendale

(University of Calgary)

John Brosz

(University of Calgary)




FAKULTÄT FÜR
INFORMATIK

FACULTY OF SCIENCE | UNIVERSITY OF CALGARY

ORT:
Otto-von-Guericke Universität
Fakultät für Informatik
Institut für Simulation und Grafik
Arbeitsgruppe *User Interface & Software Engineering*
Universitätsplatz 2
39106 Magdeburg

BEARBEITUNGSZEITRAUM:
März 2010 – Juli 2010

## ABSTRACT

It is impossible to pass the technological advance and the development of supportive devices, even for the education system. For many schools located in North America it is almost ordinary to buy and use this devices, whereas german schools only just began to do so. They started to provide mobile computers (one per child), to replace regular whiteboard with interactive and touch sensitive screens and to use multi-touch tabletops for cooperative teamwork in classrooms.

This project thesis describes and documents the project *PhysicBox*. *PhysicBox* is a new application for tabletops and comprises results of research about *(a)* the usage of physic simulation (physic engines) in tabletop applications and *(b)* modeling and analysis of interactions techniques for tabletop computing. The *PhysicBox* application is compatible with various tabletop devices and consists of 3 different small games, which represent prototypes for further educational applications.

On the basis of this thesis further projects and research work can be developed, working on exploration and development of educational tabletop applications for the use in future classrooms.



## ZUSAMMENFASSUNG

Der technische Fortschritt und die Entwicklung technischer Hilfsgeräte geht auch an unserem Bildungssystem nicht spurlos vorüber. Was in vielen Schulen Nordamerikas schon fast zum Standard gehört, hält auch in den hiesigen Schulen langsam Einzug. Die Schulen beginnen, vereinzelte Klassen mit Laptops — ein Gerät je Kind wohlgemerkt — auszustatten, die Tafeln durch neue interaktive und berührungsempfindliche Leinwände auszutauschen und für kooperatives Zusammenarbeiten in Gruppen, Multitouch-Tabletops in den Klassenräumen zur Verfügung zu stellen.

In dieser Studienarbeit wird das Projekt *PhysicBox* beschrieben und dokumentiert. Unter Berücksichtigung früherer und aktueller Forschungsarbeiten, in denen *a)* die Verwendung von Physik-Engines in Tabletop-Anwendungen sowie *b)* die Modellierung und Analyse von Interaktionstechniken auf Tabletops im Mittelpunkt standen, wurde in dem Projekt *PhysicBox* eine neue und eigenständige Tabletop-Anwendung entwickelt. Diese Anwendung lässt sich auf verschiedenen Tabletops ausführen und besteht aus drei unterschiedlichen Mini-Spielen, welche als Prototypen für weitere Anwendungen mit didaktischem Hintergrund dienen sollen.

Auf Basis dieser Studienarbeit können weiterer Projekte und Forschungsarbeiten ausgearbeitet werden, in denen es um die Erforschung und die Entwicklung didaktischer Tabletop-Anwendungen für das „Klassenzimmer der Zukunft“ geht.

# INHALTSVERZEICHNIS


1 EINLEITUNG 1
- 1.1 Kontext und Motivation 1
- 1.2 Aufgaben- und Problemstellung 2
- 1.3 Ergebnisse 3
- 1.4 Aufbau dieser Studienarbeit 3

2 VERWANDTE FORSCHUNGSARBEITEN 5
- 2.1 Tabletops und vertikale interaktive Displays 5
- 2.2 Physik-Engines in Tabletop-Anwendungen 6
- 2.3 Interaktionstechniken auf digitalen Tabletops 8
- 2.4 Zusammenfassung 10

3 PHYSICBOX – ANWENDUNGS- UND INTERAKTIONSGESTALTUNG 11
- 3.1 Anforderungen und Ziele 11
- 3.2 Entwicklungsprozess und Ideenfindung 12
- 3.3 Beschreibung des Prototypen 16
  - 3.3.1 Billard auf Glatteis 17
  - 3.3.2 Bulldozer 19
  - 3.3.3 Planeten 20
- 3.4 Interaktionstechniken 23
  - 3.4.1 Realisierte und eingesetzte Techniken 24
  - 3.4.2 Andere Techniken 27
- 3.5 Auswertung 28
  - 3.5.1 Die Anwendungen 28
  - 3.5.2 Probleme, Fragen und weiterführende Themen 30
- 3.6 Zusammenfassung 31

4 IMPLEMENTIERUNG – ANWENDUNG UND FRAMEWORKS 33
- 4.1 Die PhysicBox 33
  - 4.1.1 Aufbau und Paketstruktur 33
  - 4.1.2 Konfiguration 34
  - 4.1.3 Abhängigkeiten der PhysicBox 34
- 4.2 Entwickelte Frameworks 34
  - 4.2.1 jtl – Java Touch Library 34
  - 4.2.2 plight – Java2D Processing Renderer 37
- 4.3 Zusammenfassung 38

5 ZUSAMMENFASSUNG UND AUSBLICK 39
- 5.1 Ergebnisse 39
- 5.2 Erweiterungen und Ausblick 40

A ANHANG 43
- A.1 Standard Konfigurationsdatei 43
- A.2 DVD mit Programm, Quellcode, Dokumentation und Studienarbeit 44

LITERATURVERZEICHNIS 45

# 1 EINLEITUNG

## 1.1 KONTEXT UND MOTIVATION

Nie nutzten die Menschen mehr digitale Geräte in ihrer Freizeit also heute – und die Verkaufszahlen steigen weiter. Wir telefonieren seit Jahren in bester Sprachqualität überall und unterwegs, bezahlen unsere Fahrkarte mit dem Mobiltelefon, prüfen mit dem aktuellsten Laptop unsere E-Mail-Postfächer, schauen auf einem nur zwei Zentimeter dünnen LCD-TV die Fußball-WM oder vergrößern und verkleinern scheinbar magisch mit unseren Fingern Bilder, Landkarten und Websites auf unserem Smartphone. Aber auch in unserem Berufsleben kommen wir nur selten ohne die allgegenwärtigen technischen Geräte aus und – so langsam aber sicher halten sie auch Einzug in unsere Bildungsinstitutionen. An Universitäten sieht man schon längst kaum noch Studenten ohne Laptop unter dem Arm und der Zugang zum Internet ist auf dem gesamten Campus kabellos möglich. Auch an den Schulen geht diese Entwicklung nicht vorüber. An dieser Stelle ist vom „Klassenzimmer der Zukunft“[1,2] die rede. Ein Laptop für jeden Schüler und anstatt einer Kreidetafel eine digitale berührungsempfindliche Leinwand. Dies und digitale Tabletops verkauft die Firma SMART Technologies schon seit Jahren weltweit an Schulen.

Abbildung 1: „Das Klassenzimmer 2.0“: Die Tafel wird durch digitale, berührungssensitive Leinwände und Schreibmappen teils durch Laptops ersetzt. Die neue Technik kann die Schüler und den Lehrer im Unterricht unterstützen und das Lernen und Lehren vereinfachen. Quelle: Staatliches Gymnasium Am Lindenberg Ilmenau (2009)

In dieser Studienarbeit geht es um den Einsatz und die Entwicklung von Anwendungen für diese digitalen Tabletops, die auch in Schulen eingesetzt werden. Das Verhalten von Kindern und Erwachsenen unterscheidet sich maßgeblich. Kinder gehen in der Regel völlig ungehemmt auf neue Dinge zu, probieren herum und

[1] „NOTEBOOK LERNEN“ IM KLASSENZIMMER DER ZUKUNFT, Fakultät für Mathematik, Otto-von-Guericke-Universität Magdeburg, http://www.uni-magdeburg.de/unimagdeburg/home/rpoe/presse_medien/pressemitteilungen/pmi_2010/pressemitteilungen_april_2010/pm_36_2010.html, zuletzt aufgerufen am 14.07.2010;

[2] Das Ende der Kreide-Zeit, Wiesbadener Tagblatt vom 01.07.2010, http://www.wiesbadener-tagblatt.de/region/untertaunus/idstein/9091100.htm, zuletzt aufgerufen am 14.07.2010

entwickeln sich bei jeder ihrer Aktivitäten etwas weiter. Deshalb ist es wichtig zu erforschen, wie die Anwendungen für solche digitalen Tabletops in Klassen aussehen und funktionieren sollten. Auch die Art und Weise wie Kinder Dinge berühren, unterscheidet sich. Die Hände bewegen sich schneller, manchmal auch hektisch und für Multitouch-Geräte wichtig, die Hände und Finger sind kleiner. Kinder haben auch oftmals keine Erfahrung im Umgang mit solchen Multitouch-Geräten. Es gilt die Anforderungen kindlicher Bedienung, Gestaltung und das Verhalten von Kindern im Zusammenhang mit diesen neuen digitalen Tabletops zu erforschen. Diese Studienarbeit und das daraus resultierende Projekt soll die Grundlage für eine Studie im „Klassenzimmer der Zukunft“ legen, indem ein Prototyp für einen digitalen Tabletop entwickelt wird. Dieser Prototyp soll die Möglichkeit bieten, in weiteren Projekten eine Anwendung für eine solche Studie zu modellieren. Weiterhin sollen durch diese Arbeit weiterführende wissenschaftliche Fragen und Themen aufgedeckt und formuliert werden.

Das Thema dieser Studienarbeit ist dem Forschungsbereich *Human-Computer-Interaction* (kurz HCI) zuzuordnen. Während es in diesem Forschungsbereich ganz allgemein um Schnittstellen der Mensch-Computer-Interaktion geht, beschäftigt sich diese Studienarbeit ganz speziell mit den Teilbereichen Tabletop- und Multitouch-Interaktion. Neben der *ACM ITS*, einer Konferenz rund um das Thema Tabletops, gibt es auch viele Forschungsergebnisse auf der *ACM CHI*. Aber auch im Forschungsbereich der Informationsvisualisierung und damit auf der *IEEE VisWeek* gibt es viele Forschungsarbeiten rund um und auf Tabletops und großen Multitouch-Displays.

## 1.2 AUFGABEN- UND PROBLEMSTELLUNG

Das Resultat dieser Studienarbeit und des daran geknüpften Berufspraktikums soll eine veränderbare und erweiterbare Anwendung mit prototypischen Charakter sein. Die Anwendung soll den Fokus „digitale Tabletops im Klassenzimmer“ und „Lerninhalte“ haben. Dazu sollen beispielsweise die folgenden Fragen bearbeitet und beantwortet werden.

- Welche Inhalte mit didaktischen Hintergrund lassen sich in dem begrenzten Zeitraum sinnvoll in eine Tabletop-Anwendung umsetzen?
- Welche Interaktionstechniken scheinen geeignet für den Einsatz im Klassenzimmer?
- Wie kann einem Lehrer die Möglichkeit gegeben werden, digitale Tabletops als sinnvolle Ergänzung zum Unterricht zu nutzen?

Für das Projekt, also das Berufspraktikum und diese Studienarbeit, werden die folgenden Aufgabenstellungen und Bewertungskriterien definiert.

1. *Das Projekt soll ein Einblick und eine kleine Designstudie zum Thema „Physik-Engines in Tabletop-Anwendungen“ darstellen.*
   Dazu ist es notwendig, auf Basis früherer und verwandter Arbeiten, bisherige Forschungsergebnisse zu dokumentieren und in der Implementierung umzusetzen. Dabei sollen mehrere unterschiedliche Mini-Anwendungen mit verschiedenen Interaktionstechniken entstehen.
2. *Das Projekt soll einen Prototypen hervorbringen, der als Grundlage einer nachfolgenden Studie im „Klassenzimmer der Zukunft“ genutzt werden kann.*
   Um dies zu gewährleisten, muss der Prototyp auf möglichst unterschiedlichen Plattformen (Hardware) funktionieren. Der Quellcode des Prototypen

muss in seiner Struktur und Formatierung einen Vorlagencharakter haben, einfach zu warten und vor allem einfach zur modifizieren sein. Die Prototypen-Anwendungen sollen einen didaktischen Hintergrund haben.

3. *Im Projektverlauf sollen weitere Fragen und Themen von wissenschaftlichem Interesse aufgedeckt und formuliert werden*
   Diese Fragen und Themen müssen in dieser Studienarbeit zusammengetragen werden.

## 1.3 ERGEBNISSE

Die Studienarbeit stellt die Anwendung *PhysicBox* für verschiedene Multitouch-Tabletops vor. *PhysicBox* besteht aus drei einzelnen Anwendungen mit didaktischem Hintergrund. Jede dieser Anwendungen lässt sich durch mehrere gleichzeitige Berührungen der Tischoberfläche (Multitouch) bedienen. Dabei wurden ein Reihe verschiedener Interaktionstechniken wie beispielsweise *Flick*, *Proxy Objekte* oder eine erweiterte *Drag-and-Throw*-Technik (Hascoët, 2003) implementiert. Diese Studienarbeit verbindet die Forschungsergebnisse von Wilson u. a. (2008) und Hancock u. a. (2009) sowie verwandten Arbeiten. Sie zeigt interessante Fragen und weitere Themen auf, welche aus der Bearbeitung dieses Projektes hervorgingen.

Der mit dieser Studienarbeit vorgestellte Prototyp wurde in Java entwickelt. Java gewährleistet dabei eine plattformunabhängige Ausführbarkeit der Anwendung. Die aus diesem Projekt hervorgegangenen Bibliotheken *jtl* und *plight* wurden ebenfalls in Java geschrieben. *jtl* zeichnet sich für die Kommunikation mit der eigentlichen Hardware verantwortlich. In diesem Projekt waren das der `Microsoft Surface`, `SMART Table` und `TUIO-Simulator`. *jtl* stellt eine einheitliche Schnittstelle zur Verfügung und ermöglicht es Entwicklern, Anwendungen für verschiedene Tabletops zu entwickeln, ohne diese jeweils individuell anpassen zu müssen.

## 1.4 AUFBAU DIESER STUDIENARBEIT

Die folgenden Passagen beschreiben kurz die einzelnen Teile dieser Studienarbeit.

*Abschnitt 2 - Verwandte Arbeiten*

Der Abschnitt 2 besteht aus 3 Teilen und grenzt darin den zu dieser Arbeit gehörenden Forschungsbereich ab, indem verwandte und vorherige Forschungsarbeiten referenziert werden. Der erste Teil beschäftigt sich mit allgemeiner Literatur zu Tabletops und interaktiven Displays. Der zweite Abschnitt beschreibt Arbeiten, bei denen die Verwendung von (3D)-Physik-Engines in Tabletop-Anwendungen im Fokus stehen. Der dritte Teil schließlich mit Literatur zu verschiedenen Interaktionstechniken auf digitalen Tabletops. Diese drei Teile bilden zusammen die Grundlage für das Verständnis der aktuelle Forschungssituation und die Bearbeitung der Aufgabenstellung dieser Studienarbeit.

*Abschnitt 3 - PhysicBox: Anwendungs- und Interaktionsgestaltung*

Im Abschnitt 3 werden die Anforderungen an den Prototypen und die Anwendungen der PhysicBox im Detail vorgestellt. Es werden Gestaltungsentscheidungen erläutert und die Gründe für die Entwicklung der spezifischen Themenanwendungen dargelegt. Weiterhin werden die verwendeten Interaktionstechniken aufge-

zählt und beschrieben. Details zur Herkunft und Abstammung einiger Techniken werden ebenso verdeutlicht, wie die Referenzen zu anderen wichtigen Forschungsarbeiten. Die abschließende Auswertung geht auf die vorliegenden Erfahrungen und Meinungen zum Prototypen sowie den zuvor erläuterten Anforderungen ein.

*Abschnitt 4 - Implementierung der Anwendung und Frameworks*

Dieser Abschnitt beinhaltet die allgemeine Dokumention des Projektes. Die verwendeten externen Frameworks werden benannt und erläutert. Ebenfalls werden hier die selbstentwickelten und aus diesem Projekt hervorgegangenen Bibliotheken beschrieben und dokumentiert. Dazu zählen Informationen zu der Programmiersprache, der interne Aufbau, die Paketstruktur sowie die wichtigsten Funktionen und Schnittstellen.

*Abschnitt 5 - Zusammenfassung und Ausblick*

Der Abschnitt 5 umfasst die Ergebnisse des Projektes, fasst diese Studienarbeit zusammen und gibt einen Ausblick auf weitere Probleme, Aufgaben und Themen.

# 2 VERWANDTE FORSCHUNGSARBEITEN

In diesem Teil der Studienarbeit werden Forschungsarbeiten und -resultate vorgestellt. Dabei handelt es sich um Arbeiten aus dem Forschungsbereich, der den zugrunde liegenden Rahmen dieser Studienarbeit bestimmt. Dieser Teil besteht aus drei grundlegenden Abschnitten, welche die Grundlage dieser Studienarbeit bilden. Im Abschnitt 2.1 werden allgemeine Arbeiten zum Thema Tabletop und interaktive Displays gelistet. Der Abschnitt 2.2 beinhaltet Forschungsarbeiten zu Tabletop-Anwendungen, in denen eine Physik-Engine das Verhalten von virtuellen Objekten überwacht und umsetzt. Der Abschnitt 2.3 gibt einen Überblick über Arbeiten, in denen Interaktionstechniken auf Tabletops erforscht und analysiert werden.

## 2.1 TABLETOPS UND VERTIKALE INTERAKTIVE DISPLAYS

An Tabletops und vertikalen interaktiven Displays wurde in der Vergangenheit sehr intensiv geforscht (Elrod u. a., 1992; Pedersen u. a., 1993; Streitz u. a., 1994; Ullmer u. Ishii, 1997; Matsushita u. Rekimoto, 1997). In den letzten Jahren gab es einen regelrechten „Hype“ um berührungssensitive Displays als interaktive Leinwände, aber eben auch in Form von Multitouch-Tabletops.

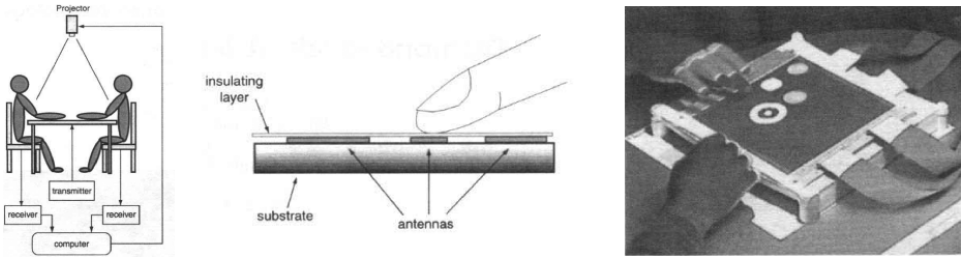



Abbildung 2: DiamondTouch: Mit Hilfe der elektrischen Widerstände werden die Kontaktpunkte (Touchpunkte) erkannt und sogar dem entsprechenden Anwender zugeordnet.

Dietz u. Leigh (2001) stellten das System DiamondTouch vor (Abb. 2). DiamondTouch beschreibt einen berührungssensitiven Mehrnutzer-Tabletop. Das System erkennt durch Messungen den elektrischen Widerstand und damit die Kontaktstellen (Touch) auf dem Tabletop. Statt einem Display wird mit Hilfe eines Projektors das Bild von oben auf den Tabletop projiziert. Eine besondere Eigenschaft des DiamondTouch-Systems, die selbst neun Jahre später von den meisten anderen Systemen nicht vollbracht werden kann, ist die Fähigkeit der Nutzerzuordnung eines Kontaktpunktes, also jedem Kontaktpunkt einem bestimmten Anwender zuordnen zu können.

TouchLight beschreibt ein berührungssensitives interaktives Displays-System, bei dem die Nutzerinteraktion von zwei zueinander versetzten Kameras aufgenommen wird. Durch dieses Prinzip der Stereoskopie kann anschließend die Position eines Objektes bestimmt werden. Da die Kameras in einem fest definierten Abstand zur Leinwand montiert sind, können Berührungspunkte erkannt und

berechnet werden. Mit Hilfe dieses Systems kann auch der unmittelbare Bereich vor der Leinwand erkannt und in das interaktive System einbezogen werden.

Wilson (2005) zeigte PlayAnywhere. Dabei handelt es sich vorrangig um die Idee, bei Bedarf aus jeder ebenen Oberfläche ein interaktives Display zu erschaffen. Das System besteht aus einem Projektor und einer Kamera, welche die Nutzerinteraktion in der Projektion aufnimmt. Ein Computer kann dann die Daten verarbeiten und die Projektion entsprechend der Interaktionen anpassen.

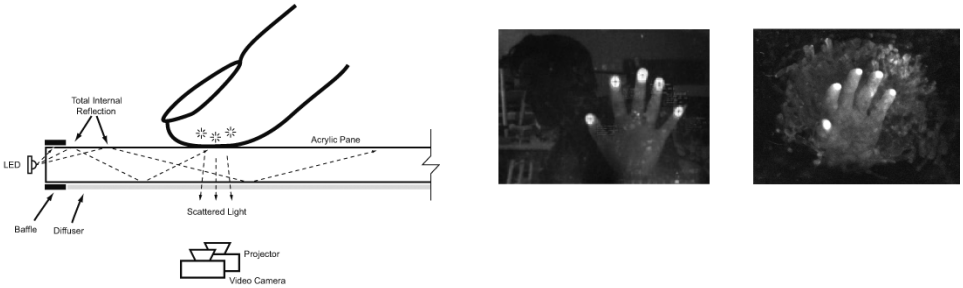



Abbildung 3: FTIR: infrarotes Licht wird in einer transparenten Acrylplatte total reflektiert. Berührt der Finger die Platte, wird das Licht gebrochen.

Han (2005) stellte dann einen Multitouch-Tabletop auf Basis der Totalreflexion vor (Abb. 3). Dabei wird infrarotes Licht in eine transparente Acrylplatte gestrahlt, in der das Licht nicht gebrochen, sondern total reflektiert wird. Wenn nun ein Finger oder ein Objekt anderer Dichte die Platte berührt, wird das Licht an dieser Stelle gebrochen und von dem Finger oder Objekt reflektiert. Nun kann eine Kamera auf der Rückseite der Acylplatte dieses reflektierte Licht aufnehmen. Ein Computer führt dann auf diesen Daten eine einfache Bildverarbeitung durch und kann so die Position der Kontaktpunkte (Touchpunkte) bestimmen. Obwohl die Grundlage der Arbeit nicht neu war (White, 1965; Johnson u. Fryberger, 1972), entwickelte sich eine Gemeinschaft aus Hobbybastlern, Informatikern und Designern und selbst die Open-Source-Gemeinschaft brachte Anleitungen und entsprechende Multitouch-Frameworks hervor (Kaltenbrunner u. Bencina, 2005; Nortd Labs, 2008; University of Augsburg, 2007; NUI Group Community, 2009).

Kommerziell verfügbare Tabletops gibt es seit einiger Zeit von Microsoft und SMART Technologies. Der Surface (Microsoft Corporation, 2007) wird häufig von Firme, Geschäften und der Forschung genutzt. Der SMART Table (SMART Technologies, 2009) hingegen wird eher in Schulen sowie der Forschung eingesetzt. Auch viele regionale Medien- oder Technologieunternehmen stellen Tabletops her.

## 2.2 PHYSIK-ENGINES IN TABLETOP-ANWENDUNGEN

Für lange Zeit waren Physik-Engines, also Systeme, die das Verhalten von virtuellen Objekten überwachen und steuern, nur aus Videospielen bekannt. Die dort eingesetzten Systeme nähern das Verhalten der virtuellen Objekte den Objekten der realen Welt an, unter Berücksichtigung der realen physikalischen Gesetzmässigkeiten auf unserer Erde. Die Spielentwickler entkoppelten mit der Zeit ihre Systeme, um eine Lizenzierung anderer Hersteller und Entwickler zu ermöglichen. Es haben sich eine Reihe kommerzieller sowie frei verfügbarer Physik-Engines entwickelt. Nach Ushakov (2009) sind, gemessen an der Häufigkeit der Verwendung, *PhysX* und *Havok* die mit Abstand erfolgreichsten Physik-Engines auf dem Markt (Abb. 4).

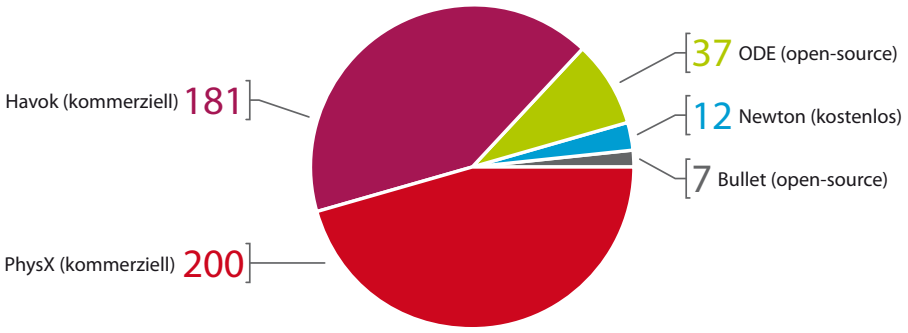



Abbildung 4: Die 5 erfolgreichsten Physik-Engines: Anzahl der Produktveröffentlichungen für die Jahre 2006 bis einschließlich 2009, die eine der Physik-Engines nutzen. Daten nach: Ushakov (2009)

Underkoffler u. Ishii (1999) stellen die Anwendung *Urp* vor, ein System zur virtuellen Stadtplanung. Bei dieser Anwendung werden reale Gittermodelle auf den Tabletop gestellt. Die Modelle werden mit Hilfe von Markierungen an der Unterseite erkannt. Anschließend kann auf dem Display des Tabletops der Schattenwurf oder auch die Auswirkung auf den Windverlauf der platzierten Gebäude simuliert werden. Dafür haben Underkoffler u. Ishii (1999) eine eigene kleine Physik-Engine entwickelt, da sie eigene physikalische Routinen implementiert, teilweise auch als eigenständiges Softwarepaket veröffentlicht haben.

Komplette Physik-Engines für Tabletop-Anwendungen einzusetzen, ist eine relativ junge Idee. Selbst für Desktop-Anwendungen außerhalb des Bereichs der Videospiele kommen sie selten zum Einsatz. Eine Ausnahme bildet das System BumpTop (Agarawala u. Balakrishnan, 2006), welches den virtuellen Schreibtisch (Desktop) in eine 3D-Variante verpackt. Ordner und Dateien werden mittels Physik-Engine als starre Körper (hier flache Boxen) repräsentiert, welche mit verschiedenen Gesten in der virtuellen Welt verschoben, gestapelt, wie ein Fächer auseinander gezogen, geschmissen und gedreht werden können. Über BumpTop wurde oft in IT-Nachrichten berichtet und im April 2010 wurde die BumpTop von Google übernommen (BumpTop, 2010).

Wilson u. a. (2008) zeigt die Verbindung von Tabletops und professionellen Physik-Engines. Kern der Arbeit ist die Frage nach einer geeigneten Methode, die Eingaben der Anwender in die Physik-Engine zu übertragen. Das absolute Positionieren der Objekte in der Physik-Engine kann zu unerwarteten Problemen und Verhalten führen, da das System normalerweise selbstständig in einer Schleife die Objekte modifiziert. Wird in das System von außen und mit unangebrachten Mitteln eingegriffen, kann es passieren, dass Objekte aus der Szene verschwinden, sich durcheinander bewegen oder unerwartet beschleunigen, hin und her springen oder andere *Konfliktzustände* in der Engine hervorgerufen werden. Wilson u. a. (2008) beschreibt und untersucht daher die folgenden Methoden, die Eingaben an die Physik-Engine weiterzugeben.

- *Direkte Kraft*: Eine Kraft wird direkt an einem Punkt auf dem Objekt angesetzt, die Engine berechnet daraus das Verhalten (Bewegungsrichtung, Rotation, Geschwindigkeit, Kollisionen).
- *Virtuelle Verbindungen und Federn*: In der Physik-Engine wird jeder Kontaktpunkt (Touch) mit einer virtuellen Verbindungen oder Federn mit dem Objekt verbunden. Die Eigenschaften (Kraft, Elastizität, Trägheit) dieser Verbindungen und Federn können dann modifiziert werden und die Engine berechnet daraus das Verhalten.

- *Proxy Objekte*: Jeder Kontaktpunkt (Touch) wird in der Physik-Engine durch ein eigenes virtuelles „Proxy"-Objekt repräsentiert. Dieses Objekt kann dann andere Objekte in der Szene beeinflussen, zum Beispiel durch Anstoßen.
- *Partikel und Verformbare 2D/3D Netze*: Stehen auch Informationen zur Form der Kontaktfläche zur Verfügung, zum Beispiel des Fingers, der Hand oder eines Armes, können diese ähnlich den Proxy Objekten in der Physik-Engine durch komplexere Partikel oder 2D/3D Netze nachmodelliert werden (Cao u. a., 2008).

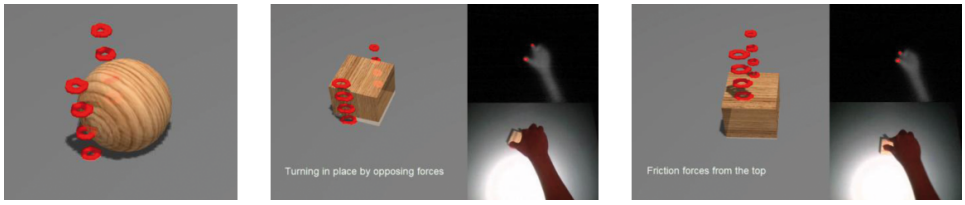



Abbildung 5: Kombination der Methoden *Proxy Objekte* und *Partikel und Verformbare 2D/3D Netze*. Stapel von Proxy Objekten ergeben *Proxy Partikel*. Quelle: Wilson (2009)

Vorteil der Integration von Physik-Engines in die Anwendung ist die einfache Handhabung der Interaktionen, denn die Physik-Engine übernimmt die komplette Berechnung durch Kollisionsmodelle. Zwar kann physikalisches Verhalten auch einfach in dem gewünschten Umfang nachimplementiert werden (Geißler, 1998; Hascoët, 2003; Wu u. Balakrishnan, 2003; Kruger u. a., 2005; Liu u. a., 2006; Reetz u. a., 2006), aber die Physik-Engine unterstützt den Entwickler hier durch eine höhere Flexibilität.

In nachfolgenden Forschungen untersucht Wilson (2009) die Möglichkeiten, *Proxy Objekte* oder *Partikel und Verformbare 2D/3D Netze* in einer Physik-Engine zu nutzen, um mehr Kontrolle über die Objekte in der Szene zu erlangen und diese mit höherer Präzision zu beeinflussen. Dafür werden die oben genannten Methoden *Proxy Objekte* und *Partikel und Verformbare 2D/3D Netze* kombiniert und als *Proxy Partikel* bezeichnet (Abb. 5). Anstatt nur ein Objekt je Kontaktpunkt (Touch) in der Szene zu platzieren, werden einfach mehrere dieser Proxy Objekte übereinander gestapelt.

Auch bei Hancock u. a. (2009) und Hancock u. a. (2010) liegt der Anwendung eine Physik-Engine zu Grunde. Hier wurde ein virtueller Sandkasten als Tabletop-Anwendung entwickelt, in der Kinder mit Hilfe unzähliger Objekte (zum Beispiel Flugzeug, Auto, Schlange, Elefant, Haus, Zaun) Geschichten erzählen können. Mit weiteren abstrakten Interaktionstechniken können der Hintergrund texturiert, Objekte vergrößert und verkleinert und die Szene gespeichert werden.

## 2.3 INTERAKTIONSTECHNIKEN AUF DIGITALEN TABLETOPS

Viele Tabletop-Anwendungen nutzen sogenannte kräftebasierende Interaktionstechniken. Dabei wird in der Regel der Einfluss von realen Kräften wie Reibung, Gravitation oder durch Motoren hervorgerufene Bewegungen auf das virtuelle Objekt und die Interaktion übertragen. Shen u. a. (2003) beschreiben eine Anwendung, bei der Informationen zwischen mehreren Anwendern über einen virtuell rotierenden Tisch („Lazy Susan") ausgetauscht werden können. Hinrichs u. a. (2006) nutzen das Prinzip eines Flusses (Abb. 6). Dabei kann dieser virtuelle Informationsfluss in seiner Form und Flussgeschwindigkeit angepasst und Objekte dem Fluss hinzugefügt und entnommen werden.

Abbildung 6: Ein „Fluss“ aus Informationen auf dem Tabletop. Die Form, Geschwindigkeit und Flussrichtung des Flusses kann durch den Anwender beeinflusst werden. Quelle: Hinrichs u. a. (2006)

Wie virtuelle Objekte bewegt und gedreht werden können, wird ebenfalls in vielen Forschungsarbeiten diskutiert. Viele schlagen dabei vor, das reale physikalische Verhalten von Objekten auch auf die Tabletop-Interaktion zu übertragen (Kruger u. a., 2004, 2005; Liu u. a., 2006; Wilson, 2009). Die von Kruger u. a. (2005) vorgestellte RNT-Technik ermöglicht es dem Anwender, ein Objekt in einer flüssigen Bewegung zu bewegen und zu drehen. Diese kombinierte Translation mit Rotation wird auch im Alltag durchgeführt, wenn beispielsweise ein Dokument über einen Tisch an eine gegenübersitzende Person übergeben werden soll.

Da Tabletops und interaktive Displays oft eine große Fläche besitzen, kann es vorkommen, dass die Anwender mit ihren Armen nicht jede Region des Displays erreichen können. Deshalb wurden verschiedene Techniken entwickelt, um virtuelle Objekte mit Interaktionstechniken auch über weite Distanzen bewegen zu können. Hascoët (2003) stellt mit *Drag-and-Throw* eine Technik vor, bei der das Objekt entgegen der Zielrichtung gezogen und wie mit einem Gummi gespannt wird. Wird dieses Objekt nun losgelassen, schießt es in die Zielrichtung. Eine andere von Hascoët (2003) vorgestellte Technik nennt sich *Push-and-Throw*, funktioniert aber nicht durch zurückziehen des Objektes, sondern durch direktes „wegschieben“. Reetz u. a. (2006) haben später diese Technik um einen Korrekturschritt erweitert, diese Techniken in ihrer Arbeit analysiert und in einer Studie unter anderem die Genauigkeit und die Ausführungszeiten ausgewertet. Geißler (1998) stellt unter anderem eine Technik vor, bei der ein Anwender nacheinander zwei Striche zeichnet, um die Richtung und Stärke einer „Wurf“-Aktion zu beschreiben.

In anderen Arbeiten werden weitere Interaktionstechniken vorgeschlagen und analysiert. Teilweise werden dabei die Konturen der Berührungsflächen von den Händen oder sogar Armen sowie mehrere Kontaktpunkte (Touchpunkte) der Anwender einbezogen (Wu u. Balakrishnan, 2003; Cao u. a., 2008). Frisch u. a. (2009) haben eine Sammlung von Finger- sowie Stift-Gesten auf einem Tabletop erfasst und analysiert. Schwerpunkt war dabei die Erstellung und Modifikation von Node-Link-Diagrammen.

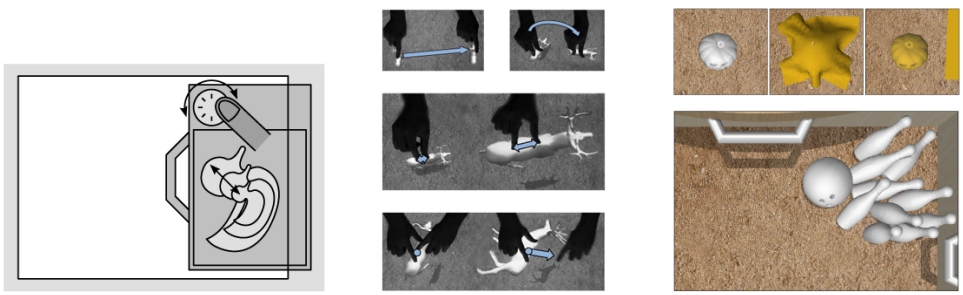


Abbildung 7: Sticky Tool: Kombination der Konzepte *Sticky Fingers*, *Opposable Thumbs* (gegenüberliegende Finger) und *Virtual Tools*. Quelle: Hancock u. a. (2009)

Hancock u. a. (2009) stellen schließlich ein Interaktionsschema vor, welches die Vorteile anderer Techniken verbindet und alle sechs Freiheitsgrade aufweist. Das Schema wurde *Sticky Tools* genannt und ist eine Kombination der Konzepte *Sticky Fingers*, *Opposable Thumbs* (gegenüberliegende Finger) und *Virtual Tools* (Abb. 7). *Sticky Fingers* meint im Grunde die einfache Interaktion (Bewegen, Drehen und Skalieren) mit zwei Fingern. Wird dazu noch ein der Finger der gegenüberliegenden Hand genutzt, sprechen Hancock u. a. (2009) von *Opposable Thumbs*. Das Konzept der *Virtual Tools*, also virtuellen Werkzeugen, beschreibt die Verwendung von virtuellen Objekten als Werkzeug, beispielsweise um ein Objekt mit einem Werkzeug anzustoßen, es zu verkleinern oder die Farbe zu ändern. An dieser Stelle beschreiben Hancock u. a. (2009) auch die Bedeutung von virtuellen Werkzeugen, indem sie ein von Underkoffler u. Ishii (1999) vorgeschlagenes Spektrum von Objektbedeutungen ganz allgemein auf virtuelle Werkzeuge übertragen. Innerhalb dieses Spektrums können virtuelle Werkzeuge als reine Objekte, Adjektive, Substantive, Verben oder als rekonfigurierbares Werkzeug agieren. Beispielsweise agiert ein Werkzeug als Verb, wenn das virtuelle Werkzeug eine bestimmte Aktion ausführt. Ein farbiges Tuch, welches über ein anderes Objekt gelegt wird, um dieses einzufärben, kann als ein solches Werkzeug angesehen werden.

## 2.4 ZUSAMMENFASSUNG

Dieser Teil der Studienarbeit hat verschiedene relevante Forschungsarbeiten aufgelistet und beschrieben. Die hier erwähnten Arbeiten bilden den Rahmen und die Forschungsgrundlage dieser Arbeit. Im ersten Abschnitt wurden allgemeine Arbeiten über Tabletops und interaktive Displays, anschließend Arbeiten zur Nutzung von Physik-Engines in Tabletop-Anwendungen und schließlich Forschungsergebnisse zu verschiedenen Interaktionstechniken aufgezeigt. Als Einstieg wurden die Arbeiten von Wilson u. a. (2008) zu Physik-Engines und Hancock u. a. (2009) zu Interaktionstechniken gewählt.

Der nachfolgende Teil 3 der Studienarbeit widmet sich dem realisierten Prototypen, angefangen bei den allgemeinen Anforderungen, dem Entwicklungsprozess, über die Ergebnisse hin zur Auswertung.

# 3

# PHYSICBOX – ANWENDUNGS- UND INTERAKTIONSGESTALTUNG

Die Entwicklung von Anwendungen durchläuft unterschiedliche Phasen – von der Ideensammlung über Prototypen hin zur finalen Version. Es werden verschiedene Entscheidungen während dieser Phasen getroffen, die das Gesamtbild des Ergebnisses maßgeblich beeinflussen. Am Ende liegt eine finale Projektversion vor, welche in sich geschlossen und vor allem die funktionieren muss. Aber auch weit nach der letzten Phase, wenn alle Entwicklungen abgeschlossen und entwickelte Dinge nur noch angewendet werden, müssen die Prozesse und Ergebnisse gut dokumentiert verfügbar gemacht werden, um möglichen Aufbauarbeiten eine Grundlage zu schaffen.

Dieses Kapitel beschäftigt sich daher mit den *Anforderungen und Zielen* der Anwendungsentwicklung, der Beschreibung des *Entwicklungsprozesses und der Ideenfindung*, der *Beschreibung der einzelnen Anwendungen* in der PhysicBox und der darin *verwendetet und implementierten Interaktionstechniken*. Dabei werden Entwicklungsentscheidungen erläutert und Funktionsweisen erklärt. Ein weiterer wichtiger Bestandteil dieses Kapitels ist die Auswertung. Hier werden die Anforderungen mit dem Ergebnis verglichen und Probleme, Fragen und weiterführende Forschungsthemen aufgezeigt.

## 3.1 ANFORDERUNGEN UND ZIELE

Die erste und wichtigste Anforderung bei der Realisierung der *PhysicBox* ist, entsprechend der Einleitung die Aufgaben- und Problemstellung (siehe 1.2, Seite 2) zu bearbeiten und dadurch Ergebnisse und Lösungsansätze hervorzubringen. Daneben wurden konkrete Anforderungen an den Prototypen formuliert.

1. Der Prototyp soll an der „SMART Table Competition“ auf der Konferenz *ITS 2009* teilnehmen.
2. Der Prototyp soll optisch ansprechend gestaltet werden.
3. Der Prototyp soll plattformunabhängig auf mehreren Tabletops lauffähig sein, mindestens jedoch auf dem `SMART Table` und dem `Microsoft Surface`.
4. Der Prototyp soll flüssig und performant auf oben genannter Hardware ausgeführt werden können (weiche Abläufe, stabile Laufzeit, gute Geschwindigkeit, kein Ruckeln).
5. Im Prototyp sollen die aus der Literatur bekannten Interaktionstechniken *Flick* und *Proxy Objekte* implementiert und verwendet werden.
6. Der Prototyp soll eine weitere und im Vergleich zur *Flick*-Technik präzisere Interaktionstechnik implementieren und verwenden.
7. Der Prototyp soll aus mindestens zwei spielbaren Anwendungen bestehen.

## 3.2 ENTWICKLUNGSPROZESS UND IDEENFINDUNG

Nachdem die ersten Skizzen und Entwürfe gezeichnet waren, wurden diese in kleiner Runde diskutiert. Dabei wurde festgelegt, zu Beginn eine relative schnell zu implementierende sowie einfach anzuwendende Anwendung zu entwickeln. Dahinter stand der Gedanke, sich als erstes mit den technischen Schwierigkeiten auseinanderzusetzen, also benötigte Frameworks und Bibliotheken auszuwählen und Paketstrukturen zu erstellen, um später eine reibungslose Realisierung anderer Anwendungen zu gewährleisten. Von dieser einen einfachen Anwendung sollten dann nach weiteren Gesprächen andere Anwendungen abgeleitet werden.

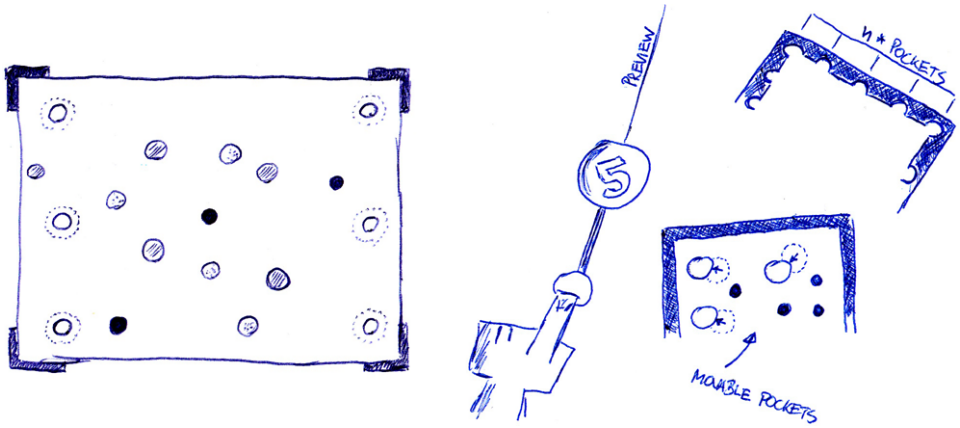



Abbildung 8: Zeichnungen für eine Billard-Anwendung; Idee für Vorschaumodus und eine variable Anzahl der Löcher (Ziele)

Die Auswahl viel auf die Idee „Billard“ (Abb. 8). Ganz klassisch sollten dabei virtuelle Kugeln in Löcher gestoßen werden, im Unterschied zu einem realen Billardspiel jedoch durch alle Mitspieler gleichzeitig. Als Interaktionstechniken kamen für alle Beteiligten nur *Flick*- und *Proxy*-Techniken (siehe 3.4, Seite 23) in Frage, da diese das Spiel durch ihre einfache Handhabung und kurze Ausführungszeiten in einen interessanten Wettbewerb verwandeln würden. Die Idee, eine Art Zwille (Katschi) als „Schusstechnik“ einzusetzen, begeisterte jedoch noch mehr. Bei dieser, später *Slingshot* (siehe 3.4, Seite 23) genannten, Technik, wird der Ball durch zwei Finger ausgewählt, mit einem dritten Finger und einem virtuellen Spanngummi nach hinten gezogen und anschließend losgelassen (Abb. 9). Als ob der Ball in eine Zwille eingespannt wurde, schießt er vorwärts.

Abbildung 9: Erste Beschreibung der Slingshot-Multitouch-Idee durch eine Fotoserie. Deutlich sind die einzelnen Schritte erkennbar – *Objekt umfassen*, *Objekt packen*, *Objekt nach hinten ziehen* und *Objekt loslassen*

Nach kurzer Entwicklungszeit konnte der erste Prototyp fertig gestellt werden. Nach ersten Tests wurden jedoch erste Probleme und Herausforderungen deutlich. Da in der Forschungsgruppe bereits einige Projekte mit der Physik-Engine *PhysX* (NVIDIA GmbH, 2010) von NVIDIA realisiert wurden, sollte die *PhysicBox* ebenfalls auf dieser Engine basieren. *PhysX* ist eine professionelle 3D-Physik-Engine und wird primär in der Entwicklung neuer 3D-Videospiele eingesetzt. *PhysX*,

mit dem riesigen Funktionsumfang, zusammen mit der Java-Portierung *JPhysX* (Kravchik, 2008) war für dieses Projekt von Beginn an überdimensioniert und erschwerte die schnelle Realisierung des Prototypen, da sie eine lange und intensive Einarbeitungszeit erforderte. Deshalb wurde die kleine und auf 2D-spezialisierte Alternative ausgewählt. *JBox2D* (Ewjordan u. a., 2010) begeisterte vor allem durch die einfachen und im Browser ausführbaren Demos und die einfache Handhabung. *JBox2D* ist eine völlig native Java-Portierung von *Box2D* (Catto, 2010), einer der erfolgreichsten open-source Physik-Engines. Binnen Minuten konnten damit simple Anwendungen entwickelt werden, die anschließend die Grundlage eines Billard-Prototypen bildeten.

*Nachträglicher Wechsel von NVIDIA PhysX (3D) zu JBox2D (siehe Logo)*

In weiteren Besprechungen wurde die Integration zusätzlicher Interaktionstechniken (*Flick-* und *Proxy-Objekte*) innerhalb des *Billard*-Prototypen vereinbart. Des Weiteren wurden Ideen für die nächsten Anwendungen gesammelt und diskutiert. Wie oben beschrieben, stand dabei unter anderem der didaktische Hintergrund auf Basis der Rahmenlehrpläne (Government of Alberta, 1996) im Vordergrund. Zu diesem Zeitpunkt wurden auch Gespräche mit einer Grundschullehrerin geführt, um Hinweise und Anmerkungen zu der entwickelten *Billard*-Anwendung, den anderen Anwendungsideen sowie der allgemeinen Projektgrundlage zu bekommen. Im Zusammenhang mit diesen Gesprächen wurden auch die Pläne für eine Feldstudie konkretisiert.

Aus den Skizzen und Vorschlägen wurden später fünf Ideen auf Grund der Eigenschaften Originalität und Zeitaufwand selektiert. Auch die vorgeschlagene Interaktionstechnik für die Idee beeinflusste die Auswahl sowie die anschließend vorgenommene Sortierung einer Entwicklungsreihenfolge. Leider konnten im Verlauf der Entwicklung, auf Grund der dafür sehr knapp bemessenen Zeit, nicht alle fünf Ideen realisiert werden.

*Idee 1: Recycling / Bulldozer*

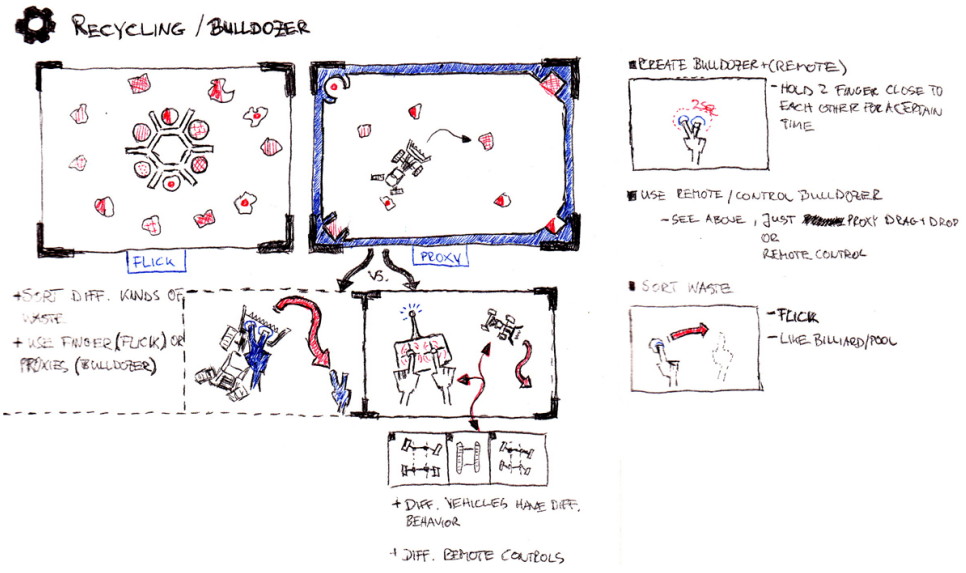



Abbildung 10: Abstrakte Steuerung eines Bulldozers auf einem Recyclinghof.

Die Idee 1 sieht ein Szenario vor, in der jeder Teilnehmer virtuellen Müll recycelt. Abbildung 10 (oben links) zeigt zwei verschiedene Spielkonzepte. Das linke Spielfeld stellt lediglich eine Art modifiziertes *Billard* dar. Die Spieler könnten

durch die *Flick*- oder auch *Slingshot*-Technik (siehe 3.4, Seite 23) verschiedene Müllsorten sortieren. Dabei würden die Zielbehälter kreisförmig in der Spielfeldmitte angeordnet werden.

Das rechte Spielfeld zeigt einen Bulldozer und in den Ecken positionierte Zielbehälter. Diese Idee bringt eine weitere Schwierigkeit und Aufgabe in das Spiel. Jeder Teilnehmer würde einen Bulldozer steuern. Schwerpunkt sollte dabei die Verwendung einer abstrakten, indirekten Steuerung sein, wie zum Beispiel durch eine Fernbedienung. Damit sollte dann der Bulldozer gesteuert und mit ihm der Müll in den entsprechenden Zielbehälter transportiert werden.

In der Abbildung 10 sind auch Skizzen zu verschiedenen Lenkungsarten eines Fahrzeugs, Steuerung des Bulldozers durch eine direkte *RNT*-Technik (siehe 3.4, Seite 23) und einer Geste zum Erzeugen eines Bulldozers zu erkennen.

Eine weitere Idee, die ebenfalls kurz vor der Realisierung stand, drehte sich um die Steuerung des Bulldozers durch ein externes Gerät, wie einem iPod Touch. Dadurch würde Platz auf dem Tabletop gespart und das Konzept der Fernbedienung mehr in den Fokus gerückt werden.

*Idee 2: Planeten und Gravitation*

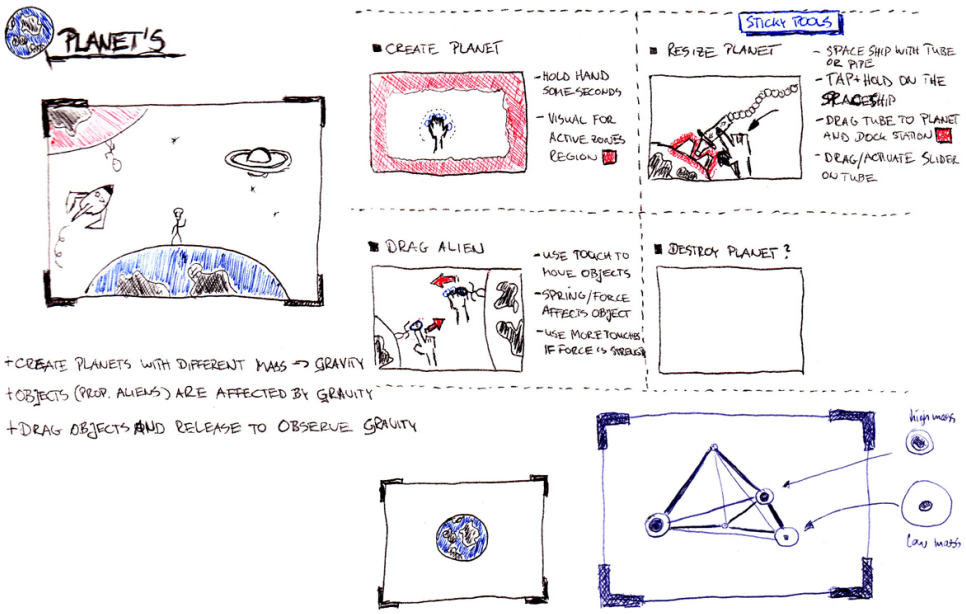



Abbildung 11: Die Planeten wirken durch ihre Größe, Masse und Dichte auf andere Objekte; Ein Raumschiff kann Planeten aufblasen.

Diese Idee beschreibt eine Anwendung, in der Planeten und ihre Gravitation im Vordergrund stehen. In der Grundversion sollten die Spieler in einer 2D-Anwendung Teile von Planeten am Rand des Tabletops sehen (Abb. 11). Auf jedem dieser Planeten sollte eine Art kleines Alien stehen. Je nach Größe und Art des Planeten würde nun die Gravitation das Alien mit einer bestimmten Kraft anziehen. Die Aufgabe besteht darin, durch einfaches *Drag'n'Drop* das Alien vom Planeten zu entfernen. Die Anwendung würde dabei die Kontaktstärke, -fläche und -anzahl berücksichtigen und entsprechend auf das Objekt (Alien) anwenden. Das heißt, bei einem großen Planeten mit sehr starker Gravitation wäre es nötig, auf das Objekt mit mehr Kraft, hier durch die Touchstärke, -fläche und -anzahl beeinflussbar, einzuwirken. Bei einem kleinen Planeten mit geringer Gravitation

würde es hingegen ausreichen, das Objekt mit einem Finger und leichtem Druck zu bewegen.

Die Idee sah vor, ein kleines Raumschiff mit einer Art Aufpump- und Absaugvorrichtung im virtuellen All fliegen zu lassen, mit dem die Spieler Planeten vergrößern und verkleinern könnten, um so die Gravitation zu beeinflussen. Neue Planeten würden durch einfache Kreis-Gesten oder lange handflächengroße Berührungen (Touches) erzeugt werden können.

Abbildung 11 (unten) zeigt auch Skizzen für eine Anwendung ohne Aliens und Raumschiff. Dabei würden sich Planeten durch den Raum bewegen und gegenseitig beeinflussen. Abhängig von Größe und Dichte würden einige eine hohe, andere eine geringe Gravitation besitzen und so auf die Laufbahnen anderer Planeten einwirken. Die Planeten könnten sich auch um eine Sonne bewegen und die Spieler können dann deren Bahn modifizieren und dabei zuschauen, wie sich das gesamte System durch kleine Änderungen entwickelt.

*Idee 3: Pflanzenwelt*


Abbildung 12: Welcher Region muss die Pflanze zugeordnet werden? Benötigt diese Pflanze viel Licht, braucht sie viel Wasser?

Bei dieser Anwendung würden die Teilnehmer unterschiedliche Pflanzen verschiedenen Klimaregionen zuordnen müssen (Abb. 12). Abhängig von der Region und dem „Wohlbefinden" der Pflanze würde diese wachsen und gedeihen – oder eingehen. Die Umgebung würde den Spielern verschiedene Regionen, einen Baum der Schatten wirft, einen Fluss zum Bewässern und eine Sammlung von Pflanzen zur Verfügung stellen. Die Spieler müssten die Pflanzen durch einfaches *Drag'n'Drop* verschieben und ausgehend vom Fluss die *Flick*- oder *Slingshot*-Technik zum Bewässern nutzen.

In den verschiedenen Szenarien könnten die Schüler so ihren Garten gestalten und pflegen. Ein Informationssystem würde detaillierte Daten und Hinweise zu den Pflanzen liefern und so den Forschung- und Lerncharakter unterstreichen.

*Idee 4: Murmelbahn*

Bei dieser Anwendung geht es nicht nur um das Beschleunigen eines Objektes wie beim Billard. Viel mehr würden die Teilnehmer die virtuelle Umgebung anpassen, um den Weg für das Objekt vorzubereiten. So entwerfen die Spieler eine Art „Murmelbahn" (Abb. 13). Dabei können sie Objekte (Hindernisse) platzieren, rotieren und skalieren. Der virtuelle Gummiball wird am Ende durch die *Slingshot*-Technik beschleunigt und auf Kurs gebracht.

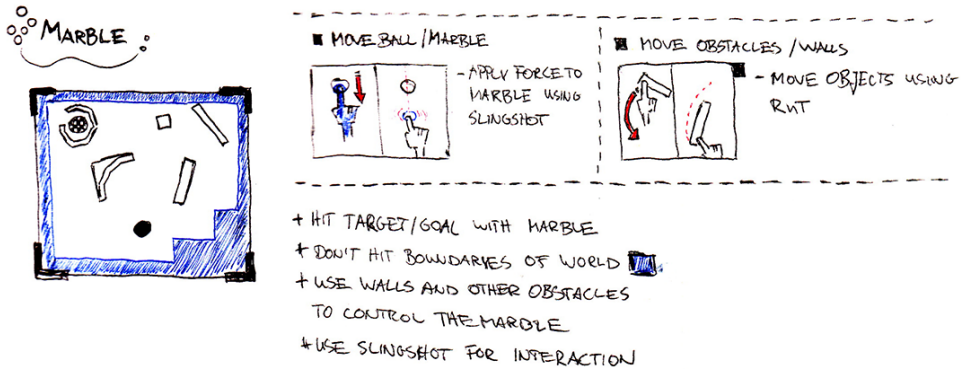



Abbildung 13: Durch den Bau und die Anpassung der Umgebung soll der Gummiball ins Ziel gebracht werden.

Der Ball könnte auch durch eine Lichtquelle, zum Beispiel durch eine Energiesparlampe oder einen Laser, die Hindernisse durch Glas oder einen Spiegel ausgetauscht werden. Eine derartige Anwendung wurde später im „SMART Table"-Wettbewerb der Konferenz *ITS 2009* in Banff, Kanada durch ein Studententeam vorgestellt.

*Idee 5: Straßenkreuzung*

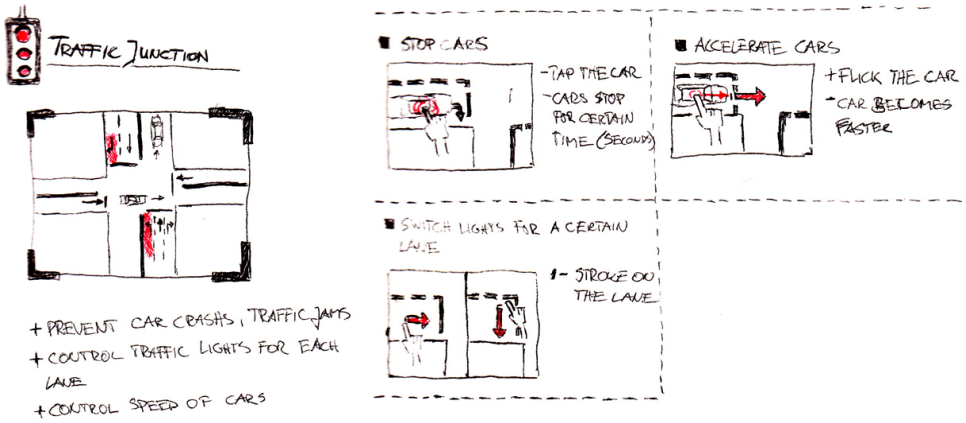



Abbildung 14: Durch reaktionsschnelles Eingreifen Unfälle und Staus verhindern – und so für Sicherheit an der Kreuzung sorgen.

Inspiriert durch die Anwendung „Traffic Rush" (Swedish Game Development AB, 2009) würden die Teilnehmer bei diesem Spiel für die Sicherheit an einer oder mehrerer Straßenkreuzungen sorgen müssen (Abb. 14). Dabei ginge es um die Vermeidung vom Unfällen und Staus. Die Spielern können dabei verschiedene Interaktionsmöglichkeiten nutzen. Ein Auto einfach zu berühren, würde dessen Hupe betätigen und andere Autos wären gewarnt. Einzelne Farbahnen und Ampeln ließen sich durch „Kreuz"-Gesten sperren und freigeben. Autos könnten ebenfalls durch die *Flick*-Technik beschleunigt oder gebremst werden.

## 3.3 BESCHREIBUNG DES PROTOTYPEN

Der Prototyp der *PhysicBox* enthält drei verschiedene Anwendungen. Zu den Anwendungen können Einstellungen in einer zentralen Konfigurationsdatei vor-

genommen werden (siehe 4.1.2, Seite 34). Als eine Art Hauptmenü der gesamten Applikation dient eine kleine Anwendung mit einem Fließband (Abb. 15). Auf dem Fließband bewegen sich in Form von Objekten die einzelnen Spiele. Um ein Spiel zu starten, muss das entsprechende Objekt vom Fließband in einen der beiden Startbereiche verschoben werden.


Abbildung 15: Im Hauptmenü bewegen sich die einzelnen Anwendungen als Objekte auf einem Fließband. Diese müssen in einen Startbereich gezogen werden, um die entsprechende Anwendung zu starten.

Bei den drei Anwendungen handelt es sich um kleine Spiele. Ein „Spiel ist nicht nur in sich vielfältig und interessant, es enthält auch ein beträchtliches Potential an Möglichkeiten, die sich eignen, Kinder gezielt zu fördern, ihre spielerischen Kompetenzen zu steigern, Entwicklungsdefizite auszugleichen, Konflikte konstruktiv zu bewältigen und vieles mehr. Kinder entwickeln im Spiel und durch das Spiel verschiedenste Handlungsweisen viel grundlegender und intensiver als sonst. Vergleichbares gilt für das kindliche Lernen. Es vollzieht sich spielerisch leichter und – was man schon frühzeitig bemerkt hat – effektiver.“ (Mogel, 1994, Kap. 4.2, S. 135, Abs. 1).

In den folgenden Abschnitten werden nun die drei Anwendungen beschrieben.

### 3.3.1 *Billard auf Glatteis*

*Eine Verbindung von Billard und Curling* wäre eine gute Beschreibung dieser Anwendung (Abb. 16). Das Ziel ist es, wie beim Billard, alle Bälle so schnell wie nur möglich in die Löcher zu schießen. Dabei stellt die Anwendung zwei Interaktionsmodi und insgesamt 5 Interaktionstechniken zur Verfügung. Die Kugeln sind eigentlich Steine und verhalten sich wie beim Curling. Der Reibungsfaktor zwischen dem Spielstein und Boden ist also sehr gering. Daraus resultieren hohen Geschwindigkeiten, langen Phasen der Geschwindigkeitsverringerung und auch starken Rotationen der Steine.

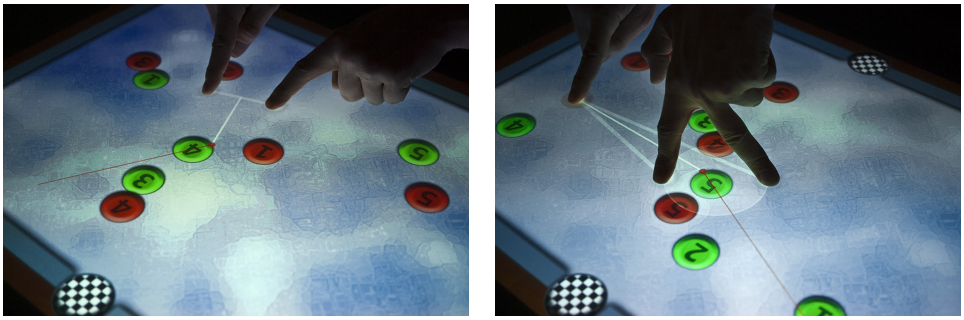


Abbildung 16: Billard auf Eis unterstützt verschiedene Interaktionstechniken

Bei all diesen Eigenschaften können dennoch alle Mitspieler gleichzeitig agieren. Die Herausforderung liegt darin, die Bälle im richtigen Moment perfekt in eine Richtung mit entsprechender Kraft zu schießen.

Von der Planung bis zur Realisierung sollte die Anwendung dabei den folgenden Anforderungen genügen:

- Kinder mit Computern und Multitouch-Technologien *in Berührung* bringen,
- Kindern mittels einfachen Bedienkonzepten die Möglichkeit geben, mit virtuellen Objekten zu interagieren und
- Kindern einige grundlegende physikalische Prinzipien zu vermitteln, wie zum Beispiel Beschleunigung, Kollision und Reibung.

Das Spiel verfügt über zwei verschiedene Interaktionsmodi. Jeder Modus stellt ein eigenes Bedienkonzept zur Verfügung.

*Der Spiel- und Spaßmodus*

Dieser Modus ermöglicht eine schnelle und direkte Interaktion mit den Steinen. So können Kinder auf einfache Art und Weise ein Multitouch-Spiel spielen, ohne sich über physikalische Prinzipien klar zu sein. Alle Eingaben werden direkt und ohne spürbare Verzögerung umgesetzt und beeinflussen das Spielgeschehen sofort. In diesem Modus werden jeweils alle Touch-Eingaben einer *Flick*- oder *Proxy*-Interaktionstechnik (siehe 3.4, Seite 23) zugeordnet. Dabei entscheidet allein die Lage, die absoluten Koordinaten des Kontaktpunktes die Zuordnung.

Liegt ein Kontaktpunkt zum Zeitpunkt des ersten Erscheinens über einem Stein, dann werden alle weiteren Änderungen dieses Kontaktpunktes entsprechend einer *Flick*-Geste interpretiert. Damit ist es möglich, einen Stein über das Spielfeld zu schieben und dann in der Bewegung den Kontaktpunkt zu lösen. Der so beschleunigte Stein rutscht dann entsprechend der erreichten Geschwindigkeit und der Richtung auf dem Eis weiter.

Trifft der Kontakt aber auf keinen Stein, liegt also auf dem Hintergrund, dann werden alle weiteren Änderungen dieses Kontaktpunktes als *Proxy*-Technik interpretiert. Unter dem Kontaktpunkt erscheint dann ein kleiner runder Puck, der dem Anwender als Schläger dient. Mit diesen Schlägern können die Steine dann angestoßen werden.

*Der Lernmodus*

In diesem Modus werden zuvor definierte Aktionen erst nach Kommando ausgeführt. Wie beim echten Billard werden die Steine mit Hilfe eines Queues (Spielstock) angestoßen. Im Lernmodus kann mindestens die Stoßkraft (Stärke des Stoßes), meist auch der Kontaktpunkt zwischen Queue und Stein, eingestellt werden. Dabei gibt es jeweils eine abstrakte Visualisierung des Queues und der Laufbahn des Steins. Die Stoßstärke wird dann durch den Abstand der Finger zum Stein und der Kontaktpunkt durch die relative Lage der Finger zum Stein definiert. Im Gegensatz zu den Freiheiten im Spaßmodus geht damit eine höhere Präzision einher, da die Geschwindigkeit und die Richtung des Steins besser Vorausgesagt werden kann. Durch die Möglichkeit der Bestimmung des Kontaktpunktes zwischen Queue und Stein kann zusätzlich „um die Ecke“ gestoßen werden.

Entsprechend zum Spaßmodus werden auch Finger-Kontaktpunkte entsprechend der Eigenschaften Lage, Reihenfolge und Anzahl einer von drei verschiedenen *Slingshot*-Techniken (siehe 3.4, Seite 23) zugeordnet. Diesen Techniken liegt

die Idee einer Zwille oder Katschi zu Grunde, wobei es sich um modifizierte *Drag'n'Drop*-Technik handelt. Durch das Bewegen eines Fingers wird die Zwille gespannt beziehungsweise der Queue von der Kugel entfernt und so die Stoßstärke beeinflusst.

Mithilfe des Lernmodus können Spielzüge geplant und verstanden werden. Er bietet gerade einem Lehrer oder Betreuer die Möglichkeit, die Prinzipien wie Beschleunigung, Kollision oder Reibung an geeigneter Stelle zu erläutern und zu vermitteln.

### 3.3.2 *Bulldozer*

In der Anwendung Bulldozer übernimmt jeder Teilnehmer die Kontrolle über einen großen Bulldozer (Abb. 17). Dieser wird über eine Fernbedienung gesteuert. Jeder Bulldozer besitzt zwei gegenüberliegende Kettenantriebe, ähnlich einem Panzer. Jede dieser Ketten kann unabhängig von der anderen angetrieben werden. Dadurch ist es eine wahre Herausforderung und bedarf einer gewissen Trainingsphase, den Bulldozer zu kontrollieren. Danach jedoch sind schnelle, wendige und komplexe Fahrmanöver möglich und die Vorzüge dieser Antriebsart zeigen sich. Ziel ist es, neben der Kontrolle des Bulldozers, vier verschiedene Müllarten zu trennen und mit dem Bulldozer in dafür vorgesehene Bereiche abzutransportieren. Der Bulldozer fungiert in dieser Situation also als eine Art Werkzeug (Tool), welches es zu beherrschen gilt, um damit andere schwierigere Aufgaben zu erledigen. Ist das Feld auch vom letzten Müll gesäubert, wird automatisch für Nachschub gesorgt und eine neue noch größere Ladung Müll erscheint auf dem Feld.

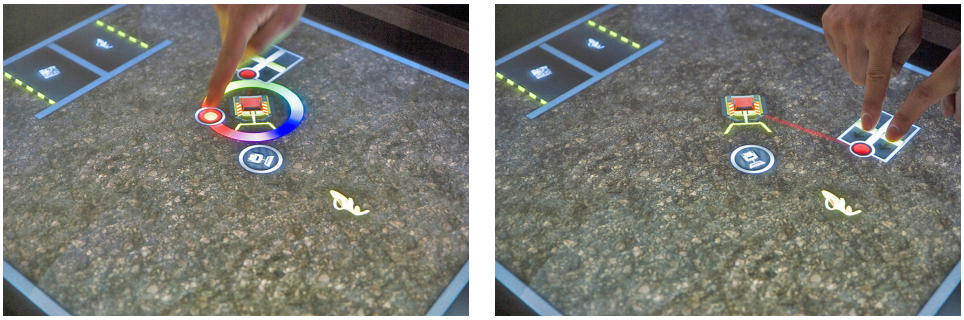


Abbildung 17: Der Anwender kontrolliert seinen eigenen Bulldozer mithilfe einer Fernbedienung. Der Bulldozer als Werkzeug dient der Unterstützung der Mülltrennung und des Abtransportes.

*Warum Bulldozer steuern und damit Müll trennen?*

In vielen Schulen und den entsprechenden Lehrplänen ist das Thema Umweltschutz fest verankert. Viele Kinder wiederum kennen und mögen Spielzeuge wie „Ferngesteuerte Autos". Wenn diese beiden Aspekte gemischt werden, entsteht eine Idee ähnlich dieser Anwendung – Mülltrennung und abstrakte Fahrzeugkontrolle. Für Grundschulen in Alberta, Kanada stehen sogar beide Themen im Lehrplan für die vierte Klasse Government of Alberta (1996).

Aus einem zentral gelegenen und für alle gut erreichbaren „Button" kann sich jeder Teilnehmer seinen eigenen Bulldozer per *Drag'n'Drop* (siehe 3.4, Seite 23) ziehen. Anschließend erscheint dieser zusammen mit der Fernbedienung. Eine Fernbedienung ist immer mit dem entsprechenden Bulldozer durch eine farbige Linie verbunden. Die Fernbedienung besteht aus zwei Schiebereglern (Abb. 18). Der linke Regler steuert den Antrieb (Geschwindigkeit) der linken Kette des Bulldozers, der rechte Regler analog dazu die rechte Kette. Beide Regler können unabhängig voneinander eingestellt werden. So kann beispielsweise der linke

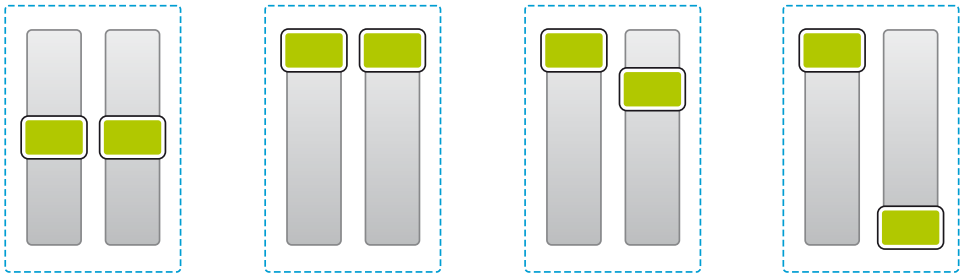



Abbildung 18: *a)* Nullstellung, keine Beschleunigung; *b)* Vollgas, beide Ketten laufen mit voller Geschwindigkeit vorwärts; *c)* Lange Kurve, die linke Kette beschleunigt schneller, der Bulldozer machte eine langgezogene Rechtskurve; *d)* Wende, die Ketten beschleunigen entgegengesetzt, der Bulldozer dreht sich „auf der Stelle“

Regler auf Maximum (Vorwärts) und der rechte Regler auf Minimum (Rückwärts) stehen. Dadurch dreht sich der Bulldozer um das Antriebszentrum beider Kettenantriebe rechtsherum. Dieses Lenkungsprinzip kommt ebenfalls bei realen Kettenfahrzeugen vor und wurde schon in der Multitouch Spieleanwendung „Tank“ (Wallin, 2008) angewendet.

*Dem Bulldozer kann ein Farbanstrich verpasst werden.*

Mit einer direkten Berührung des Bulldozers wird ein Farbkreis sichtbar beziehungsweise wieder unsichtbar geschaltet (Abb. 17). Dieser Kreis ermöglicht es, die Farbe des Bulldozer zu verändern und damit den Unterschied zwischen den Fahrzeugen im Spiel zu verdeutlichen. Gleichzeitig übernimmt auch ein Teil der Fernbedienung und die Verbindungslinie zwischen Fernbedienung und Bulldozer die ausgewählte Farbe.

### 3.3.3 *Planeten*

Die Anwendung Planeten beschreibt eine interaktive Sternwarte. Dieses virtuelle astronomische Observatorium dient der Beobachtung und Analyse von Gravitation. Diese wird dabei durch die Flug- oder im besten Fall Umlaufbahnen von Planeten um die Sonne erfahrbar gemacht. In der Anwendung können mehrere Teilnehmer gleichzeitig Planeten in einem Sonnensystem platzieren und anschließend eine Initialbewegung (-beschleunigung) sowie -richtung zuweisen. Abschließend muss der Planet durch eine einfache Berührung als „frei“ markiert werden. Daraufhin fließt seine Größe, Masse, Geschwindigkeit und Position in die Berechnung des Sonnensystems ein – das heißt, er wird von anderen Himmelskörpern beeinflusst, genauso wie er eine Wirkung auf diese anderen Planeten hat.

Der Weltraum – die unendliche Weite und die Fülle an Unerforschtem fasziniert den Menschen schon seit je her. Für Grundschulen in Alberta, Kanada steht das Thema Astronomie zum ersten Mal in Klasse 6 im Lehrplan (Government of Alberta, 1996). Einige deutsche Grundschulen lehren erste Sachverhalte ab der 5. Klasse (Engel, 2009). In der Regel können die Schüler in Deutschland aber erst in der Sekundarstufe das Fach Astronomie wählen.

Die Anwendung wird komplett durch die einfache *Drag'n'Drop*-Technik und durch „Antippen“ (Touch) bedient. Nach dem Spielstart besteht das System nur aus einer Sonne. Diese Sonne bildet das Zentrum – und damit den Mittelpunkt des Systems. Die Sonne als Objekt stellt den Teilnehmern zwei wichtige Funktionen zur Verfügung (Abb. 20).

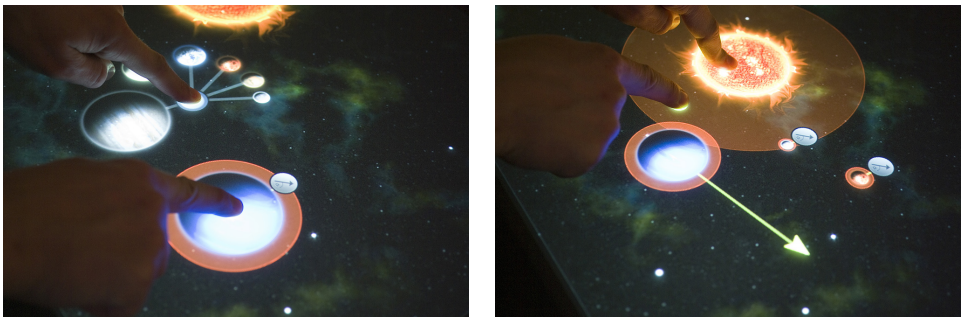


Abbildung 19: Planeten können ausgewählt und platziert sowie nach Festlegung der Initialgeschwindigkeit und -richtung „gestartet“ werden.

a) Wird die Sonne durch einen Teilnehmer verschoben, verschiebt sich automatisch auch die Sicht auf das Sonnensystem. Dabei wird also das komplette System verschoben beziehungsweise die Sicht auf das System. *Durchführung: Die Sonne berühren und anschließend den Finger in die gewünschte Richtung bewegen.*

b) Mit Hilfe der Sonne kann die Ansicht auf das System vergrößert (zoom-in) oder verkleinert (zoom-out) werden. *Durchführung: Die Sonne berühren mit einem Finger berühren (Finger A) und anschließend mit einem zweiten Finger (Finger B) den roten Kreis um die Sonne berühren. Nun den zweiten Finger (Finger B) linksdrehend oder rechtsdrehend um den ersten Finger (Finger A) bewegen.*

a) b)

Abbildung 20: *a)* Ein Verschieben der Sonne bewirkt eine Verschiebung des gesamten Systems; *b)* Zoom-Geste, zweiten Finger linksdrehend oder rechtsdrehend um den ersten Finger (die Sonne), bewirkt eine Verkleinerung beziehungsweise Vergrößerung der Ansicht des Sonnensystems

Die oben beschriebenen Funktionen beeinflussen die komplette Darstellung der Anwendung. Daher werden solche Funktionen als „global“ bezeichnet, da sie, bezogen auf Applikationen für mehrere Anwender, das Interface aller Anwender beeinflusst. Hier wurde bewusst entschieden, keine *Drag*- oder *Pinch*-Technik auf dem Hintergrund für die Aktionen zu nutzen. Diese sind zwar allgemein bekannt und werden mittlerweile auch in den meisten Multitouch-Smartphones eingesetzt, rufen hier jedoch einige Probleme hervor. So kann es vorkommen, dass ein Anwender einen Planeten stoppen und verschieben möchte, diesen jedoch am Anfang verfehlt, den Hintergrund trifft und so den Bildausschnitt verschiebt. Es ist auch problematisch, wenn zwei Anwender gleichzeitig den Bildausschnitt verschieben möchten. Wenn zwei Anwender auf engem Raum ihre Aktionen ausführen und dabei ein Fehler auftritt und die Anwender ihr Finger auseinander oder zueinander bewegen, könnte dies auch als *Pinch*-Geste interpretiert werden.

*Die Verzögerung von einer Sekunde bewirkt, dass nicht jeder Fehlversuch, ein Objekt anzuklicken oder zu selektieren, mit dem Bauwerkzeug verdeckt wird.*

Um einen oder mehrere Planeten für das System zu erzeugen, muss der Anwender den Hintergrund, also den Bereich um die Sonne, berühren. Nach ungefähr einer Sekunde erscheint dort ein Werkzeug mit Planeten (Abb. 21). Nun können entweder ein oder mehrere Planeten mit einem anderen Finger auf den Hintergrund gezogen werden (*Drag'n'Drop*). Sobald der Finger des Bauwerkzeuges losgelassen wird, verschwindet das Werkzeug. Auf diese Weise lassen sich Planeten erzeugen, welche noch nicht zum System gehören, sich nicht bewegen und keine anderen Planeten beeinflussen. Dies wird durch einen roten Kreis um den Planeten verdeutlicht. In diesem Zustand lassen sich die Planeten einfach verschieben. In der Nähe und direkt mit dem Planeten verbunden, liegt eine kleine Schaltfläche mit einem Pfeil. Diese Schaltfläche (Abb. 21) lässt sich unabhängig bewegen und definiert die initiale Richtung und Geschwindigkeit des Planeten. Die Geschwindigkeit wird hierbei über die Länge des Pfeils, den Abstand zum Planeten definiert. Wenn der Planet aktiviert und an das System übergeben wird, dann bekommt dieser im ersten Moment genau diese definierte Richtung und Geschwindigkeit.

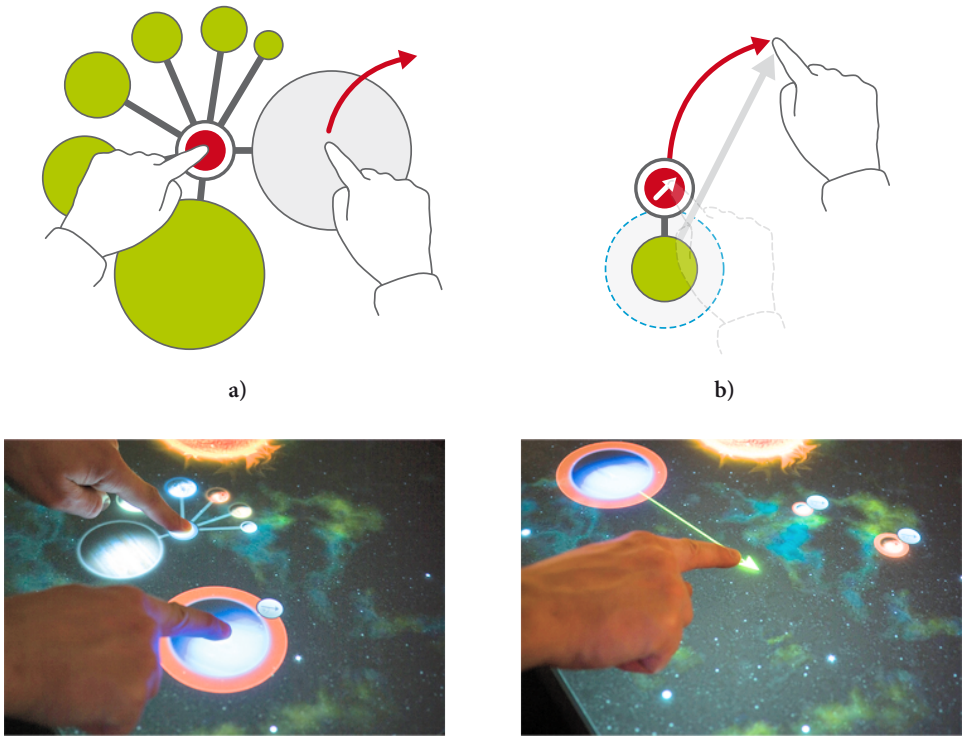



Abbildung 21: *a)* Werkzeug dient der Auswahl und Erstellung von Planten für das Sonnensystem; *b)* Schaltfläche zur Definition der ersten Bewegungsrichtung und -geschwindigkeit eines Planeten

Die Planeten lassen sich durch anklicken aktivieren und deaktivieren. Somit ist es möglich, einen sich bewegenden Planeten anzuhalten, einen oder mehrere Parameter (Position, Richtung, Geschwindigkeit) zu ändern oder nur zu studieren. Sollten zwei Planeten miteinander kollidieren, dann verschwindet der kleinere. Kollidiert ein Planet mit der Sonne, verschwindet dieser. Es erfordert Übung und Eingewöhnung, mehrere Planeten in eine stabile Laufbahn um die Sonne zu bringen, aber es ist möglich. Um die Laufbahnen von Planeten zu verfolgen, zeichnen diese einen farbigen Pfad hinter sich. Dieser Pfad verblasst nach einiger Zeit.

## 3.4 INTERAKTIONSTECHNIKEN

Eine besondere Anforderung an die Anwendungen war, eine möglichst unterschiedliche Interaktion von Anwendung zu Anwendung zur Verfügung zu stellen. So nutzen die drei implementierten Anwendungen eine Vielzahl von verschiedenen, teilweise auch aufeinander aufbauenden, Interaktionstechniken. Eine Übersicht der verwendeten Techniken gewährt die Tabelle 1.

| | Menü | Billiard (Spaßmodus) | Billiard (Lernmodus) | Bulldozer | Planeten |
|---|---|---|---|---|---|
| Antippen (Klick) | | ○ | | ● | ● |
| Drag'n'Drop | ● | ○ | ○ | ● | ● |
| Flick | | ● | | | |
| Proxy Objekte | | ● | | | |
| Slingshot | | | ● | | |
| Virtuelle Werkzeuge | | ○ | | ● | ● |
| Rotate | | | | | ● |

*Legende* ● *Technik wird eingesetzt* ○ *Technik wird nur teilweise eingesetzt*

Tabelle 1: Übersicht der verwendeten Interaktionstechniken innerhalb der einzelnen Anwendungen.

In Tabelle 1 wird deutlich, dass es Interaktionstechniken gibt, die wiederum andere Techniken voraussetzen, um eine andere, komplexere oder vielfältigere Interaktionstechnik zur Verfügung zu stellen. Einfache *Klick-* und *Drag'n'Drop*-Aktionen etwa finden sich in sehr vielen Techniken wieder. So basieren die verwendeten *Flick-*, *Proxy-* und *Slingshot*-Techniken auf *Drag'n'Drop*-Aktionen. *Werkzeuge* wie Fernbedienungen oder das Planetenwerkzeug nutzen *Klick-* und *Drag'n'Drop*-Aktionen. Auch Gesten wie zum Beispiel eine „Rechteck-Zeichen-Geste" basieren auf *Klick-* und *Drag'n'Drop*-Aktionen.

*Open-Loop ist On-the-fly, Closed-Loop ist eher Turn-by-turn*

Des Weiteren sind die Techniken in Tabelle 1 als „Closed-Loop"- oder „Open-Loop"-Technik gekennzeichnet. *Closed-Loop*-Techniken setzen voraus, dass der Anwender seine Eingaben kontinuierlich an die Objekteigenschaften (Position, Rotation, Geschwindigkeit, usw.) anpasst. Diese Eigenschaften müssen in Form von visuellem Feedback an den Anwender vermittelt werden. *Open-Loop*-Techniken hingegen ähneln realen Aktionen wie Werfen, Stoßen oder Schieben. Sobald das Objekt die Hand des Anwenders verlässt, hat er keine Kontrolle über das Objekt und kann keine exakte Verhaltensänderung hervorrufen (Reetz u. a., 2006). *Open-Loop* lässt sich sehr gut mit „On-the-fly" beschreiben, während *Closed-Loop* eher durch „Turn-by-turn" beschrieben werden kann. Reetz u. a. (2006) beschreiben den Vorteil von *Open-Loop* damit, dass sich diese Techniken schnell ausführen lassen und der Anwender seine Aufmerksamkeit sehr schnell auf andere Dinge lenken kann. Jedoch benötigen diese Techniken in der Regel mehr Übung, um einen gewissen Grad an Präzision zu erreichen. Dies lässt sich sehr gut am Beispiel des Spaßmodus der Anwendung *Billard auf Eis* (siehe 3.3.1, Seite 17) nachvollziehen.

### 3.4.1 *Realisierte und eingesetzte Techniken*

*Klick und Drag'n'Drop*

*Klick und Drag'n'Drop*-Aktionen sind wie oben beschrieben zwei elementare Aktionen. Sie werden in sehr vielen Anwendungen direkt benutzt und sind Teil einer anderen Interaktionstechnik. Oft werden diese Aktionen eingesetzt, um virtuelle Objekte zu (de-)selektieren, bewegen und verschieben oder Buttons zu aktivieren.

*Flick*

*Flick* ist eine vielseitige und oft eingesetzte *Open-Loop*-Technik, um virtuelle Objekte über lange Distanzen an durch den Nutzer nicht direkt erreichbare Orte zu verschieben. Bei der *Flick*-technik werden virtuelle Objekte durch direktes auswählen und eine anschließende „Schiebe-Aktion" in eine Richtung beschleunigt. Ein Beispiel:

BSP. An einem langen Tisch sitzen sich zwei Personen gegenüber. Die eine Person möchte der anderen eine leere Kaffeetasse zu kommen lassen. Dazu stellt sie die Tasse auf den Tisch, fasst diese an, beginnt die Tasse mit hoher Geschwindigkeit in Richtung der zweiten Person zu schieben und lässt die Tasse dann los. Wurde die Tasse genug beschleunigt und die Richtung stimmt, wird diese über den Tisch gleiten und die andere Seite erreichen.

Reetz u. a. (2006) haben Wurf- beziehungsweise Schiebetechniken in verschiedenen Szenarien getestet und analysiert und die Technik „Superflick" vorgestellt. Diese Technik erweitert die herkömmliche Flick-Technik um einen Korrekturschritt. Dabei kann der Anwender die Richtung des Objektes anpassen, solange sich das Objekt bewegt. Die Analyse der Techniken ergab, dass die herkömmliche Flick-Technik die schnellste Technik ist. Im Vergleich zu den anderen Techniken waren die Ausführungszeiten also deutlich kürzer. Dem gegenüber steht aber die hohe Fehlerrate, also die Treffsicherheit beziehungsweise die Präzision.

*Proxy Objekte*

*Proxy Objekte* beschreiben eine Interaktionstechnik, bei der andere virtuelle Objekte als Hilfsmittel genutzt werden. Eine gute Beschreibung findet sich bei Wilson u. a. (2008):

BESCHREIBUNG Kontaktpunkte oder Touchpunkte werden in der Anwendung durch starre Körper wie zum Beispiel Würfel oder Kugeln repräsentiert. Diese Körper dienen als Näherung der Kontaktpunkte und interagieren mit anderen virtuellen Objekten durch Kollision und Reibung.

Die Technik lässt sich sehr leicht in Anwendungen umsetzen, denen eine physikalische Schnittstelle zu Grunde liegt. Hier können die Kontaktpunkte (Touchpunkte) ganz einfach durch das Einsetzen einfacher Objekte in die Szene verarbeitet werden. Die eigentliche Kollisionskontrolle, also die Verarbeitung der Kontaktpunkte, übernimmt dann die Physik-Engine.

In der Anwendung *Billard auf Eis* (siehe 3.3.1, Seite 17) agieren die Finger als eine Art Schläger. Die Steine prallen von den „Finger-Objekten" ab. Dieser Effekt verstärkt sich, wenn die Finger selbst bewegt oder beschleunigt werden. Auf Grund der geringen Kontrollmöglichkeiten ist die *Proxy*-Technik eine *Open-Loop*-Technik (Tabelle 1).

*Slingshot*

*Slingshot* als Technik basiert auf der von Hascoët (2003) vorgestellten *Drag-and-Throw*-Technik. Die Funktionsweise dieser Technik ist von einer realen Zwille (Katschi) abgeleitet. Dabei wird das Objekt von seiner Position weggezogen (entgegengesetzt zum Ziel) und dann losgelassen. Das Wegziehen entspricht dem „Spannen des Gummis“ und dem Zielen. Sobald das Objekt losgelassen wird, wird es durch die Kraft der Spannung im Gummi beschleunigt. Bei Hascoët (2003) wird dem Anwender eine Vorschau als visuelles Feedback gegeben, an welcher er seine Eingabe anpassen kann und er wurde durch eine „Snap-to-target“-Funktion (Andocken an möglichen Zielen in der Umgebung) unterstützt.

Ohne eine vollständige Vorschau der Geschehnisse oder eine „Snap-to-target“-Funktion zu geben, wird die *Drag-and-Throw*-Technik im Lernmodus der Anwendung *Billard auf Eis* (siehe 3.3.1, Seite 17) eingesetzt. Hier basiert die *Slingshot*-Technik eher auf dem Prinzip des Anstoßes mit einem Queue wie beim Billard. Da die *Drag-and-Throw*-Technik hier um Multitouch erweitert wurde und insgesamt aus drei einzelnen Techniken besteht, wurde dieses Paket in *Slingshot*-Technik umbenannt und besteht aus je einer Ein-, Zwei- und Drei-Fingertechnik (Abb. 22). Bei allen Slingshot-Varianten bestimmt der Abstand gleichzeitig die Kraft, mit der das selektierte Objekt angestoßen wird.

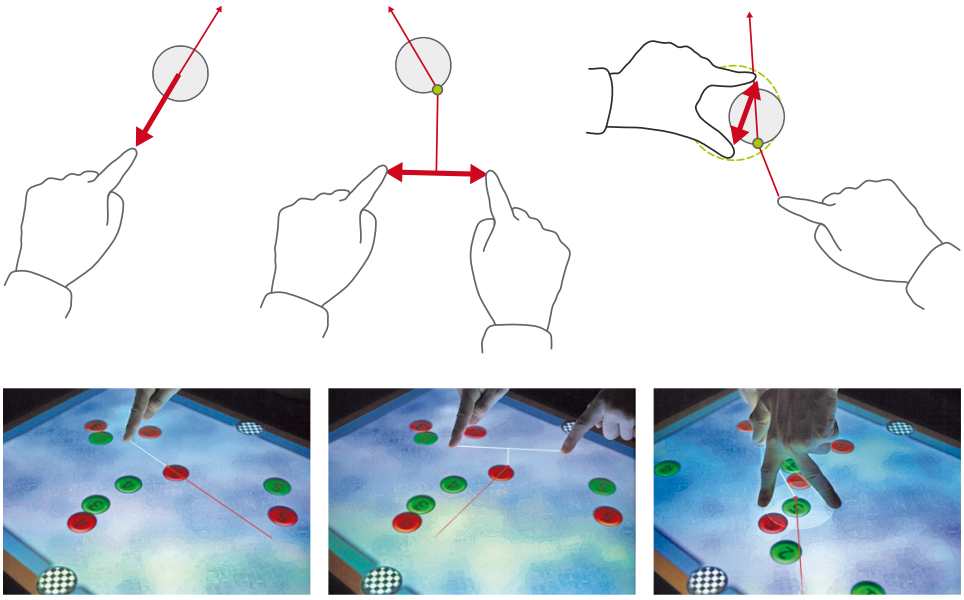


Abbildung 22: Je eine *Slingshot*-Technik für eine Ein-, Zwei- oder Drei-Fingerinteraktion.

EIN-FINGERTECHNIK Bei dieser *Slingshot*-Variante muss der Anwender das Objekt berühren und dann seinen Finger entgegen der Zielrichtung ziehen. Dabei erhält er eine Vorschau der Laufbahn in form einer dünnen geraden Linie. Je weiter sich der Finger von dem Objekt entfernt, desto höher wird die Beschleunigung des Objektes sein. Wird der Finger losgelassen, wird das Objekt unmittelbar angestoßen.

ZWEI-FINGERTECHNIK Hier beginnt der Anwender wie bei der oben beschriebenen Ein-Fingertechnik. Bevor er das Objekt jedoch los lässt, nimmt er einen zweiten Finger, berührt damit das schon selektierte Objekt und führt auch diesen vom Objekt weg. Die beiden Finger bilden nun einen Vektor. Der Richtungs-

beziehungsweise Zielvektor resultiert aus dem orthogonalen Vektor der beiden Finger. Als ob der virtuelle Queue nun von der Mitte der beiden Finger und auf diesem orthogonalen Vektor liefe, kann der Anwender so den Kontaktpunkt, den Ansatzpunkt der Kraft beziehungsweise den Kollisionspunkt zwischen Queue und Objekt definieren. So kann ein Objekt „um die Ecke“ gestoßen werden. Dabei erhält das Objekt gemäß physikalischer Grundlagen eine Rotation um den eigenen Schwerpunkt. Sobald einer der beiden Finger nun losgelassen wird, wird das Objekt angestossen.

DREI-FINGERTECHNIK Bei dieser *Slingshot*-Variante umgibt der Anwender das gewünschte Objekt mit zwei Fingern. In Abhängigkeit des Zeitpunktes beider Kontaktpunkte werden diese als Paar erkannt und einem zwischen Ihnen liegenden Objekt zugeordnet. Nun wird mit einem dritten Finger, nach Hancock u. a. (2009) einem Finger der gegenüberliegenden Hand, das selektierte Objekt berührt und der Finger wie bei den Ein- und Zwei-Fingervarianten vom Objekt wegbewegt. Dieser dritte Finger und die Mitte der beiden anderen Finger definieren nun den Richtungs- beziehungsweise Zielvektor. Als letzte Aktion muss der Anwender nun den dritten Finger loslassen und das Objekt wird angestoßen. Verglichen mit der Ein- und Zwei-Fingervarianten, kommt diese Variante dem realen Anwenden einer Zwille (Katschi) am Nächsten.

### *Virtuelle Werkzeuge*

*Virtuelle Werkzeuge* als Technik beschreiben eine abstrakte Form der Objektmanipulation. In Hancock u. a. (2009) werden diese als vorhandene virtuelle Objekte bezeichnet, welche auf andere virtuelle Objekte einwirken und so Änderungen an diesem hervorbringen können. Die oben erwähnten *Proxy-Objekte* bilden dabei eine Unterklasse dieser *virtuellen Werkzeuge*, denn sie können Änderungen in Form von beispielsweise Position, Beschleunigung und Rotation hervorrufen, wie in dem Spaßmodus der Anwendung *Billard auf Eis* (siehe 3.3.1, Seite 17).

Es gibt aber auch andere Beispiele für virtuelle Werkzeuge. Hancock u. a. (2009) haben eine virtuelle Sandbox für Kinder entwickelt, in der ein virtueller Schlauch, angeschlossen Farb- beziehungsweise Textur-Eimer, die Möglichkeit bietet, andere Objekte zu kolorieren. Eine andere gezeigte Möglichkeit, Objekte einzufärben, ist die Verwendung eines farbigen Tuches, welches über das Objekt gelegt werden muss. Sobald dieses Tuch losgelassen wird, wird das drunterliegende Objekte eingefärbt und das Tuch verschwindet. Ein anderes Objekt wiederum kann die Tageszeit, also den Sonnenstand und damit die Stärke und Länge der Objektschatten, ändern.

In der Anwendung *Bulldozer* (siehe 3.3.2, Seite 19) agiert der Bulldozer selbst als *virtuelles Werkzeug*, denn er beeinflusst andere Objekte in der Position. Auch die Fernbedienung kann als eine Art abstraktes Werkzeug angesehen werden, denn diese bestimmt die Ausrichtung und Beschleunigung und damit die Position des Bulldozers.

### *Rotieren*

*Rotieren* beschreibt keine komplizierte oder in der Literatur schon oft behandelte Technik. Viel mehr handelt es sich dabei um die Umsetzung einer Idee für die Realisierung einer kontinuierlichen Wertänderung. Angelehnt an einen realen drehbaren Regler für die Lautstärke an Radios oder Receivern, funktioniert diese Technik ähnlich in der Anwendung *Planeten* (siehe 3.3.3, Seite 20). Da eine Ein-Finger-Aktion auf dem Objekt (hier die Sonne) schon für eine andere Aktion

vergeben ist, wurde die *Rotationstechnik* hier als Zwei-Finger-Technik implementiert.

Der Anwender hält mit dem ersten Finger das Objekt fest und führt einen zweiten Finger in einer Kreisbahn um den ersten Finger (Abb. 20). Die Winkeländerung zwischen den beiden Fingern kann anschließend einer anderen Funktion (hier das Vergrößern oder Verkleinern) zugeordnet werden.

### 3.4.2 *Andere Techniken*

Die folgenden Techniken wurden ebenfalls in Betracht gezogen, implementiert und getestet, sind jedoch in der vorliegenden Programmversion nicht vorhanden.

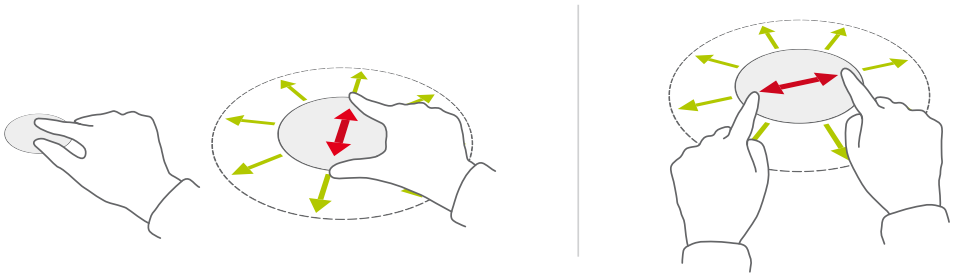


Abbildung 23: *Pinch*-Technik: Wenn der relative Fingerabstand erhöhet wird, bedeutet dies in der Regel Vergrößern; wird der Abstand verringert, bedeutet dies Verkleinern

#### *Pinch*

*Pinch* oder *Pinching* bezeichnet eine Technik, bei der die relative Lageänderung von zwei Fingern einer anderen Funktion zugeordnet wird. Gewöhnlich wird das auseinander bewegen der Finger als Vergrößerung und das aufeinander zubewegen als Verkleinerung interpretiert (Abb. 23). Bekannt ist die Technik von vielen erhältlichen Multitouch-Smartphones oder Galerie-Applikationen für Multitouch-Tabletops. Sie kann auf Grund des hohen Verwendungsgrades als „Standard"-Technik für Zooming (Vergrößerung und Verkleinerung) verstanden werden.

Diese Technik wurde während der Entwicklung der Anwendung *Planeten* (siehe 3.3.3, Seite 20). als Zoom-Technik in Betracht gezogen. Sie wurde nicht in die vorliegende Anwendungsversion übernommen, da in einer sehr verkleinerten Systemansicht keine zwei Finger von einem Anwender auf der Sonne platzieren ließen. Der für eine gut funktionierende *Pinch*-Technik nötige und maximale Verkleinerungsgrad wurde während der Entwicklung als nicht ausreichend bewertet.

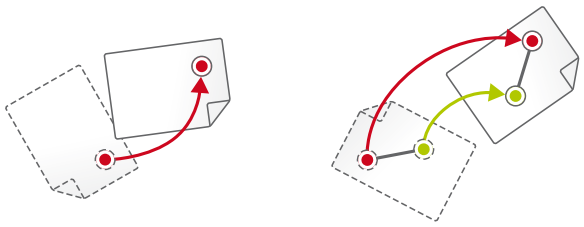


Abbildung 24: *RNT*-Technik, nach Hancock u. a. (2006)

*RNT*

*RNT* steht für „Rotate N' Translate". Hancock u. a. (2006) beschreiben ein Verfahren bei dem sich virtuelle Objekte durch eine einzige Technik in Hinsicht auf Translation und Rotation manipulieren lassen. Diese Technik ist ein weiteres Beispiel der Überführung einer aus der Realität bekannten Technik in die virtuelle Welt. Das folgende Beispiel erklärt die Funktionsweise der Technik.

BSP. Auf einem Tisch liegt ein einzelnes Blatt Papier. Drückt eine Person nun mit einem Finger leicht auf eine *Ecke* des Papierblattes und bewegt dabei den Finger über den Tisch, wird das Blatt, an der Ecke fixiert, über den Tisch gezogen und dabei auch gedreht. Dies lässt sich gut beobachten, wenn die Person eine große Kreisbewegung ausführt.

In der Realität funktioniert diese Technik auf Grund der Zugkraft an der Ecke des Blattes und der vorhandenen Reibung zwischen dem Blatt Papier und dem Tisch. Hancock u. a. (2006) beschreiben Algorithmen für *RNT*-Techniken, die sowohl mit einem als auch zwei mit Kontaktpunkten (Touchpunkten) arbeitet (Abb. 24).

## 3.5 AUSWERTUNG

Die *PhysicBox* wurde in Java programmiert. Da auch alle verwendeten Bibliotheken (siehe 4.1.3, Seite 34) plattformunabhängig sind, trifft diese Eigenschaft auch auf den Prototypen zu. Dank der Einbindung und Verwendung der Bibliothek *jtl* (siehe 4.2.1, Seite 34) läuft die *PhysicBox* auf verschiedenen Tabletops. Derzeit läuft der Prototyp auf dem `Microsoft Surface`, `SMART Table` und `TUIO-basierten Tabletops`. Die *PhysicBox* läuft inklusive der Nutzerinteraktion auf dem `SMART Table` und `Microsoft Surface` durchgängig mit mindestens 30, meist mehr Bildern in der Sekunde. Damit läuft der Prototyp flüssig. Nach zahlreichen langen Testszenarien, unter anderem bei Demonstrationen für Mitarbeiter der Firma SMART Technologies, anderen Forschern und einer Lehrerin, Videoaufnahmen und auch am Tag der offenen Universitätstür der Universität Magdeburg, kann der Prototyp ebenfalls als stabil bezeichnet werden, da er in diesen unterschiedlichen Situationen nicht abstützte oder neugestartet werden musste.

Leider konnte der Prototyp an der damaligen „SMART Table Competition" auf der *ITS 2009* nicht teilnehmen, da zu diesem Wettbewerb nur Entwickler aus Kanada, USA und Großbritannien zugelassen waren.

### 3.5.1 *Die Anwendungen*

Der Prototyp besteht aus insgesamt drei eigenständigen und lauffähigen Anwendungen. Zusätzlich gibt es als eine Art Startmenü eine Fließband-Anwendung, mit der die eigentlichen Anwendungen gestartet werden können. Alle Anwendungen wurden mit ansprechenden und aufwändig gestalteten Grafiken versehen. Das verleiht der *PhysicBox* einen professionellen Charakter, obwohl es sich um einen Prototypen handelt.

Eine häufig bemängelte Funktion betraf die kleinen Schaltflächen in den Ecken einer jeden Anwendung. Das System kehrte nach einmaligem Auswählen einer der vier Schaltflächen zum Hauptmenü zurück. Oft kamen Anwender zufällig und unabsichtlich auf eine der Schaltflächen und das Spiel endete. Dem wurde in einer Überarbeitung vorgebeugt, in dem nun mindestens zwei der vier Schaltflächen innerhalb eines kurzen Zeitraums betätigt werden müssen. Diese Technik der

Mehrfachbestätigung ist in Mehrnutzer-Anwendungen durchaus üblich. Dieses Problem verdeutlicht aber eine sehr interessante Problematik von Mehrnutzer-Anwendungen. Es gibt immer Aktionen oder Befehle, die eine Auswirkung auf das ganze System, also auch auf alle anderen Anwender, haben. Die Ausführung und Aktivierung dieser sogenannten „globalen Operationen" müssen immer mit großer Sorgfalt in ein System eingebettet werden. Ein weiteres Bespiel dieser Problematik lässt sich in der Anwendung *Planeten* beobachten, wenn ein Spieler gerade einen Planeten positionieren, ein Anderer aber den Bildausschnitt bewegen, vergrößern oder verkleinern möchte.

BILLARD AUF EIS als erste Anwendung implementiert, nutzt in den zwei Ausführungsmodi unterschiedliche Interaktionstechniken. Im „Spaßmodus" werden die Techniken *Flick*- und *Proxy Objekte* eingesetzt. Die *Proxy*-Technik bringt ein sehr schnelles, aber ungenaues Spiel hervor, wohingegen sich die *Flick*-Technik unerwartet präzise einsetzen lässt. Das ist damit begründet, dass Entfernungen auf den getesteten Tabletops sehr gering sind. In der Regel wird die Technik zur Überbrückung langer Distanzen eingesetzt. Das ist hier nicht der Fall. Ein weiterer Vorteil der *Flick*-Technik ergibt sich aus der virtuellen Verbindung zwischen Kontaktpunkt (Finger) und Objekt. Wird das Objekt nicht losgelassen und der Finger bewegt, folgt das Objekt dem Finger, als ob ein elastisches Band beide verbindet.

Im „Lernmodus" der Anwendung wird die präzisere *Slingshot*-Interaktionstechnik eingesetzt, wobei jeweils eine Ein-, Zwei und Dreifinger-Variante verfügbar ist. Die Einfinger-Variante wurde von sehr vielen Anwendern als einfachste und favorisierte Variante bezeichnet. Die Möglichkeit, den Kollisionspunkt zwischen virtuellen Queue und Spielstein zu beeinflussen, wurde von den wenigsten Anwendern genutzt und als „zu kompliziert" beschrieben. Zum einen fehlte Einigen eine visuelle Rückmeldung, wenn der virtuelle Queue am Objekt vorbei zielt. Zum anderen folgt einem am Spielstein seitlich ausgeführten Stoss lediglich eine Rotation des Steins selbst. Das „um die Ecke schießen" ließe sich auch mit der Einfinger-Variante und der einfachen Verlegung des Queues realisieren. Wenn es jedoch um den Lernaspekt oder die Nachvollziehbarkeit des Spielkonzeptes ging, wurde gerade die Zweifinger-Variante positiv bewertet. Einzig die Dreifinger-Variante wurde zwar als „lustig" und „spaßig", in der Gesamtheit aber oft als zu kompliziert beschrieben. Meist kam es vor, dass sich die Spieler gegenseitig mit ihren beiden Händen in die Quere kamen oder sie gar nicht den wichtigen Teil der Darstellung, den Spielstein und den Kollisionspunkt (zwischen Queue und Stein) sehen konnten, da die komplette Hand den Bereich verdeckte. Daher scheint die Dreifinger-Variante in dieser Form eher für einen Spaßmodus geeignet zu sein.

BULLDOZER sorgte bei vielen Anwendern für einen „Ahh"-Effekt. Die Steuerung bedurfte bei vielen einer intensiven Lern- und Testphase, während es bei einigen auf Anhieb klappte. Diese Anwendung sorgte für viel Spaß und Unterhaltung bei den meist erwachsenen Anwendern. Oftmals führten diese mit ihren Bulldozern ein Rennen und einen Kampf um die Müll-Objekte. Das eigentliche Recyceln geriet dabei in den Hintergrund. Bemängelt wurden die kleinen Fernbedienungen. Gerade bei Personen mit größeren Fingern ähnelte die Fahrt eher einem ständigen Wechsel zwischen Stopp und einer Vollgasfahrt. Eine feine Geschwindigkeitsregulierung war oft nicht möglich. Für viele Anwenderinnen war dies jedoch kein Problem.

Einigen Anwendern war der Hintergrund des Spielfeldes zu strukturiert und ablenkend. Vor allem jedoch fehlten visuelle Rückmeldungen während der Erzeugung eines Bulldozers. Während dieses Vorganges konnten die Anwender nicht

erkennen, was gerade passiert. Dies ist ein sehr grundlegendes Problem, welchem unbedingt Rechnung getragen werden muss, denn der Anwender sollte zu jedem Zeitpunkt und in jeder nur denkbaren Situation genau wissen, was gerade in der Anwendung passiert.

Ein weiteres Manko der Anwendung stellt eine fehlende Möglichkeit da, den Bulldozer eines Spielers wieder aus dem Spiel zu entfernen. Es ist zwar möglich, für jeden Spielen einen Bulldozer zu erzeugen, eine Funktion zum Entfernen fehlt der Anwendung bisher. Gerade bei langen Laufzeiten, wie es an einem Tag der offenen Tür vorkommt, machte sich dieses Problem bemerkbar. Eine mögliche Lösung wäre eine weitere Option während der Farbwahl. Elegant und gut in das Spiel integriert wäre eine Art Garage, in die der Bulldozer gefahren werden kann.

PLANETS wurde insgesamt als die komplizierteste Anwendung beschrieben. Gleichzeitig wurde sie aber wesentlich länger und auch öfter ausgeführt. Oft beobachteten die Anwender einfach ihre Sonnensysteme minutenlang, manchmal wurde auch ein regelrechter Wettkampf um das „schönste“ System geführt. Trotz aller Kompliziertheit faszinierte diese Anwendung am meisten. Aber es handelt sich auch um die Anwendung, welche mehrere Instruktionen benötigte, bis die Anwender, meist Erwachsene, alle Funktionen beherrschten. Alle Anwender hatten große Schwierigkeiten, den geeignetsten Abstand zur Sonne und die Geschwindigkeiten eines Planeten zu definieren. Das Handeln der Anwender konnte hier am besten mit „Try-and-Error“ (Versuch-und-Fehler) beschrieben werden. Hier ließe sich durch verschiedene visuelle Rückmeldungen eine Verbesserung herbeiführen. Während sich ein Planet bewegt, könnte der aktuelle Geschwindigkeitsvektor dauerhaft sichtbar sein. Auch könnte eine abstrakte Wertskala die Einstellung der Geschwindigkeit verbessern. Eine gänzlich andere Idee kehrt das Bedienkonzept um. Anstatt die Geschwindigkeit und die Richtung des Planeten am Anfang zu definieren, könnte der Anwender auch den Anfang der Laufbahn skizzieren.

*Die Planetenbahn skizzieren statt Geschwindigkeit und Richtung einzustellen*

Sehr oft kam es vor, dass vor allem erfahrene Computeranwender dem Spiel eine Zoom-Funktion voraussagten und als erstes die von Multitouch-Smartphones bekannte und allzu gegenwärtige *Pinch*-Geste (siehe 3.4.2, Seite 27) ausführten. Da im Prototypen eine andere Technik zum Vergrößern und Verkleinern der Ansicht gewählt wurde, suchten den Anwender den Fehler bei sich, wiederholten die Geste oder schauten skeptisch zum Entwickler. Es gibt für beide Techniken Vor- und Nachteile. Hier sollte jedoch die Verwendung der Zoom-Technik neu überdacht werden, um die Erwartungen der Anwender zu bestätigen. Gleiches gilt für das Verschieben des Blickausschnittes. Hier versuchten es die Anwender einfach auf dem Hintergrund, in der Erwartung das System zu verschieben.

Eine weitere und häufige Anmerkung war der Wunsch, die Zeit in dem Sonnensystem anzuhalten oder sogar zu variieren. In der Tat würde ein solches Merkmal den didaktischen Anspruch unterstreichen und beispielsweise einem Lehrer sie Zeit geben, die zugrundeliegenden Prinzipien zu erläutern.

### 3.5.2 *Probleme, Fragen und weiterführende Themen*

Während der Entwicklung des Prototypen und auch aus diesen Auswertungen ergeben sich einige interessante Probleme, Fragen und Themen, die nicht in dieser Arbeit adressiert und bearbeitet wurden.

1. „Globale Operationen“: Entwicklung von Richtlinien und Hinweisen für die Behandlung und Verwendung von globalen Operationen

- Wann treten diese Situationen in der realen Welt auf und wie werden diese bewältigt?
- Lassen sich diese realen Strategien generalisieren und in die virtuelle Welt übertragen?
- Wie werden diese Operationen in vorhandenen Anwendungen realisiert?
- Lassen sich vorhandene Lösungen anderer Anwendungsbereiche auf Tabletops und vor allem Multitouch-Tabletops übertragen?

2. Entwicklung von Anwendungen mit didaktischem Hintergrund
   - Würden Tabletops in der Schule als Gesamtheit akzeptiert werden und einen Mehrwert bringen?
   - Wie muss solch eine Anwendung aus Sicht des Lehrers aussehen und wie oft würde er diese einsetzen?
   - Ab welcher Klasse oder welchem Alter lassen sich Tabletops sinnvoll einsetzen?
   - Visuelle Gestaltung der Anwendungen: Sprechen gezeichnete, Comic-stilisierte Darstellungen Kinder mehr an als realistische Stile? Ist überhaupt ein Unterschied messbar?
3. Kinder und der Einsatz von Multitouch-Tabletops
   - Lassen sich bekannte und erforschte Interaktionstechniken auch bei Kindern sinnvoll einsetzen?
   - Welche Auswirkungen hat der physische Unterschied zwischen Erwachsenen und Kindern (Körper-, Hand- und Fingergröße)?
   - Wie verhalten sich Kinder in einer Tabletop-Umgebung? Stichwort private Bereiche und Aktionsradien (Reichweite)
4. Interaktionstechniken in der Analyse
   - Vergleich der Interaktionstechniken bei verschiedenen Aufgabenstellungen
   - Untersuchung der Interaktionstechniken und ihre Auswirkung auf Faktoren wie Spielspaß, Spieldauer, Behaltensleistung

## 3.6 ZUSAMMENFASSUNG

Dieser Teil der Studienarbeit widmet sich den Anforderungen an den Prototypen, dem Entwicklungsprozess und der Ideenfindung, sowie der Beschreibung der einzelnen Anwendungen und den Interaktionstechniken. Abschließend wurden dieser Prototyp hinsichtlich der Anforderungen ausgewertet.

Die finale Version der *PhysicBox* enthält die drei Anwendungen *Billard auf Eis* (siehe 3.3.1, Seite 17), *Bulldozer* (siehe 3.3.2, Seite 19) und *Planeten* (siehe 3.3.3, Seite 20). Die Realisierung dieser Anwendungen fußt zum Teil auf festgelegten Lehrplänen für den Schulunterricht (Government of Alberta, 1996) oder wie im Fall der Anwendung *Billard auf Eis* auch auf der Einführung neuer Technologie in den Alltag junger Menschen.

Des Weiteren wurden in diesem Kapitel die verwendeten sowie einige nicht verwendete Interaktionstechniken aufgelistet und erläutert. Viele der verwendeten Techniken basieren auf einfachen *Klick*- und *Drag & Drop*-Techniken. Die in der

Anwendungszeit sehr schnellen *Flick*- und *Proxy-Objekt*-Techniken werden hauptsächlich in Anwendungen eingesetzt, in denen Geschwindigkeit und nicht so sehr Präzision erforderlich ist. Hier ist es vor allem die Anwendung *Billard auf Eis*. Die *Slingshot*-Technik, abgeleitet von dem Prinzip einer Zwille (Katschi) oder auch eines Queues, erhöht die Trefferquote und Reproduzierbarkeit der Aktionen, senkt jedoch die Spielgeschwindigkeit. Die Zuordnung der Techniken zu *Open-Loop*- oder *Closed-Loop*-Techniken beschreibt den Unterschied zwischen „On-the-fly"- und „Turn-by-turn"-Techniken.

Die Auswertung zeigte Schwächen in den vorhandenen visuellen Rückmeldungen der Anwendungen auf. Die Erwartungen der Anwender in Bezug auf die Interaktionstechniken für das Vergrößern, Verkleinern und Verschieben in der Anwendung *Planeten* unterschied sich von den tatsächlich ausgewählten Techniken, was vielen Anwendern missfiel. Eine sehr interessante Frage, die sich aus diesem Teil der Studienarbeit ergibt, ist: Welche Auswirkungen hat der physische Unterschied zwischen Erwachsenen und Kindern (Körper-, Hand- und Fingergröße) bei der Verwendung digitaler Multitouch-Tabletops?

Der nächste Teil der Studienarbeit beschäftigt sich mit der Dokumention des Projektes.

# 4 IMPLEMENTIERUNG – ANWENDUNG UND FRAMEWORKS

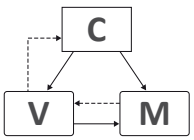
In der Entwicklung befindliche Projekte werden selten komplett neuentwickelt. Sie basieren in der Regel auf anderen schon vorhandenen Arbeiten. In der Softwareentwicklung sind Themen wie Modularisierung und Wiederverwendbarkeit von großer Bedeutung, denn in den verschiedenen Entwicklungsphasen Entscheidungen getroffen werden, die später für den Erfolg des Projektes entscheidend sein können. Das fängt bei der Wahl der Programmiersprache an, geht über die Paketstrukturen und ein Schema für die Benennung von Variablen und Konstanten und endet mindestens in der Frage der verwendeten Softwarelizenz.

In diesem Kapitel werden Softwarepakete aufgelistet, welche zur Ausführung, aber auch zur Weiterentwicklung der *PhysicBox* nötig sind. Diese Abhängigkeiten oder Mindestvoraussetzungen werden erläutert und begründet. Des Weiteren werden Schnittstellen beziehungsweise Frameworks vorgestellt, die aus diesem Projekt hervorgegangen sind und in gekapselter Form der *PhysicBox* beiliegen.

## 4.1 DIE PHYSICBOX

Der Prototyp *PhysicBox* wurde komplett in Java programmiert. Als Entwicklungsumgebung kam die *Eclipse IDE* zum Einsatz. Um die Erweiterbarkeit, Fehleranalyse und Wartung des Quellcodes zu erleichtern, wurde der Prototyp nach dem Model-View-Controller-Konzept, kurz MVC, umgesetzt. Dadurch sind Daten und grafische Ausgaben sowohl im Quellcode als auch im ausgeführten Programmcode voneinander getrennt. Nach MVC dient dabei das Modell als reines Datenobjekt. Im Modell sind keine grafischen Elemente oder Routinen enthalten. Ein View dient in der Regel der grafischen Ausgabe eines Modells, wobei es auch mehrere View-Objekte zu einem aktuellen Modell geben kann. Der Controller dient als Schnittstelle für Nutzereingaben und propagiert die entsprechenden Änderungen mindestens an das Modell. Ein Modell kann andere Modelle beinhalten, hat aber keine expliziten Informationen über ein View- und Controller Objekt. Lediglich indirekte Assoziationen zu View-Objekten sind erlaubt. In Java lässt sich dies beispielsweise durch das Listener-Prinzip realisieren. Ein View-Objekt kennt sein direktes Modell und kann indirekte Verknüpfungen zu Controllern besitzen. Ein Controller besitzt jeweils direkte Verbindungen zu einem View- und Modell-Objekt.

MVC




### 4.1.1 *Aufbau und Paketstruktur*

Der Quellcode des Prototypen ist in sieben Pakete unterteilt (Abb. 25). Neben den Ordnern für die einzelnen Anwendungen `pool` (für Billard), `bulldozer`, `planets` und `mainmenu` gibt es noch die Pakete `general`, für allgemeine und oft genutzte Klassen, `gui` für alle Dateien, die Klassen mit grafischen Ausgaben enthalten und das Paket `util` für Hilfsklassen.

Das der Prototyp nach dem Model-View-Controller-Prinzip entwickelt wurde, lässt sich an der Benennung der Dateien und Klassen ablesen. Beginnt der Datei- und Klassennamen mit einem *M*, so handelt es sich um ein Modell, bei einem *V*

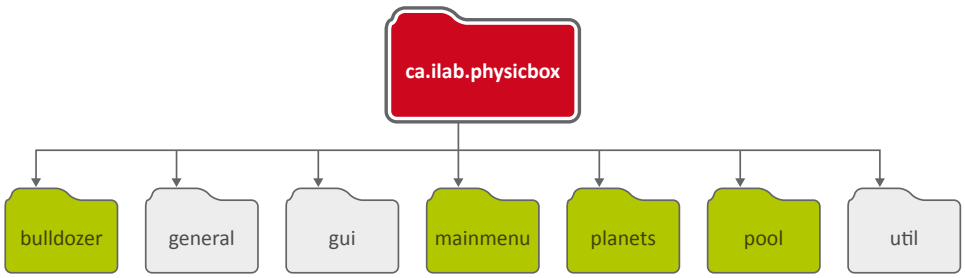



Abbildung 25: Interne *PhysicBox*-Struktur, Pakete

ist es die grafische Repräsentation eines Modells (View) und beginnt der Name mit einem *C*, stellt dies einen Controller dar.

### 4.1.2 *Konfiguration*

Die Einstellungen des Prototyps lassen sich mit Hilfe einer Konfigurationsdatei festlegen. Im Anhang A.1 wurde eine Standard-Konfigurationsdatei hinterlegt. Diese Datei wurde mit Kommentaren versehen und beschreibt alle möglichen Einstellungen für die *PhysicBox*. Sie ist in der fertigen Applikation im `data`-Ordner (nicht in der internen *PhysicBox*-Struktur) zu finden.

### 4.1.3 *Abhängigkeiten der PhysicBox*

*PhysicBox* wurde in Java geschrieben und setzt mindestens `Java 6` voraus. Des Weiteren nutzt *PhysicBox* die Bibliotheken *jtl* und *plight*. Diese und weitere Abhängigkeiten finden sich in der folgenden Liste.

- Java in der Version 6, inklusive Java2D (Standard)
- *jtl* und damit verbunden, die *TUIO*-Implementierung für Java
- *plight* als Zeichen-Bibliothek
- *JBox2D* für virtuelle physikalische Umgebungen (Version 2.0, `http://www.jbox2d.org/`)
- `vecmath.jar` aus der Java3D-Bibliothek (`http://java3d.dev.java.net/`)

## 4.2 ENTWICKELTE FRAMEWORKS

### 4.2.1 *jtl – Java Touch Library*

*jtl* ist eine kleine Bibliothek für die Anbindung verschiedener Multitouch-fähiger Geräte. Eines der Ziele des Projektes ist die Unterstützung und Verwendung unterschiedlicher Tabletops. In der aktuellen Version unterstützt die Bibliothek den Microsoft Surface, den SMART Table und TUIO-basierte Tabletops. Um eine möglichst breite Verwendung und Kompatibilitätsproblemen vorzubeugen, wurde *jtl* in Java geschrieben. Die Grundlage dieser Bibliothek bildete der Quellcode von Mark Hancock, Ph.D. Student bei Dr. Sheelagh Carpendale in der Innovis Group, University of Calgary, Alberta, Kanada. Die Konzepte des Quellcodes von Mark Hancock wurden übernommen, gekapselt und damit von dem vorliegendem Projekt gelöst. Einige Zeilen wurden übernommen und sind als solche auch gekennzeichnet.

Diese Bibliothek wurde auf Grund des Fehlens eines Standards und der Möglichkeit für Entwickler, ihre Anwendungen ohne Mehraufwand für unterschiedliche Multitouch-Tabletops zu entwickeln, implementiert und in den Arbeitsgruppen veröffentlicht. *jtl* wurde neben dem *PhysicBox*-Prototypen ebenfalls in einem weiteren studentischen Projekt erfolgreich eingesetzt.

*Prozess, Aufbau und Anwendung*

Da es derzeit keinen geräteübergreifenden Standard für die Übermittlung von Kontaktpunkten (Touchpunkten) gibt, große weltweit erfolgreiche Firmen wie Microsoft auch kein Interesse an der Ausarbeitung eines solchen Standards haben, müssen aufwändige Prozesse die spezifischen Daten generalisieren und über eine gemeinsame Schnittstelle propagieren.

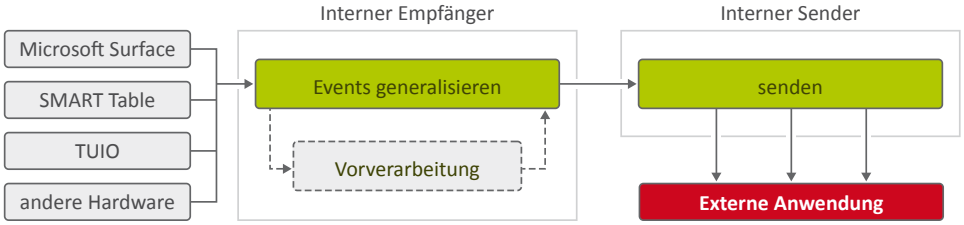



Abbildung 26: Allgemeiner Prozessablauf einer mit *jtl* realisierten Anwendung

*jtl* stellt alle nötigen Schnittstellen für die Entwicklung von Anwendungen für die unterstützten Multitouch-Tabletops zur Verfügung (Abb. 26). Dabei muss der Entwickler lediglich einen eigenen Event-Empfänger nach einer durch *jtl* gegebenen Schnittstelle implementieren, den *jtl*-Prozess starten und die in einem allgemeinen Format gesendeten Events verarbeiten. *jtl* übernimmt dabei den Empfang der Kontaktdaten (Touchdaten) von der ausgewählten Hardware, überführt diese in ein allgemeines Format, führt auf Anforderung eine Vorverarbeitung nach Kontakttyp (Touchtyp) durch und sendet schließlich die gesammelten Daten in dem schon erwähnten einheitlichen Format.

*jtl* beinhaltet für jeden unterstützten Multitouch-Tabletop einen eigenen Event-Empfänger, im Package devices (Abb. 27), der die Events abfängt und dann weiterverarbeitet. Ein weiteres Modul (`class: TouchDB` in library (Abb. 27)) sammelt diese Kontaktdaten und dient als Datenbank für die Events. Bei dieser Datenbank muss durch den Entwickler ein Empfänger registriert werden (`TouchDB.addListener( "developers receiver");`), der die Schnittstelle `TouchDBListener` im Paket `library` (Abb. 27) implementiert.

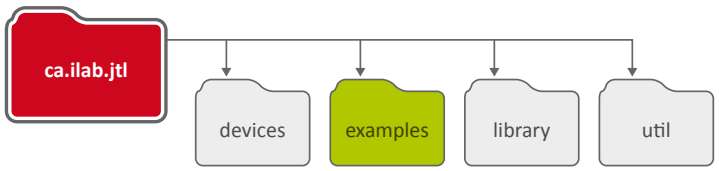



Abbildung 27: Interne *jtl*-Struktur, Pakete

Das Paket `util` (Abb. 27) enthält Hilfsklassen sowie mathematische Konstanten. Im Ordner `examples` liegen drei Minimalbeispiele.

*Kontaktdaten (Touchdaten)*

Bestandteil der Bibliothek *jtl* ist ein allgemeines, von der Hardware entkoppeltes Format für Kontaktpunkte (Touchpunkte). Abbildung 28 liefert eine Übersicht der

zur Verfügung stehenden Eigenschaften eines Kontaktpunktes. Die folgende Liste beschreibt die Eigenschaften.

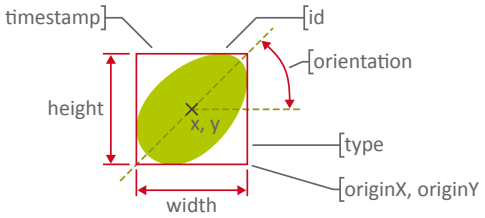



Abbildung 28: Durch *jtl* zur Verfügung stehende Eigenschaften eines Kontaktpunktes, `id`, `x`, `y`, `orientation`, `width`, `height`, `originX`, `originY`, `timestamp` und `type`

ID eine eindeutige Identifikationsnummer des Kontaktpunktes

X,Y die Koordinaten des Kontaktpunktes, sowohl relativ als auch absolut (in Pixel)

ORIENTATION die mathematische Ausrichtung des Kontaktpunktes, 0 oder $2\pi$ entspricht von links nach rechts, $\frac{\pi}{2}$ ist von unten nach oben, $\pi$ ist von rechts nach links, $\frac{3\pi}{2}$ ist von oben nach unten

WIDTH, HEIGHT die Breite und Höhe der Bounding-Box eines Kontaktpunktes, sowohl relativ als auch absolut (in Pixel)

ORIGINX, ORIGINY die Koordinaten des ersten Auftretens des Kontaktpunktes, sowohl relativ als auch absolut (in Pixel)

TIMESTAMP der Zeitstempel des ersten Auftretens des Kontaktpunktes, in Millisekunden

TYPE die Art des Kontaktpunktes (`pressed`, `dragged`, `clicked`, `released`)

*Abhängigkeiten*

Derzeit besitzt *jtl* nur eine direkte Abhängigkeit einer anderen Bibliothek – die der *TUIO*-Implementierung für Java. Natürlich ist *jtl* aber auch abhängig von den Event-Services der entsprechenden Tabletops. Das heißt, es ist zwingend erforderlich, beispielsweise den „SMART Table Service“ auszuführen, bevor *jtl* seinen Dienst startet. Daraus ergibt sich die folgende Liste:

- *TUIO*-Implementierung für Java (*TUIO* Client Reference Implementations) TUIO_JAVA.zip auf `http://www.tuio.org/?software` (zuletzt aufgerufen am 14.07.2010)
- Hardwareabhängigkeiten:
  - für *SMART Table*: der „SMART Table Service“ muss im Hintergrund laufen
  - für *Microsoft Surface*: setzt eine spezielle Socket-Anwendung voraus, welche die Events als XML-Stream weiterleitet
  - für *TUIO-Tabletops*: setzt eine TUIO-kompatible Anwendung vorraus, welche die TUIO-Events generiert, zum Beispiel CCV (Community Core Vision, `http://ccv.nuigroup.com/`)

### 4.2.2 *plight – Java2D Processing Renderer*

*plight* ist ein Java2D-Grafik-Renderer, der die Zeichenfunktionen der beliebten Java-Bibliothek *Processing* (`http://processing.org/`) implementiert. Daher auch der Name *plight*, *p* für *Processing* und *light* für leicht, verkleinert und verringerter Umfang. Weil es während der Entwicklung der PhysicBox zu Problemen zwischen *Processing* und dem *SMART Table SDK* kam, wurden mit *plight* die Standard-Zeichenmethoden *Processing*s neu implementiert. Bei der Entwicklung der Bibliothek wurde die *Eclipse IDE* verwendet. Entsprechende Projektdateien liegen der DVD in A.2 bei.

Diese neue Bibliothek ist wesentlich kleiner als *Processing* und zeichnet sich durch eine bessere Performance aus. Es wurden bei weitem nicht alle *Processing*-Funktionen implementiert, viel mehr wurden nur die nötigsten Funktionen nachgebildet.

*plight* ist also eine kleine 2D-Zeichen-Bibliothek. Die Funktionsnamen wurden exakt aus der *Processing*-API übernommen. So können Entwickler ohne großen Änderungsaufwand zwischen beiden Bibliotheken wechseln.

#### *Methoden*

Da die Funktionen der *Processing*-API nachempfunden sind, ist die Dokumentation dieser API (`http://processing.org/reference/`) ebenfalls für die in *plight* vorhandenen Funktionen gültig. Die folgende Liste bietet einen Überblick der wichtigsten Funktionen.

| | |
|---|---|
| `background()` | füllt den kompletten Hintergrund mit einer Farbe |
| `ellipse()` | zeichnet eine Ellipse |
| `image()` | zeichnet eine Grafikdatei |
| `line()` | zeichnet eine Linie |
| `point()` | zeichnet einen Punkt |
| `rect()` | zeichnet ein Rechteck |
| `triangle()` | zeichnet ein Dreieck |
| `text()` | gibt einen Text aus |
| `pushMatrix()` | speichert die aktuelle Transformations-Matrix und wechselt dann zur Einheits-Matrix |
| `popMatrix()` | wechselt zur vorherigen Transformations-Matrix zurück |
| `fill()` | bestimmt die Füllfarbe für weitere Zeichenaktionen |
| `stroke()` | bestimmt die Konturfarbe für weitere Zeichenaktionen |
| `strokeWidth` | setzt die Konturstärke für weitere Zeichenaktionen |
| `translate()` | führt eine Translation auf der aktuellen Transformations-Matrix aus |
| `rotate()` | führt eine Rotation auf der aktuellen Transformations-Matrix aus |

#### *Abhängigkeiten*

*plight* ist derzeit nur von der lokalen Java-Installation abhängig. In dieser ist standardmäßig Java2D enthalten. *plight* wurde jedoch nur mit `Java 6` getestet und setzt daher mindestens diese Version 6 voraus.

## 4.3 ZUSAMMENFASSUNG

In diesem Teil der Studienarbeit wurden die Implementierungen des Prototypen dokumentiert. Die *PhysicBox* wurde in Java und nach dem MVC-Konzept entwickelt. Durch die Entwicklung der eigenständigen Bibliothek *jtl* ist die Anwendung auf dem SMART Table, dem Microsoft Surface und TUIO-basierten Tabletops ausführbar. Der Prototyp setzt Java 6 und die Physik-Engine JBox2D voraus.

Im nächsten Teil der Studienarbeit wird die gesamte Arbeit zusammengefasst und ein Ausblick auf weitere Arbeiten gegeben.

# 5 ZUSAMMENFASSUNG UND AUSBLICK

Dieser Teil der Studienarbeit fasst die Ergebnisse der Studienarbeit zusammen und verbindet sie mit der Problem- und Aufgabenstellung der Einleitung. Der zweite Abschnitt beschreibt Möglichkeiten weiterer Forschungsarbeiten auf Basis dieses Projektes und gibt somit einen Ausblick zukünftiger Projekte.

## 5.1 ERGEBNISSE

Diese Studienarbeit hat gezeigt, dass in einem Zeitraum von knapp sechs Monaten gut drei Anwendungen mit didaktischem Inhalt realisiert werden und viele weitere Ideen entstehen können. Von den geplanten fünf Mini-Anwendungen wurden während des Praktikums drei umgesetzt. Eine kleine, fast vierte Anwendung dient als Startmenü und Ausgangspunkt der Anwender.

In den Anwendungen kommen verschiedene Interaktionstechniken, ausgehen von früheren Forschungsarbeiten zum Einsatz. Ob diese sich ebenfalls im Klassenzimmer, also für den Einsatz in Anwendungen für Kinder eignen, bleibt aber durch diese Studienarbeit unbeantwortet (siehe 5.2, Seite 40).

Die realisierten Anwendungen *Billard auf Eis* (siehe 3.3.1, Seite 17) und *Planeten* (siehe 3.3.3, Seite 20) demonstrieren, wie der Schulunterricht bereichert und ein Lehrer einen Tabletop mit diesen Anwendungen als sinnvolle Ergänzung zum Unterricht nutzen kann. Hierzu empfiehlt es sich, einige der Anwenderanmerkungen aus 3.5.1 zu realisieren beziehungsweise in das System zu integrieren. Mit Hilfe des Lernmodus der *Billard*-Anwendungen kann der Lehrer Prinzipien, wie die Kollision von Objekten, Kraftübertragung, Elastischer Stoß oder auch Rotationen um die eigene Achse erklären. Sollen die Kinder erst einmal die Prinzipien spielerisch erfahren, dann startet der Lehrer einfach den Spaßmodus der Anwendung. Die *Planeten*-Anwendung eignet hingegen als Versuchslabor oder Experiment für den Physik- oder Astronomieunterricht. Der Lehrer kann mit Hilfe der Größenunterschiede der Planeten aber auch durch die Visualisierung der Flugbahn der Planeten die Grundlagen der Gravitation erläutern.

In Abschnitt 1.2 der Studienarbeit wurden Aufgabenstellungen und Bewertungskriterien zu diesem Projekt formuliert. In dem folgenden Teil werden nun, unter Einbeziehung der Kriterien, die Ergebnisse der einzelnen Aufgaben aufgezeigt.

### *Das Projekt als Einblick und Designstudie zum Thema „Physik-Engines in Tabletop-Anwendungen“*

In dieser Studienarbeit wurden wichtige frühere Forschungen zusammengetragen (siehe 2, Seite 5) und strukturiert miteinander verknüpft. Das Fundament der Arbeit wurde maßgeblich durch die Arbeiten von Wilson u. a. (2008) und Hancock u. a. (2009) geprägt. Die Implementierung nutzt eine Physik-Engine, um das Verhalten der virtuellen Objekte zu kontrollieren. Die Interaktionstechniken wurden nach dem Konzept der *Sticky Tools* ausgewählt und angewendet.

Die drei entwickelten Anwendungen geben einen guten Einblick in die Möglichkeiten, welche durch die Verwendung von Physik-Engines entstehen. Die Phase der Ideenfindung (siehe 3.2, Seite 12) brachte viele interessante und teils neue

Anwendungsideen der in der Literatur erwähnten Techniken hervor. Diese Ideen können als Grundlage weiterer Projekte und Forschungsarbeiten dienen.

*Der Prototyp als Grundlage einer Studie im „Klassenzimmer der Zukunft“*

Der implementierte Prototyp lässt sich auf unterschiedlichen Plattformen (siehe 4.1, Seite 33), vor allem aber dem `SMART Table`, `Microsoft Surface` und `TUIO-basierten Tabletops`, ausführen. Aus der Anwendung des Model-View-Controller-Konzeptes in der Entwicklung des Prototypen, resultiert eine flache und einfache Codestruktur (siehe 4.1.1, Seite 33). Das unterstützt das Verständnis und die Wartung des Quellcodes. Jede der implementierten Anwendungen kann dabei als Vorlage für die Erstellung einer neuen Anwendung dienen. Dazu muss lediglich ein Anwendungs-Ordner (`pool`, `bulldozer` oder `planets`) kopiert, der Inhalt entsprechend angepasst und dann mit der *Fließband-Anwendung* verknüpft werden.

Während des Berufspraktikums wurde intensiv an der Vorbereitung einer Studie gearbeitet. In dieser Studie sollte das Verhalten von Kindern (4 Gruppen zu je 3-4 Kindern) mit Video und Ton aufgezeichnet und anschließend analysiert werden. Dabei sollten die Kinder im Alter von 8-10 Jahren sein. Die Durchführung der einzelnen Versuche würden von einem Lehrer begleitet werden. Interessiert an dem Verhalten der Kinder im allgemeinen, Verhalten mit dem Tabletop oder beispielsweise der Gruppendynamik während des Spielens, sollte die Studie durch einfache Beobachtungen eine Grundlagen für die Entwicklung weiterer Tabletop-Anwendungen schaffen.

*Aufgezeigte Fragen und Themen von wissenschaftlichem Interesse*

Die Studienarbeit hat weitere interessante Fragen und andere mögliche Themen aufgezeigt (siehe 3.5.2, Seite 30). Diese können durch die folgenden thematischen Überschriften zusammengefasst werden.

1. „Globale Operationen“: Entwicklung von Richtlinien und Hinweisen für die Behandlung und Verwendung von globalen Operationen;
2. Vorraussetzungen für die Entwicklung von Anwendungen mit didaktischem Hintergrund;
3. Kinder und der Einsatz von Multitouch-Tabletops;
4. Interaktionstechniken in der Analyse;

Diese Themen sind auch für den nächsten Abschnitt dieses Teils der Studienarbeit grundlegend und bieten einen Ausblick auf mögliche Erweiterungen dieser Arbeit.

## 5.2 ERWEITERUNGEN UND AUSBLICK

Diese Studienarbeit und das beschriebene Projekt lassen sich durch verschiedene Frage- und Aufgabenstellungen erweitern. Die Anzahl dieser Fragen und weiterführenden Themen ist so Umfangreich, dass damit gleich mehrere studentische Projekte zu realisieren wären.

*Analyse und Vergleich der Interaktionstechniken*

Unter Berücksichtigung des Alters der Zielgruppe dieser Prototyp-Anwendung, stellt sich die Frage, ob Ergebnisse aus früheren Forschungsarbeiten, die Interaktionstechniken analysieren, ohne messbaren Unterschied für Kinder gelten und zu übernehmen sind. Es ist interessant zu prüfen, ob Kinder, die in der Regel über keine langjährige Erfahrung mit Technik und deren Bedienung verfügen, andere Lösungsansätze und auch Interaktionstechniken finden und favorisieren.

Ein anderer wichtiger Aspekt der Interaktionstechniken ist deren Einfluss auf den Anwender. Wie verändert sich das Verhalten oder die Lösungsstrategien, wenn Interaktionstechniken modifiziert oder eingeschränkt werden? Im Bereich der Anwendungen mit didaktischem Hintergrund oder Anwendungen für Kinder ist interessant, ob die Wahl oder genaue Umsetzung der Interaktionstechnik einen Einfluss auf beispielsweise das Spaßempfinden oder die Ausdauer der Kinder hat.

*Gestaltung eines Game-Authoring-Werkzeuges*

Der vorliegende Prototyp könnte ebenfalls als Grundlage eines mächtigeren Game-Authoring-Werkzeuges dienen. Dabei gilt es, die verwendeten Methoden zu abstrahieren und zu verallgemeinern. Diese Anwendung könnte dann als Grundlage für studentische Projekte, Anwendungen für Studien und Spielentwicklung auf Tabletops dienen. Um die Flexibilität eines solchen Werkzeuges zu gewährleisten, sollte der Einsatz einer 3D-Physik-Engine überdacht und gegebenenfalls in das System integriert werden.

Ein solches Werkzeug könnte auch Lehrer bei der Gestaltung und Anreicherung des Unterrichtes unterstützen, indem sie die kleinen Anwendungen mit einfachen Mitteln selbst gestalten können. Lehrer könnten die Ergebnisse dann als Abwechslung für die Schüler, für Gruppenarbeiten oder auch als „Experimentelle Fabrik und Versuchsstätte“ nutzen. Für die Erstellung einer solchen Anwendung ist die weitere und intensive Zusammenarbeit mit Lehrern unabdingbar. Auch müssen Altersunterschiede der Schüler sowie Unterschiede in den Lehrplänen der Regionen und sogar der Länder berücksichtigt werden. Dieser Vorschlag resultiert sicher in einem interdisziplinären Projekt, da hier viele Fachfremde Aspekte (Stichwort *Educational Software*) von Bedeutung sind. Großartige Kompetenzen sowie viele Forschungsarbeiten zu diesem Thema finden sich im Forschungsbereich der *Digital Game Studies*.

# ANHANG

## A.1 STANDARD KONFIGURATIONSDATEI

```
! JAVA .properties file
#
#
#

! WINDOW
window_width = 1024
window_height = 768
window_undecorated = true

! MULTI-TOUCH
! uncomment one of the following lines to set a certian touch input source
touch_input = TUIO
# touch_input = SMART_NEW
# touch_input = SMART_OLD
# touch_input = SURFACE

! APPLICATION
! uncomment one of the following lines to set a certian starter application
starter_app = APP_MAIN_MENU
# starter_app = APP_BULLDOZER
# starter_app = APP_PLANETS
# starter_app = APP_POOL_FUN
# starter_app = APP_POOL_LEARN

! POOL APP
! use values between 0...3 for pockets_x and 0...2 for pockets_y
pool_pockets_x = 1
pool_pockets_y = 0
! use value between 1...4 for teams and 1...10 for team_balls
pool_teams = 2
pool_team_balls = 4

! BULLDOZER APP
! use values between 1...10 for waste_level
waste_level = 4
```

## A.2 DVD mit Programm, Quellcode, Dokumentation und Studienarbeit

Die DVD enthält die folgenden Inhalte:

- *jtl-src*: Ordner mit dem Quellcode der Bibliothek *jtl*
- *PhysicBox-WIN*: Ordner mit einer ausführbaren Variante der *PhysicBox* für Windows
- *PhysicBox-src*: Ordner mit dem Quellcode der *PhysicBox*
- *plight-src*: Ordner mit dem Quellcode der Bibliothek *plight*
- *Studienarbeit*: Ordner mit den LaTeX $2_\varepsilon$ Quellen dieser Arbeit
- *2010-07-23_Studienarbeit-Ricardo-Langner.pdf*: diese Studienarbeit als digitale Version mit interen und farbigen Links
- *2010-07-23_Studienarbeit-Ricardo-Langner_PRINT.pdf*: genau diese Studienarbeit (ohne farbige Links), geeignet für den Druck
- *PhysicBox.mov*: ein kleines Video über die *PhysicBox*

[Agarawala u. Balakrishnan 2006] Agarawala, Anand ; Balakrishnan, Ravin: Keepin' it real: pushing the desktop metaphor with physics, piles and the pen. In: *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*. New York, NY, USA : ACM, 2006. – ISBN 1–59593–372–7, 1283–1292 (Zitiert auf Seite 7.)

[BumpTop 2010] BumpTop: An important BumpTop announcement: End of Life. (2010). http://eol.bumptop.com/, Abruf: 16.07.2010 (Zitiert auf Seite 7.)

[Cao u. a. 2008] Cao, Xiang ; Wilson, Andrew D. ; Balakrishnan, Ravin ; Hinckley, Ken ; Hudson, Scott E.: ShapeTouch: Leveraging contact shape on interactive surfaces. In: *Tabletop*, IEEE, 2008, S. 129–136 (Zitiert auf Seiten 8 und 9.)

[Catto 2010] Catto, Erin: Box2D Physics Engine. (2010). http://www.box2d.org/, Abruf: 16.07.2010 (Zitiert auf Seite 13.)

[Dietz u. Leigh 2001] Dietz, Paul ; Leigh, Darren: DiamondTouch: a multi-user touch technology. In: *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*. New York, NY, USA : ACM, 2001. – ISBN 1–58113–438–X, 219–226 (Zitiert auf Seite 5.)

[Elrod u. a. 1992] Elrod, Scott ; Bruce, Richard ; Gold, Rich ; Goldberg, David ; Halasz, Frank ; Janssen, William ; Lee, David ; McCall, Kim ; Pedersen, Elin ; Pier, Ken ; Tang, John ; Welch, Brent: Liveboard: a large interactive display supporting group meetings, presentations, and remote collaboration. In: *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY, USA : ACM, 1992. – ISBN 0–89791–513–5, S. 599–607 (Zitiert auf Seite 5.)

[Engel 2009] Engel, Michael: Unendliche Weiten im Unterricht. In: *Deutschlandfunk - PISAPlus* (2009). http://www.dradio.de/dlf/sendungen/pisaplus/1085982/, Abruf: 9.07.2010 (Zitiert auf Seite 20.)

[Ewjordan u. a. 2010] Ewjordan ; Quixotearg ; Dmurphy: JBox2D: Java port of Box2D physics engine. (2010). http://www.jbox2d.org/, Abruf: 16.07.2010 (Zitiert auf Seite 13.)

[Frisch u. a. 2009] Frisch, Mathias ; Heydekorn, Jens ; Dachselt, Raimund: Investigating multi-touch and pen gestures for diagram editing on interactive surfaces. In: *ITS '09: Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. New York, NY, USA : ACM, 2009. – ISBN 978–1–60558–733–2, S. 149–156 (Zitiert auf Seite 9.)

[Geißler 1998] Geissler, Jörg: Shuffle, throw or take it! Working Efficiently with an Interactive Wall. (1998). http://dx.doi.org/10.1.1.47.5980. – DOI 10.1.1.47.5980 (Zitiert auf Seiten 8 und 9.)

[Government of Alberta 1996] Government of Alberta: Elementary Science Curriculum, Alberta, Canada. In: *Government of Alberta - Programs of Study* (1996). http://education.alberta.ca/teachers/program/science/programs.aspx (Zitiert auf Seiten 13, 19, 20 und 31.)

[Han 2005] Han, Jefferson Y.: Low-cost multi-touch sensing through frustrated total internal reflection. In: *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*. New York, NY, USA : ACM, 2005. – ISBN 1–59593–271–2, 115–118 (Zitiert auf Seite 6.)

[Hancock u. a. 2009] Hancock, Mark ; Cate, Thomas ten ; Carpendale, Sheelagh: Sticky tools: full 6DOF force-based interaction for multi-touch tables. In: *ITS '09: Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. New York, NY, USA : ACM, 2009, S. 133–140 (Zitiert auf Seiten 3, 8, 9, 10, 26 und 39.)

[Hancock u. a. 2010] Hancock, Mark ; Cate, Thomas ten ; Carpendale, Sheelagh ; Isenberg, Tobias: Supporting sandtray therapy on an interactive tabletop. In: *CHI '10: Proceedings of the 28th international conference on Human factors in computing systems*. New York, NY, USA : ACM, 2010. – ISBN 978–1–60558–929–9, S. 2133–2142 (Zitiert auf Seite 8.)

[Hancock u. a. 2006] Hancock, Mark S. ; Vernier, Frederic D. ; Wigdor, Daniel ; Carpendale, Sheelagh ; Shen, Chia: Rotation and Translation Mechanisms for Tabletop Interaction. In: *TableTop2006: Proceedings of IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, IEEE Computer Society, 2006, S. 79–86 (Zitiert auf Seiten 27 und 28.)

[Hascoët 2003] Hascoët, M.: Throwing models for large displays. In: *In Proceedings of HCI'03* (2003) (Zitiert auf Seiten 3, 8, 9 und 25.)

[Hinrichs u. a. 2006] Hinrichs, Uta ; Carpendale, Sheelagh ; Scott, Stacey D.: Evaluating the effects of fluid interface components on tabletop collaboration. In: *AVI '06: Proceedings of the working conference on Advanced visual interfaces*. New York, NY, USA : ACM, 2006. – ISBN 1–59593–353–0, 27–34 (Zitiert auf Seiten 8 und 9.)

[Johnson u. Fryberger 1972] Johnson, Ralph G. ; Fryberger, David: Touch actuable data input panel assembly. In: *U.S. Patent 3,673,327* (1972) (Zitiert auf Seite 6.)

[Kaltenbrunner u. Bencina 2005] Kaltenbrunner, Martin ; Bencina, Ross: reacTIVision: a toolkit for tangible multi-touch surfaces. (2005). http://reactivision.sourceforge.net/, Abruf: 16.07.2010 (Zitiert auf Seite 6.)

[Kravchik 2008] Kravchik, Yuri: JPhysX: Java wrapper to the PhysX engine. (2008). http://www.jphysx.com/, Abruf: 16.07.2010 (Zitiert auf Seite 13.)

[Kruger u. a. 2004] Kruger, Russell ; Carpendale, Sheelagh ; Scott, Stacey D. ; Greenberg, Saul: Roles of Orientation in Tabletop Collaboration: Comprehension, Coordination and Communication. In: *Comput. Supported Coop. Work* 13 (2004), Nr. 5-6, 501–537. http://dx.doi.org/http://dx.doi.org/10.1007/s10606-004-5062-8. – DOI http://dx.doi.org/10.1007/s10606–004–5062–8. – ISSN 0925–9724 (Zitiert auf Seite 9.)

[Kruger u. a. 2005] Kruger, Russell ; Carpendale, Sheelagh ; Scott, Stacey D. ; Tang, Anthony: Fluid integration of rotation and translation. In: *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY, USA : ACM, 2005. – ISBN 1–58113–998–5, 601–610 (Zitiert auf Seiten 8 und 9.)

[Liu u. a. 2006] Liu, Jun ; Pinelle, David ; Sallam, Samer ; Subramanian, Sriram ; Gutwin, Carl: TNT: improved rotation and translation on digital tables. In: *GI '06: Proceedings of Graphics Interface 2006*. Toronto, Ont., Canada, Canada : Canadian Information Processing Society, 2006. – ISBN 1–56881–308–2, 25–32 (Zitiert auf Seiten 8 und 9.)

[Matsushita u. Rekimoto 1997] Matsushita, Nobuyuki ; Rekimoto, Jun: HoloWall: designing a finger, hand, body, and object sensitive wall. In: *UIST '97: Proceedings of the 10th annual ACM symposium on User interface software and technology*. New York, NY, USA : ACM, 1997. – ISBN 0–89791–881–9, S. 209–210 (Zitiert auf Seite 5.)

[Microsoft Corporation 2007] Microsoft Corporation: Microsoft Surface. (2007). http://www.microsoft.com/surface/, Abruf: 16.07.2010 (Zitiert auf Seite 6.)

[Mogel 1994] Mogel, Hans: *Psychologie des Kinderspiels*. 2. Auflage. Springer-Verlag, 1994. – ISBN 3–540–58088–3 (Zitiert auf Seite 17.)

[Nortd Labs 2008] Nortd Labs: TouchKit API. (2008). http://labs.nortd.com/touchkit/download/, Abruf: 16.07.2010 (Zitiert auf Seite 6.)

[NUI Group Community 2009] NUI Group Community: Community Core Vision. (2009). http://ccv.nuigroup.com/, Abruf: 16.07.2010 (Zitiert auf Seite 6.)

[NVIDIA GmbH 2010] NVIDIA GmbH: NVIDIA PhysX: Physics-Engine. (2010). http://www.nvidia.de/object/nvidia_physx_de.html, Abruf: 16.07.2010 (Zitiert auf Seite 12.)

[Pedersen u. a. 1993] Pedersen, Elin R. ; McCall, Kim ; Moran, Thomas P. ; Halasz, Frank G.: Tivoli: an electronic whiteboard for informal workgroup meetings. In: *CHI '93: Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems*. New York, NY, USA : ACM, 1993. – ISBN 0–89791–575–5, S. 391–398 (Zitiert auf Seite 5.)

[Reetz u. a. 2006] Reetz, Adrian ; Gutwin, Carl ; Stach, Tadeusz ; Nacenta, Miguel ; Subramanian, Sriram: Superflick: a natural and efficient technique for long-distance object placement on digital tables. In: *GI '06: Proceedings of Graphics Interface 2006*. Toronto, Ont., Canada, Canada : Canadian Information Processing Society, Jan 2006, S. 163–170 (Zitiert auf Seiten 8, 9, 23 und 24.)

[Shen u. a. 2003] Shen, Chia ; Lesh, Neal ; Vernier, Frédéric: Personal digital historian: story sharing around the table. In: *interactions* 10 (2003), Nr. 2, 15–22. http://dx.doi.org/http://doi.acm.org/10.1145/637848.637856. – DOI http://doi.acm.org/10.1145/637848.637856. – ISSN 1072–5520 (Zitiert auf Seite 8.)

[SMART Technologies 2009] SMART Technologies: SMART Table: interactive learning center. (2009). http://smarttech.com/table, Abruf: 16.07.2010 (Zitiert auf Seite 6.)

[Staatliches Gymnasium Am Lindenberg Ilmenau 2009] Staatliches Gymnasium Am Lindenberg Ilmenau: Notebookklassen – Schule der Zukunft. (2009). http://www.gym-amlindenberg.de/joomla/index.php?option=com_content&view=article&id=72&Itemid=124, Abruf: 16.07.2010 (Zitiert auf Seite 1.)

[Streitz u. a. 1994] Streitz, Norbert A. ; Geissler, Jörg ; Haake, Jörg M. ; Hol, Jeroen: DOLPHIN: integrated meeting support across local and remote desktop environments and LiveBoards. In: *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*. New York, NY, USA : ACM, 1994. – ISBN 0–89791–689–1, S. 345–358 (Zitiert auf Seite 5.)

[Swedish Game Development AB 2009] Swedish Game Development AB: Traffic Rush. (2009). http://itunes.apple.com/at/app/traffic-rush/id322423174?mt=8, Abruf: 16.07.2010 (Zitiert auf Seite 16.)

[Ullmer u. Ishii 1997] Ullmer, Brygg ; Ishii, Hiroshi: The metaDESK: models and prototypes for tangible user interfaces. In: *UIST '97: Proceedings of the 10th annual ACM symposium on User interface software and technology*. New York, NY, USA : ACM, 1997. – ISBN 0–89791–881–9, S. 223–232 (Zitiert auf Seite 5.)

[Underkoffler u. Ishii 1999] Underkoffler, John ; Ishii, Hiroshi: Urp: a luminous-tangible workbench for urban planning and design. In: *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY, USA : ACM, 1999. – ISBN 0–201–48559–1, 386–393 (Zitiert auf Seiten 6, 7 und 10.)

[University of Augsburg 2007] University of Augsburg: TWING: framework for tangible and multitouch interface based applications. (2007). http://xenakis.origo.ethz.ch/, Abruf: 16.07.2010 (Zitiert auf Seite 6.)

[Ushakov 2009] Ushakov, Fedor G.: Popular Physics Engines comparison: PhysX, Havok and ODE. (2009). http://physxinfo.com/articles/?page_id=154, Abruf: Add data for field: Lastchecked (Zitiert auf Seiten 6 und 7.)

[Wallin 2008] Wallin, David: Tank Game for CCV Flash Examples. In: *http://ccv.nuigroup.com/* (2008) (Zitiert auf Seite 20.)

[White 1965] White, W.: Method for Optical Comparison of Skin Friction-Ridge Patterns. In: *U.S. Patent 3,200,701* (1965) (Zitiert auf Seite 6.)

[Wilson 2005] Wilson, Andrew D.: PlayAnywhere: a compact interactive tabletop projection-vision system. In: *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*. New York, NY, USA : ACM, 2005. – ISBN 1–59593–271–2, 83–92 (Zitiert auf Seite 6.)

[Wilson 2009] Wilson, Andrew D.: Simulating Grasping Behavior on an Imaging Interactive Surface. In: *ITS '09: Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. New York, NY, USA : ACM, 2009, S. 125–132 (Zitiert auf Seiten 8 und 9.)

[Wilson u. a. 2008] Wilson, Andrew D. ; Izadi, Shahram ; Hilliges, Otmar ; Garcia-Mendoza, Armando ; Kirk, David: Bringing physics to the surface. In: *UIST '08: Proceedings of the 21st annual ACM symposium on User interface software and technology*. New York, NY, USA : ACM, Jan 2008, 67–76 (Zitiert auf Seiten 3, 7, 10, 24 und 39.)

[Wu u. Balakrishnan 2003] Wu, Mike ; Balakrishnan, Ravin: Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. In: *UIST '03: Proceedings of the 16th annual ACM symposium on User interface software and technology*. New York, NY, USA : ACM, 2003. – ISBN 1–58113–636–6, 193–202 (Zitiert auf Seiten 8 und 9.)

KOLOPHON

Diese Arbeit wurde im Zeitraum März bis Juli 2010 mit LaTeX 2ε erstellt. Für die Gestaltung, das Layout und die Typografie zeichnen sich die Stile *classicthesis*, *arsclassica* und kleinere eigenverantwortliche Modifikationen verantwortlich.

# ERKLÄRUNG

Hiermit versichere ich, dass ich die Studienarbeit mit dem Titel *Interaktive Tabletop-Anwendungen zum Lernen physikalischer Grundlagen* selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Die Stellen meiner Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken entnommen sind, habe ich in jedem Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht. Dasselbe gilt sinngemäß für Tabellen und Abbildungen. Diese Arbeit hat in dieser oder einer ähnlichen Form noch nicht im Rahmen einer anderen Prüfung vorgelegen.

Hiermit erkläre ich mich einverstanden, dass meine Studienarbeit als pdf-Datei auf der Internetseite der Arbeitsgruppe *User Interface & Software Engineering* der Öffentlichkeit zur Verfügung gestellt wird.

*Magdeburg, 23. Juli 2010*

Ricardo Langner