
run
KLOTZEN

Masterprojekt
Masterstudiengang Interaktive Mediensysteme
Sommersemester 2009
Hochschule Augsburg

Datum: 29.07.2009

Autoren:
Dennis Barth
Melanie Friedrich
Christian Kandler
Birgit Seebauer

Betreuer:
Prof. KP Ludwig John
Prof. Thomas Rist

vorwort

Dieser Bericht dokumentiert die Konzeption und Umsetzung des Spiels runKlotzen, das im Rahmen einer Projektarbeit im Sommersemester 2009 im Masterstudiengang Interaktive Mediensysteme an der Hochschule Augsburg entstanden ist.

inhaltsverzeichnis

1. einleitung

1.1 Motivation

Interaktive Tische sind derzeit in aller Munde und bieten eine neue Plattform der Kommunikation. Allerdings dienen sie meist dem kollaborativen Arbeiten eines Teams oder dem Austausch von Medien, sprich Fotos oder Videos. Dabei liegt es nahe, die gesellige Komponente eines Tisches mehr auszureizen. Schließlich ist es schon immer ein Ort gewesen, an dem Menschen zusammentreffen und sich unterhalten. Was könnte also geselliger und unterhaltsamer sein als ein kurzweiliges Brettspiel? Die Idee: ein Spielkonzept entwickeln, das die Technik interaktiver Tische als Grundlage nutzt.

1.2 Zielsetzung

Das Ziel des Masterprojekts „runKlotzen“ ist die Konzeption und Umsetzung eines Brettspiels, dessen Basis ein interaktiver Tisch bildet. Allerdings erfolgt die Bedienung nicht direkt per Touch, sondern es wird auf eine bekanntere Form der Interaktion zurückgegriffen. Während eine taktile Bedienung eine gewisse Lernphase fordert, da noch nicht jeder damit vertraut ist, findet eine Metapher Anwendung, die der Tisch selbst mit sich bringt – man stellt etwas darauf. Folglich wird die Eintrittsbarriere heruntergesetzt, da der Spieler wie mit einem gewöhnlichen Gesellschaftsspiel interagiert. Das Ergebnis sind Spielfiguren aus Holz auf einer projizierten Spielfläche, die im ständigen Wechselspiel zueinander stehen. Eine gelungene Kombination aus digitalen und realen Elementen. Der Mehrwert entsteht durch eine audiovisuelle Rückmeldung, die das Spielerlebnis intensiviert.

2.1 Bestehende Projekte

2.1.1 Reactable

http://www.reactable.com/

Bei Rectable werden mehrere „Mäuse“ auf einem Tisch durch die eigene Hand umher geschoben und damit Klangereignisse ausgelöst. Aus technischer Sicht befindet sich eine Kamera und ein Beamer unterhalb des Tisches. Die Mäuse werden gescannt und ihre Drehachse erkannt. Dadurch werden bestimmte Vorgänge ausgelöst, die auch durch Drehen der Mäuse verändert werden können. Die beweglichen Mäuse werden Tangibles (lat. tangere = berühren) genannt. Dieser Begriff stammt aus der Welt des Bogenschießens und beschreibt die zu treffenden Ziele. Man wählte diesen Begriff, da Form und Position der Tangibles ebenfalls eine Rolle spielen und es auch mehr als ein solches Objekt geben kann.
Reactable beamt und scannt von unten, wodurch keine Verdeckung durch die Hände entstehen kann. Auch gibt es auf dem Lichttisch keine explizite Cursorposition oder einen Mauszeiger. In der Zukunft sollte man sich also von der Idee der Maus an eindeutiger Position verabschieden und eher an eine gleichzeitige Interaktion durch „Hineinbringen“ und Bewegen von Objekten denken, um das Konzept zu verstehen.
Eine Liste anderer Lichttische gibt es unter: http://mtg.upf.es/reactable/?related.

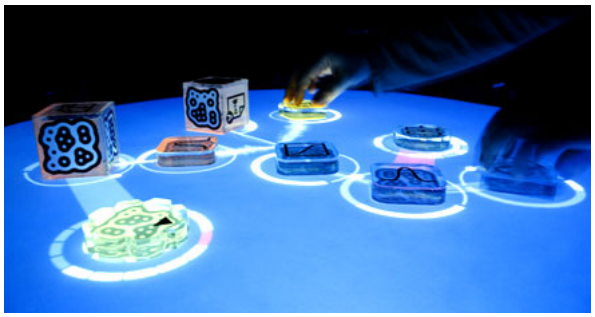


Abbildung 2.1: Reactable

Abbildung 2.2: Microsoft Surface

2.1.2 Microsoft Surface

http://www.microsoft.com/surface/

Microsoft Surface ist ein Computer der Firma Microsoft. Er wurde im Mai 2007 vorgestellt.
Der Surface basiert auf einer neuen Benutzeroberfläche, genannt Multitouch. Es werden Infrarot-Reflexionen menschlicher Fingerspitzen von fünf innerhalb der Box installierten Kameras aufgezeichnet. Das System ist der Funktionsweise eines Touchscreens vergleichbar, ermöglicht aber wesentlich komplexere Eingaben. So ist es beispielsweise möglich, beliebig viele Finger gleichzeitig zu benutzen. Dadurch kann der Nutzer Aktionen mit beiden Händen ausführen oder auch mit anderen Nutzern zusammen auf dem Bildschirm arbeiten. Das Gerät ist in der Lage, insgesamt 52 unterschiedliche Eingaben zu registrieren und zu verarbeiten. Das heißt, es können theoretisch maximal fünf Personen

mit jeweils allen zehn Fingern Objekte steuern bzw. Eingaben machen. Ferner können alle Aktionen, für die bei herkömmlichen PCs eine Maus erforderlich ist, mit einem oder mehreren Fingern gesteuert werden, etwa das Verschieben oder Skalieren von Fenstern.
Der Bildschirm liegt waagerecht auf einem schwarzen Kasten, in dem sich die Hardware befindet. Er ähnelt im Design einem Couchtisch. Dies ermöglicht es mehreren Personen, sich um das Gerät zu platzieren und es gemeinsam zu benutzen.
Eine weitere Neuigkeit des Surface ist die Synchronisation mit anderen Geräten, wie Mobiltelefonen oder Digitalkameras. Verfügen diese über WLAN, so genügt es, sie auf den Bildschirm zu legen. Sie werden vom Surface erkannt und können per Drag & Drop Daten mit dem Computer austauschen. Diese werden einfach mit dem Finger „in das Gerät geschoben“.
Der Surface verwendet Microsofts aktuelles Betriebssystem Vista und verfügt unter anderem über WLAN, Ethernet 10/100 und Bluetooth 2.0.
Der Surface misst 56 cm in der Höhe, 53 cm in der Tiefe und 107 cm in der Breite. Sein Display ist 30“ groß und besteht aus Plexiglas, welches auf einem pulverbeschichteten Stahlrahmen aufliegt. Die Standard-Auflösung beträgt 1024x768 Pixel.

2.1.3 Cubit

http://labs.nortd.com/cubit/

Cubit ist die neue günstige Variante von Microsoft Surface und ist ein OpenSource- Produkt.
Cubit setzt auf Offenheit sowohl auf der Hardware- als auch auf der Software-Seite: Der Quellcode ist ebenso verfügbar wie eine genaue Bauanleitung samt Einkaufsliste. Die Kosten für einen Multitouch-Tisch sollen sich so deutlich reduzieren. Gleichzeitig will Eyebeam Entwickler in aller Welt ins Boot holen, um auf der offenen Plattform neuartige Anwendungen für die berührungsempfindliche Oberflächen zu schaffen. Addie Wagenknecht, Fellow bei Eyebeam, schuf Cubit, um das Thema Multitouch zu entmystifizieren. Sie und ihr Mitstreiter Stefan Hechenberger wollten beweisen, dass jeder Nutzer mit einigen simplen Komponenten ein solches Gerät bauen könne. Neben der Verfügbarmachung der Software im Internet verkauft Wagenknecht auch Do-it-yourself-Kits. Zusammengenommen soll ein persönlicher Multitouch-Tisch so nur noch zwischen 500 und 1000 Dollar kosten, je nachdem, wie leistungsstark die verwendete Hardware ist.
Eine einzige Kamera im Tisch, eine einfache Webcam mit Infrarotfilter reicht aus. Diese wird dabei mit einem kleinen Bildprojektor kombiniert, den es bereits ab 300 Dollar zu kaufen gibt. Der Nutzer muss dann nur noch die Kamera in den Computer einstecken, die Cubit-Software aus dem Internet installieren, den Projektor anschließen und Bilder auf die Oberfläche projizieren. In Wagenknechts Kit steckt außerdem noch eine Tischabdeckung, die über eine Beschichtung verfügt, die es für die Kamera leichter macht, Objekte zu erkennen. Ebenfalls enthalten sind Infrarot-LED-Streifen, die auf die Rückseite der Oberfläche platziert werden.

2.1.4 „180“

http://www.timroth.de/180

Tim Roth beendete sein Studium 2007 an der Züricher University of the Arts mit der Diplomarbeit „180“. Er entwarf einen Multitouch-Tisch, der für Beratungen wie z.B. der Züricher Bank entworfen wurde. Damit wollte er das Problem beheben, dass sich Menschen bei Meetings meist gegenüber sitzen und so Maus und Tastatur schlecht gemeinsam nutzen können.
Innerhalb seines Multitouch-Tisches befinden sich ein Beamer, Spiegel, eine IR-Kamera und Infrarot-LED‘s, die in Acrylglas in regelmäßigen Abständen eingelassen sind. Durch die LED‘s und die IR-Kamera werden die Bewegungen auf der Plexiglasplatte erkannt. Diese Technik wird FTIR – Frustrated Total Internal Reflection – genannt. Berührt man die Oberfläche mit den Fingern, wird das IR-Licht an dieser Stelle gebrochen und verlässt die Platte, die entstehenden IR-Spots werden von einer Kamera erfasst, die mit einem IR-Filter bestückt ist und daher nur Infrarotlicht empfängt. Die empfangenen Informationen werden mit einer Software ausgewertet. Für diese Arbeit kam die Touchless Bibliothek innerhalb von Flash in ActionScript 3.0 zum Einsatz.

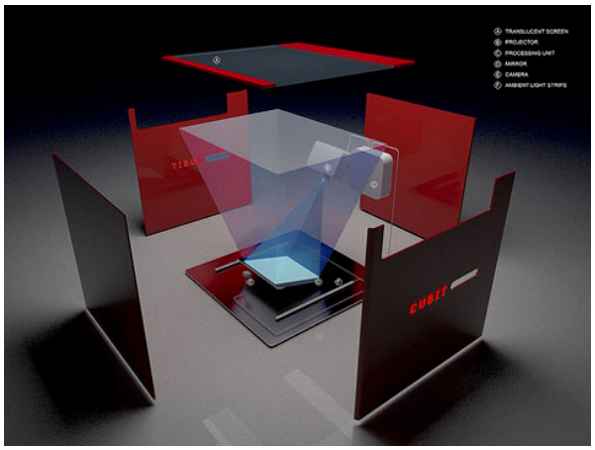


Abbildung 2.3: Cubit – OpenSource-Multitouch-Tisch

Abbildung 2.4: 180

2.2 Mögliche Techniken zur Umsetzung des Projekts

2.2.1 Wii-Controller

http://www.nintendo.de/NOE/de_DE/wii_54.html

Die Wii ist eine fernsehgebundene Videospiel-Konsole der japanischen Firma Nintendo, die seit Ende 2006 auf dem Markt ist. Ihr wesentliches Merkmal ist ein neuartiger Controller, der herkömmlichen Fernbedienungen ähnelt, aber über eingebaute Bewegungssensoren verfügt. Diese registrieren die Position und die Bewegungen des Controllers im Raum und setzen sie in entsprechende Bewegungen von Spielfiguren oder -elementen auf dem Bildschirm um. Mussten die Nutzer bei herkömmlichen Spielsystemen Knöpfe des Controllers oder Analogsticks betätigen, so können sie die Spiele nun steuern, indem sie den Controller selbst bewegen. Das Spielsystem aus Hardware und Software misst dabei die dreidimensionalen Bewegungen des Nutzers. Mithilfe zweier Referenzpunkte in der Sensorleiste, welche unten vor oder oben auf dem Fernseher platziert werden und einer Infrarotkamera an der Vorderseite der Wiimote kann die Position und Lage des Controllers relativ zum Bildschirm bestimmt werden. Dadurch ist es möglich, Spielobjekte auf dem Bildschirm direkt anzuvisieren. Die Präzision ist vergleichbar mit der eines Mauszeigers. Zusätzlich enthält der Controller einen Beschleunigungssensor, mit dem Bewegungen und Drehungen des Controllers erfasst und direkt für die Spielsteuerung genutzt werden können. Außerdem sind verschiedene Knöpfe und ein Steuerkreuz vorhanden. Die Kommunikation mit der Konsole erfolgt kabellos via Bluetooth. Der maximale Abstand beträgt 10 Meter. Ein interner Speicher ermöglicht das Speichern von Benutzerprofilen.
Die Wii-Gestenerkennung kann mit einen Programm der Uni Augsburg bearbeitet werden. Der Wii-Controller lässt sich per Bluetooth mit einem PC verbinden und mit Hilfe der eingebauten Infrarotkamera lassen sich Bewegungen von Infrarotlichtpunkten aufzeichnen und am PC darstellen. Dadurch ist es beispielsweise möglich, mit einem Infrarotstift seinen Computerbildschirm in einen Touchscreen zu verwandeln.

Abbildung 2.5: Wii-Controller

Abbildung 2.6: Anwendung „Map“ des Touchless SDKs

2.2.2 Touchless SDK

http://www.codeplex.com/touchless

Touchless SDK ist ein umfangreiches SDK, das es erlaubt, durch Gesten gesteuerte Anwendungen (Multitouch-Anwendungen) zu erstellen. Dabei können Alltagsobjekte als „Controller“ genutzt werden. Dies wird durch Farbtracking ermöglicht.
Touchless SDK ist ein OpenSource-Programm das von Microsoft-Mitarbeitern entwickelt wurde. Es wurde im Oktober 2008 veröffentlicht und seit dem ständig weiterentwickelt. Die genutzte Programmiersprache ist C#. Touchless benötigt einen Computer mit Windows XP oder Windows Vista sowie eine Webcam die DirectX 9 kompatibel ist.
Die Anwendung kann nach dem Herunterladen und Extrahieren einfach über die Datei Touchless-Demo.exe gestartet werden. Die Demo besteht aus vier verschiedenen Programmen. Das bekannte Spiel Snake, Defender ist eine Pong-Variante, die mit bis zu vier Spielern gespielt werden kann. Map stellt eine Karte dar die man rotieren, zoomen und bewegen kann (siehe Abbildung 2.6). Draw, die letzte Anwendung, ist ein einfaches Malprogramm.
Um Touchless SDK in eigenen Projekten nutzen zu können benötigt man:
- Visual Studio 2005/2008, oder Visual Studio Express Edition
- .NET 3.0 oder höher
- „TouchlessLib.dll“ und „WebcamLib.dll“
- Webcam die DirectX 9 kompatibel ist

2.2.3 CamSpace

http://www.camspace.com/

Bei CamSpace werden Alltagsobjekte als „Controller“ für Computerspiele verwendet. Ermöglicht wird dies über Farbtracking, wofür jede Webcam verwendet werden kann.
Die Webcam verfolgt dabei die Bewegung des Controllers, die von der Software des israelischen Unternehmens CamTrax Technologies (http://www.camtraxtechnologies.com) in Steuersignale übersetzt wird.
Noch ist die Palette der speziell für die CamSpace-Steuerung entwickelten Games und der mit Controller-Emulationen unterstützten Spieletitel klein. Um sie zu erweitern, setzt CamTrax auf die Gamer-Community.
CamSpace wurde im Juni 2008 das erste Mal veröffentlicht. Es ist kostenlos und wird ständig weiterentwickelt. Derzeit ist es aber nur für Windows erhältlich. Die genützte Programmiersprache ist LUA (Skriptsprache).
Um CamSpace zu nutzen benötigt man:
- Windows XP/Vista
- Webcam
- gut beleuchtete Arbeitsumgebung

2.2.4 Processing

http://www.processing.org

Die von Wiring abgeleitet, C-ähnliche Sprache Processing ist eine auf die Einsatzbereiche Grafik, Simulation und Animation spezialisierte objektorientierte, stark typisierte Programmiersprache mit zugehöriger integrierter Entwicklungsumgebung. Sie wird in einem quelloffenen Projekt entwickelt, das am Massachusetts Institute of Technology in Boston initiiert wurde. Processing hat den Charakter einer stark vereinfachten Version der Programmiersprache Java. Diese erlaubt es Interaktionen und visuelle Elemente zu programmieren und richtet sich vorwiegend an Gestalter, Künstler und Programmieranfänger.
Die Klassenbibliotheken der Programmiersprache zielen vor allem auf das Einsatzgebiet von Processing und berücksichtigen die Themen Video, Grafik, Grafikformate, Sound, Animation, Typographie, 3D, Simulation, Datenzugriff und -transfer, sowie Netzwerkprotokolle. Des Weiteren gibt es für Processing zahlreiche weitere Bibliotheken auf die man für die Programmentwicklung zugreifen kann. Unter anderem steht eine spezielle Arduino-Bibliothek zur Verfügung.

2.2.5 Arduino

http://www.arduino.cc/

Die Arduino-Plattform ist eine aus Soft- und Hardware bestehende Physical Computing-Plattform. Beide Komponenten sind im Sinne von OpenSource quelloffen. Die Hardware besteht aus einem einfachen I/O-Board mit einem Mikrocontroller und analogen sowie digitalen Ein- und Ausgängen. Die Entwicklungsumgebung ist eine plattformunabhängige Java-Anwendung, die auf auf Processing und Wiring (Java-Dialekten) beruht. Die Arduino-IDE soll insbesondere Künstlern, Designern, Hobbyisten und anderen Interessierten den Zugang zur Programmierung und zu Mikrocontrollern erleichtern.
Sie dient als Code-Editor und -Compiler und ist auch in der Lage, die Programme (Sketches) auf die Hardware (siehe Abbildung 2.7) zu übertragen.
Ein Arduino kann benutzt werden, um eigenständige interaktive Objekte zu steuern oder um mit Softwareanwendungen auf Computern zu interagieren (z. B. Adobe Flash, Processing). Die Hardware basiert auf einem Atmel AVR-Mikrocontroller, der über einen FTDI-USB-Seriell-Konverter mit dem PC verbunden werden kann. Alle Boards werden entweder über USB oder eine externe Stromquelle mit 5-Volt versorgt und verfügen über einen 16 MHz Quarzoszillator. Der Mikrocontroller ist mit einem Boot-Loader vorprogrammiert. Die Programmierung erfolgt direkt über die USB-Schnittstelle, ein externer Programmierer ist nicht erforderlich.
Die Arduino-Boards stellen die meisten I/O-Pins des Mikrocontrollers zur Nutzung für oder durch andere elektronische Schaltungen zur Verfügung. Die aktuell gängigen Boards bieten 14 digitale I/O-Pins, von denen 6 PWM-Signale ausgeben können und zusätzlich 6 analoge Eingänge.

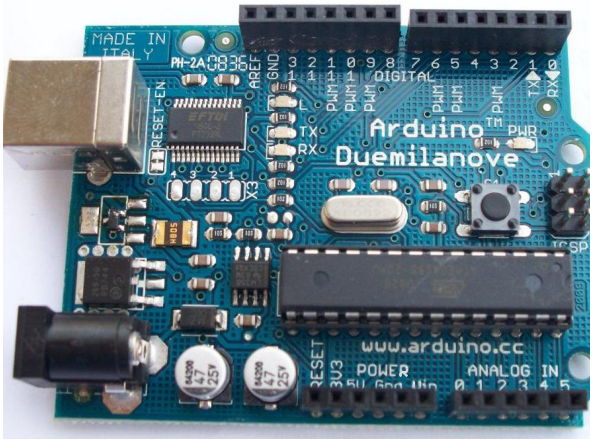


Abbildung 2.7: Arduino

2.3 Tischbau

Für die Konstruktion unseres Tischs haben wir uns unter Anderem die Bauanleitungen von „180“ in der Zeitschrift Page Januar/08 sowie die Anleitungen von Cubit angesehen.

2.3.1 „180“

Beschreibung für den Aufbau eines Tischs für die FTIR-Technik:
Ausschlaggebend für den technischen Aufbau ist der Mindestabstand vom Projektor zur Projektionsfläche, der durch die Projektionsgröße und die Spezifikationen des Projektors bestimmt wird. Außerdem muss man mit den IR-Strahlern die komplette Projektionsfläche gleichmäßig beleuchten. Grundsätzlich sollte man nicht zu kompakt bauen, damit genügend Spielraum für Anpassungen bleibt.
Für die Konstruktion der Kiste kann man einfache Sperrholzplatten nutzen. Die Öffnung für den Projektor an einer der vier Seitenwände wird mit einer Stichsäge heraus gesägt. Die Seitenwände werden mit Winkeln montiert und die Tür mit Scharnieren befestigt. Anschließend wird passend für die verwendete Acrylplatte ein Loch herausgesägt.

Erstellung einer flexiblen Spiegelhalterung:
Da sowohl das Bild des Projektors als auch das Tracking der Kamera über den Spiegel verläuft, muss man seine Position nachträglich verändern können. Dafür wird eine einfache Winkelkonstruktion erstellt, die dem Spiegel halt bietet, es aber auch erlaubt ihn zu kippen um den Projektions- und Blickwinkel zu verändern.

Projektorhaltung montieren:
Ebenfalls muss man den Projektor kippen können, um einen möglichst guten Projektions- und Blickwinkel zu erreichen. Die Konstruktion besteht aus vier rechteckigen Platten, von denen drei mit Scharnieren verbunden sind. Die letzte Platte wird mittels Winkeln an der kürzesten fixiert.

Die weitere Bauanleitung befasst sich mit der Möglichkeit Multitouch über IR-Strahler und IR-Filter zu realisieren, welche von uns nicht verwendet wird, da sie keine Differenzierung zwischen verschiedenfarbigen Spielsteinen ermöglicht.

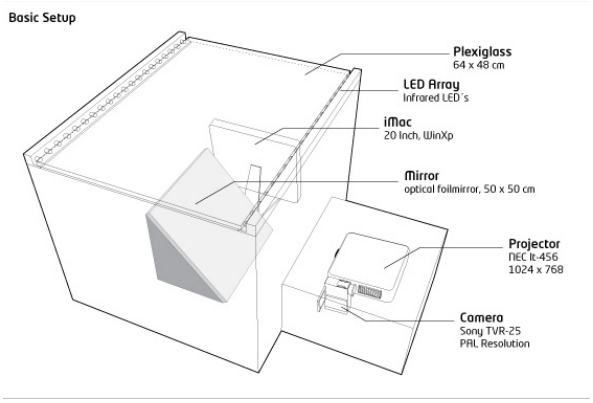


Abbildung 2.8: Aufbau von Projekt „180“

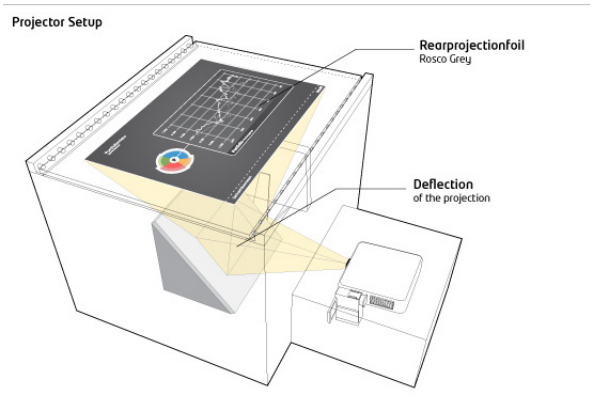


Abbildung 2.9: Aufbau mit Beamerreflektion

2.3.2 Cubit

Für den Cubit gibt es sehr viele unterschiedliche Beschreibungen, die man sich im Forum („http://forum.nortd.com/“) und unter „http://labs.nortd.com/touchkit/“ ansehen kann (siehe Abbildung 2.3). Der Cubit wird von seiner Community mit mehreren neuen Darstellungsmöglichkeiten ständig weiterentwickelt. Die Bauweise entspricht im Wesentlichen dem Projekt „180“.

3. anforderungen

3.1 Spiel

Maßgeblich für einen Erfolg des Spiels ist der Spaßfaktor für den Spieler. runKlotzen soll spannend, kurzweilig und gesellig sein. Damit einher geht die Benutzerfreundlichkeit, sprich Spielregeln und Gestaltung sollen einfach und schnell verständlich sein. Das Spiel sollte auch nach mehreren Stunden in Betrieb fehlerfrei laufen. Grundvoraussetzung ist die nahtlose Kombination von digitaler und realer Welt, die im Spielkonzept vollends ausgereizt werden soll. Dabei kommt vor allem dem visuellen und akustischen Feedback eine große Rolle zu. Ebenfalls konsequent umzusetzen ist die Integration von Buzzern (Not-Aus-Taster). Auf dem Brettspielemarkt stellen sie ein vielversprechendes Alleinstellungsmerkmal (USP: Unique Selling Proposition) dar.

3.2 Interaktiver Tisch

Ein wichtiger Fokus des Projekts liegt auf der Herstellung des interaktiven Tisches, der eigens für das Spiel runKlotzen entworfen wird, um den Anforderungen vollends zu entsprechen. In diesem Fall wird eine Rückprojektion angestrebt, da anderenfalls bei einer Projektion von oben die Hände der Spieler die Darstellung des Spielfelds stören. Deshalb ist die Größe des Tischs entsprechend zu wählen, um Platz für Beamer, Spiegel, Kamera und Audioanlage zu finden. Aus Gründen der Abgeschlossenheit und Transportfähigkeit wird der benötigte Rechner und Beamer für das Spielgeschehen ebenfalls im Tisch untergebracht. Hinsichtlich eines quadratischen Spielfelds, erhält auch der Tisch eine quadratische Form. Außerdem sollte der Tisch auch etwas unsanfterer Behandlung widerstehen können.

3.3 Anwendungsbereiche

runKlotzen wird von Einzelpersonen aus der Zielgruppe mindestens zu zweit, höchstens zu viert, gespielt – zur Unterhaltung und zum Zeitvertreib. Das Projekt kann grob in die Kategorie Gesellschaftsspiel eingeordnet werden. Obwohl es ein Brettspiel ist, wird es hinsichtlich der Umgebung eher mit Billard oder Kicker auf eine Ebene gestellt. runKlotzen wird am interaktiven Tisch im Stehen gespielt und könnte somit auch in einer Bar zum Einsatz kommen.

3.4 Zielgruppe

Prinzipiell ist das Spielkonzept ab einem Alter von acht Jahren geeignet. Durch die physikalische Größe des Tisches von einem Meter wird das Alter allerdings hinaufgesetzt. Der Spieler muss eine gewisse Körpergröße haben, um am Spiel teilnehmen zu können. Daraus ergibt sich eine Zielgruppe von Personen zwischen 12 und 99 Jahren. runKlotzen konzentriert sich nicht speziell auf eine Bevölkerungsschicht, Altersgruppe oder Geschlecht. Jeder, der gerne Brettspiele in Gesellschaft spielt oder Angebote wie Dart, Kicker oder Billard in einer Bar nutzt, ist auch ein potentieller Anwender.

4.1 Spiel

Die Projektion in der Mitte des Tisches besteht aus 15 x 15 Quadraten, die als Spielfelder zusammen das Spielbrett ergeben. Zwischen vier Arten von Spielfeldern wird unterschieden: inaktiv (grau), aktiv (weiß), Aktionsfeld (quadratischer Rand) und Zielfeld (kreisförmiger Rand). Inaktive Felder sind nicht begehbar. Der mittlere Bereich (drei auf drei Felder) dient der Information für den Spieler. Hier werden die Zeit (Ring) und weitere grafische Elemente wie der Würfel oder Icons dargestellt (siehe Abbildung 4.1). Alle grafischen Elemente müssen von allen vier Seiten gleich gut zu erkennen sein. Die Spielfiguren bestehen aus Holzklötzchen in Form von Zylindern und sind in den vier Farben Rot, Grün, Gelb und Blau gestrichen. Jeder Spieler startet mit sieben Stück. An jeder Seite des Tisches befindet sich eine Homebase aus Plexiglas in der jeweiligen Farbe der Spielfiguren. Eine LED darunter leuchtet auf, wenn der Spieler am Zug ist. Die Interaktion wird durch einen eigenen Buzzer neben der Homebase ergänzt (siehe Abbildung 4.2). Eine Animation der Spielregeln dient als „Bildschirmschoner“, wenn nicht gespielt wird. Einerseits soll sie erklären, aber andererseits auch zum Spielen anregen.

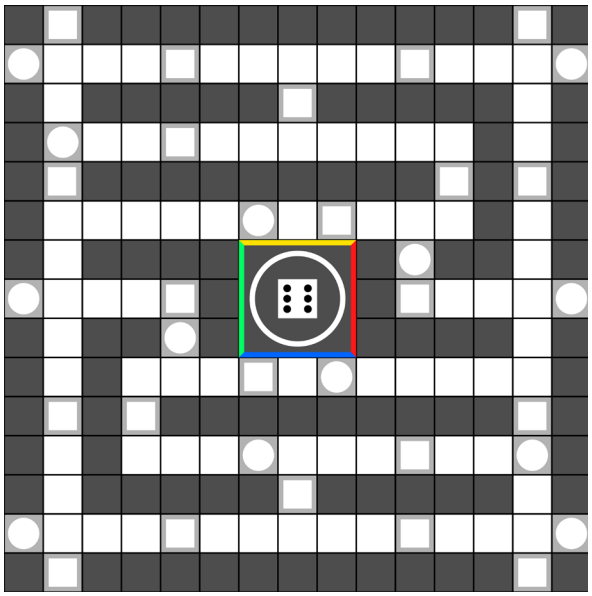


Abbildung 4.1: Spielfeld

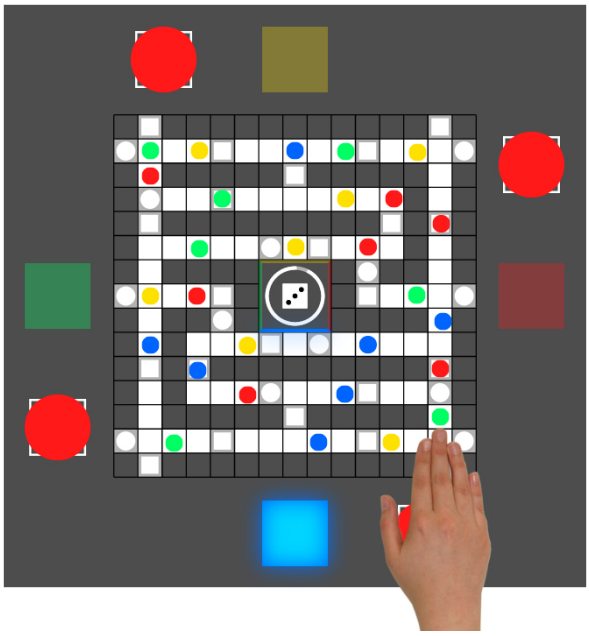


Abbildung 4.2: Interaktiver Tisch

4.2 Spielregeln

Das Ziel des Spiels ist es möglichst viele Spielfiguren auf die Zielfelder zu bewegen, bevor die Zeit abgelaufen ist oder alle Felder besetzt sind. Insgesamt werden vier Ebenen durchspielt, in denen sich die Anzahl der Zielfelder jeweils verringert – 13, 8, 4 und 1. Gewinner ist der Spieler, der zuerst seine Spielfigur in der letzten Ebene auf das Zielfeld bewegt. Zu Beginn stellen alle Spieler ihre Figuren auf die Homebase. Das System leitet das Spiel ein und ermittelt durch Setzen eines Klötzchens, ob die Farbe mitspielt. Nachdem alle Spieler ihre sieben Figuren gesetzt haben, beginnt das eigentliche Spiel. Durch Würfeln wird ein Spielzug begonnen, indem der Spieler den digitalen Würfel in der Mitte mit seinem Buzzer stoppt. Der Spieler zieht mit einer Figur seiner Wahl um die Anzahl der gewürfelten Augen. Auf einem Zielfeld ist die Spielfigur vorerst in Sicherheit, auf einem weißen Feld passiert nichts weiter – der nächste Spieler ist an der Reihe. Erreicht das Klötzchen aber ein Aktionsfeld, buzzert der Spieler um das gewünschte Ereignis. Wiederum in der Mitte des Spielfelds wechseln die Aktionen in Form von Icons durch und werden durch Betätigen des Buzzers angehalten. Zur Auswahl stehen:

Einfrieren: Spielfigur eines Gegner für eine Runde außer Gefecht setzen. Dazu wird die gewünschte Figur auf den Kopf gestellt.

Wettbuzzern: Alle Mitspieler buzzern um Geschwindigkeit. Verschiedene Farben werden per Zufall in der Spielfeldmitte angezeigt. Bei der Farbe Weiß muss der Taster gedrückt werden. Der Schnellste darf daraufhin eine Spielfigur seiner Wahl direkt auf ein Zielfeld bewegen.

Farbwechsel: Alle Spieler rotieren im oder gegen den Uhrzeigersinn um den Tisch und wechseln somit Farbe und Spielfiguren. Die Richtung bestimmt der Spieler, der an der Reihe ist.

Befinden sich bereits alle Figuren des Spieler auf Zielfeldern, das Spiel ist aber noch im Gange, besteht Zugzwang. Er muss also eine Figur wieder heraus setzen und um die gewürfelten Augen ziehen. Ist der Spielzug beendet, drückt der Spieler auf den Buzzer und der nächste ist an der Reihe. Die Zeit pro Ebene läuft nach 20 Minuten ab, das Spiel schaltet in die nächste Ebene. Dies kann bereits vorher der Fall sein, wenn alle Zielfelder belegt wurden.

5.1 Hardware

5.1.1 Tischbau

Ähnlich dem Aufbau anderer interaktiver Tische (vgl. Kapitel 2.3) kommt auch bei diesem Projekt eine Holzkiste als Tisch zum Einsatz. In dieser können sowohl der benötigte Beamer und ein Spiegel für die Spielfeldprojektion als auch eine Kamera, eine Soundanlage und ein PC untergebracht werden. Der erste Schritt beim Entwurf der Kiste war, eine passende Höhe und Breite zu finden. Wie im Kapitel Konzeption bereits beschrieben ist runKlotzen ein Spiel, das im Stehen gespielt wird. Daher musste eine Höhe gefunden werden, die ein angenehmes Stehen, Anlehnen und Spielen ermöglicht. Des Weiteren war darauf zu achten, dass die Spielfläche inklusive Rand so dimensioniert ist, dass auch kleinere Spieler noch problemlos das gegenüberliegende Ende der Spielfläche erreichen können. Vergleiche mit Küchenarbeitsflächen und einige Versuche ergaben schließlich eine Höhe und eine Breite von je 100 cm. Beeinflusst durch die Projektionsfläche des Beamers und der Spielfeldkonzeption ergab sich für die zentriert angebrachte Plexiglasscheibe eine Größe von 54,5 cm x 54,5 cm. Die Abbildungen 5.1 und 5.2 zeigen erste Entwürfe für den Tischaufbau.

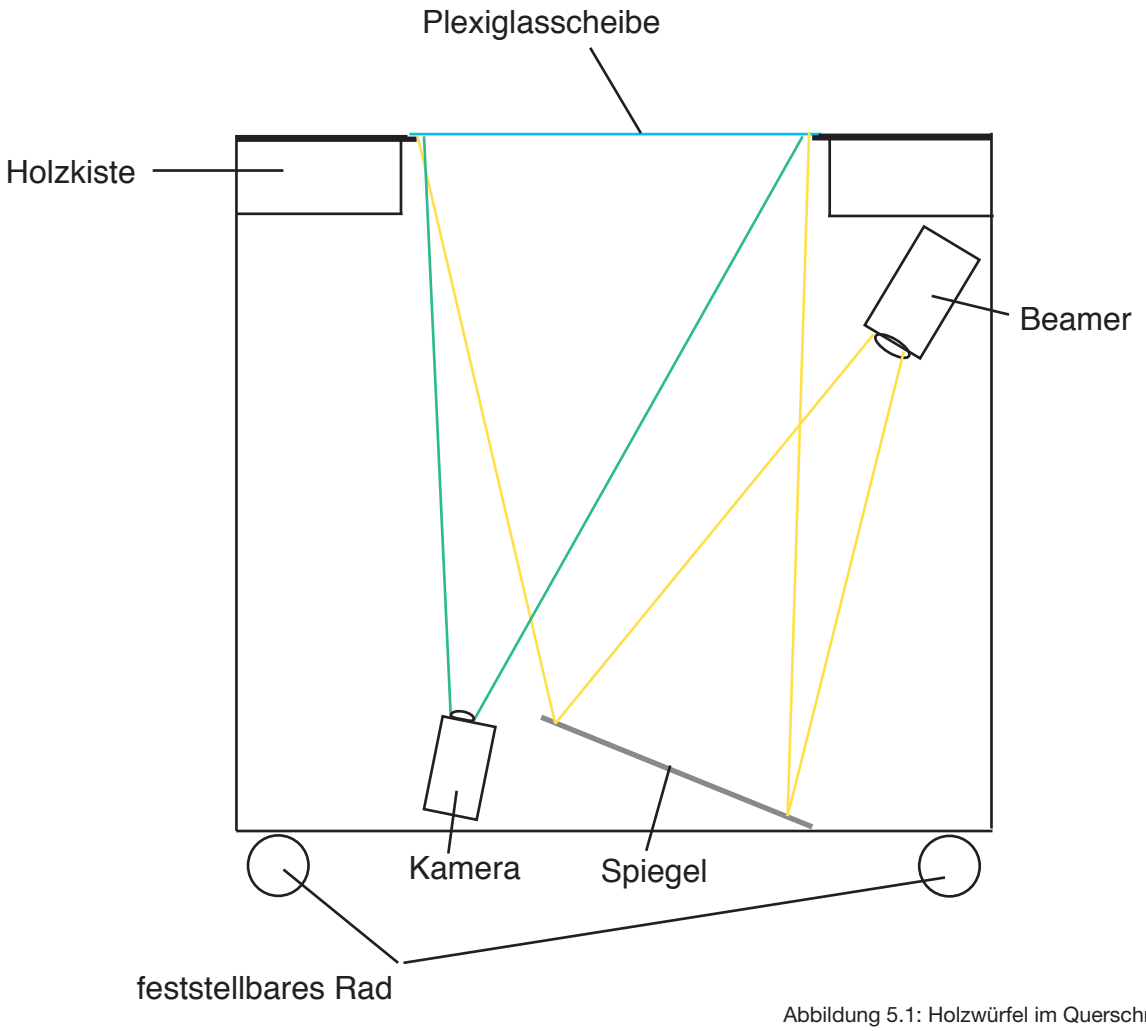


Abbildung 5.1: Holzwürfel im Querschnitt

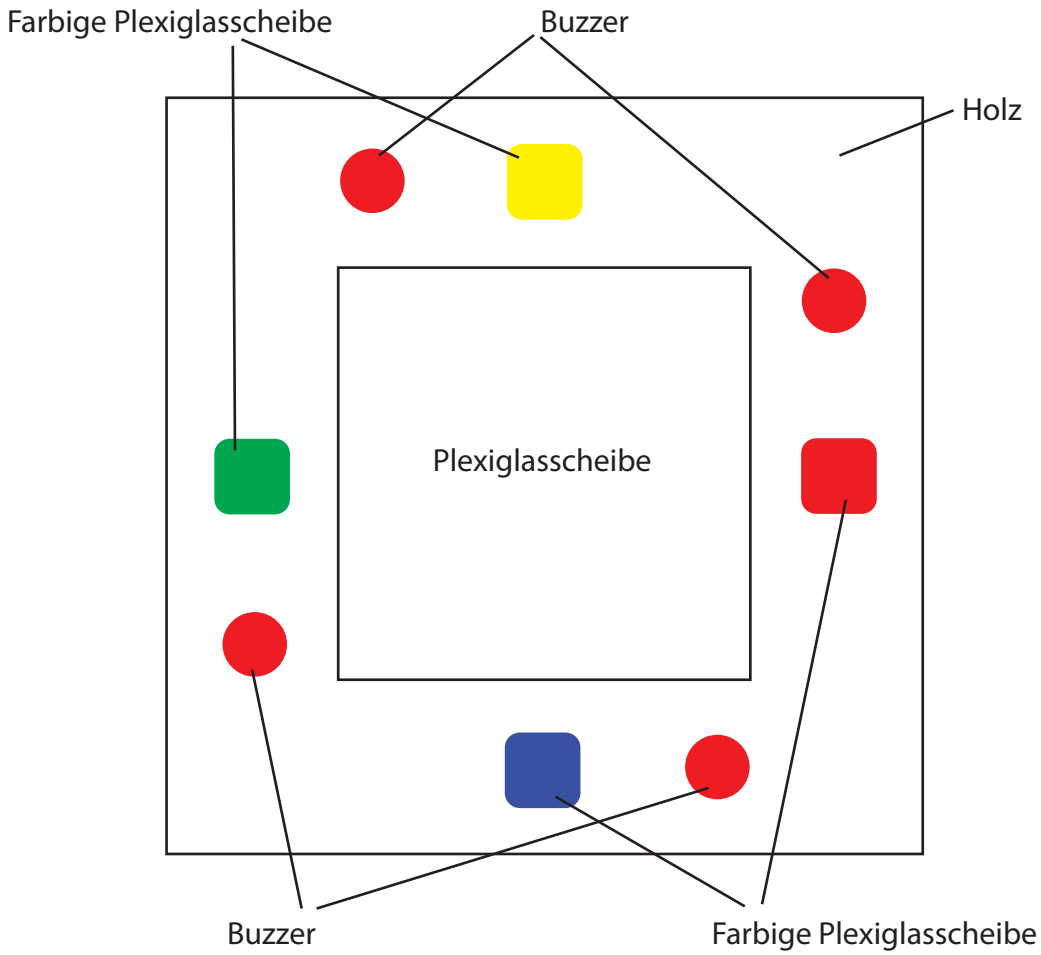


Abbildung 5.2: Tischansicht von Oben

Die Planung sah vor, dass der Holzquader auf vier feststellbaren Rollen stehen sollte. Spätere Überlegungen ließen uns aber zum Entschluss kommen, den Würfel auf zwei feststellbar Rollen und zwei Holzfüße zu stellen. So kann der Würfel einerseits leicht verschoben werden, weißt aber andererseits eine höhere Stabilität und Standfestigkeit auf.
Das Spiel ist auf bis zu vier Spielern ausgelegt. Auf jeder Würfelseite befindet sich ein Buzzer und eine kleine farbige Plexiglasscheibe, die sogenannte „Homebase“, in blau, grün, gelb bzw. rot. Diese zeigt die dem Spieler zugeordnete Farbe an. Ist ein Spieler am Zug, wird dessen „Homebase“ von unten mit einem LED-Spot beleuchtet. Die unter jedem der Buzzer und „Homebase“ gelegene Holzkiste dient sowohl als Befestigungsmöglichkeit für den Buzzer und die Lampe als auch als Abschirmung der farbigen Plexiglasscheibe vor dem Licht des Beamers (vgl. Abb. 5.1).

Um eine klarere Vorstellung vom Tisch zu erhalten, wurde ein Modell im Verhältnis 1:10 angefertigt. Dieses ist in Abbildung 5.3 und 5.4 dargestellt.

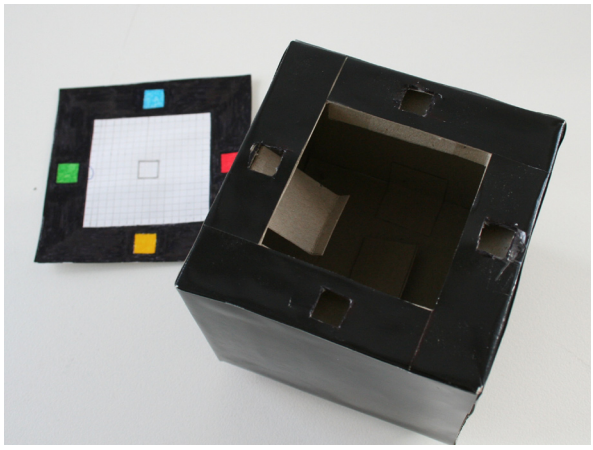


Abbildung 5.3: Innenansicht des Würfelmodells

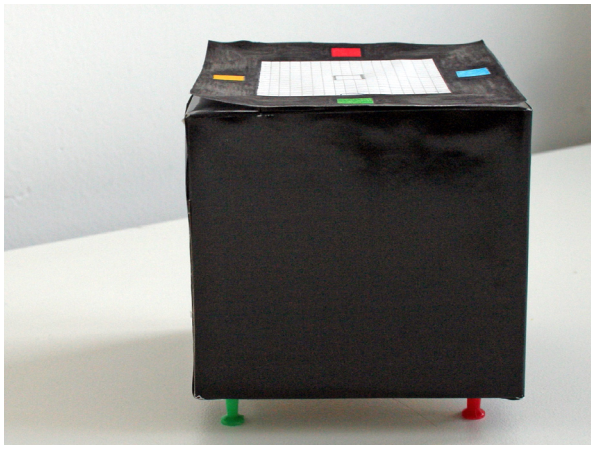


Abbildung 5.4: Seitenansicht

Parallel zur Entwurfsphase des Tischs wurden einige Tests bezüglich der Beschaffenheit der Plexiglasscheibe für die Projektion des Spielfelds durchgeführt. Nachdem eine komplett getönte Milchglasscheibe wegen zu großen Streuungseffekten nicht mehr in Frage kam, experimentierten wir mit verschieden stark lichtdurchlässigen Milchglasfolien auf einer klaren Plexiglasplatte (siehe Abbildung 5.5). Die Wahl fiel schließlich auf eine schwach milchige, statisch haftende Folie von d-c-fix. Diese Folie lässt zum einen genug Licht von unten durch, sodass die Darstellung der Spielfläche gut für die Spieler erkennbar ist. Zum anderen lassen sich aber auch farbige Gegenstände, die sich auf der Spielfläche befinden, noch gut von unten erkennen.

Abbildung 5.5: Plexiglasscheibentest

In Zusammenarbeit mit der Schreinerei der Hochschule Augsburg entstand der ein Kubikmeter große Holzkubus (siehe Abbildung 5.6). In weiteren Arbeitsschritten wurde dieser mit grauem, stoßfesten Acryllack gestrichen, die Boxen in die Seitenwände eingebaut und mit Silikon versiegelt, die Plexiglasscheiben eingepasst und die Buzzer sowie die LED-Lampen eingebaut und verkabelt. Das Ergebnis ist in Abbildung 5.7 zu sehen.

Abbildung 5.6: Tischbau in der Schreinerei

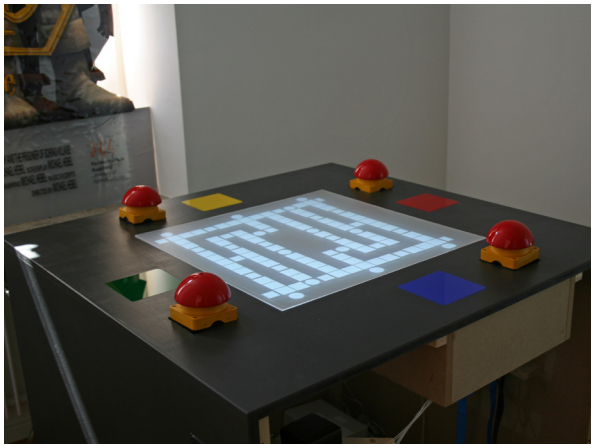


Abbildung 5.7: Gestrichener Tisch mit Buzzern

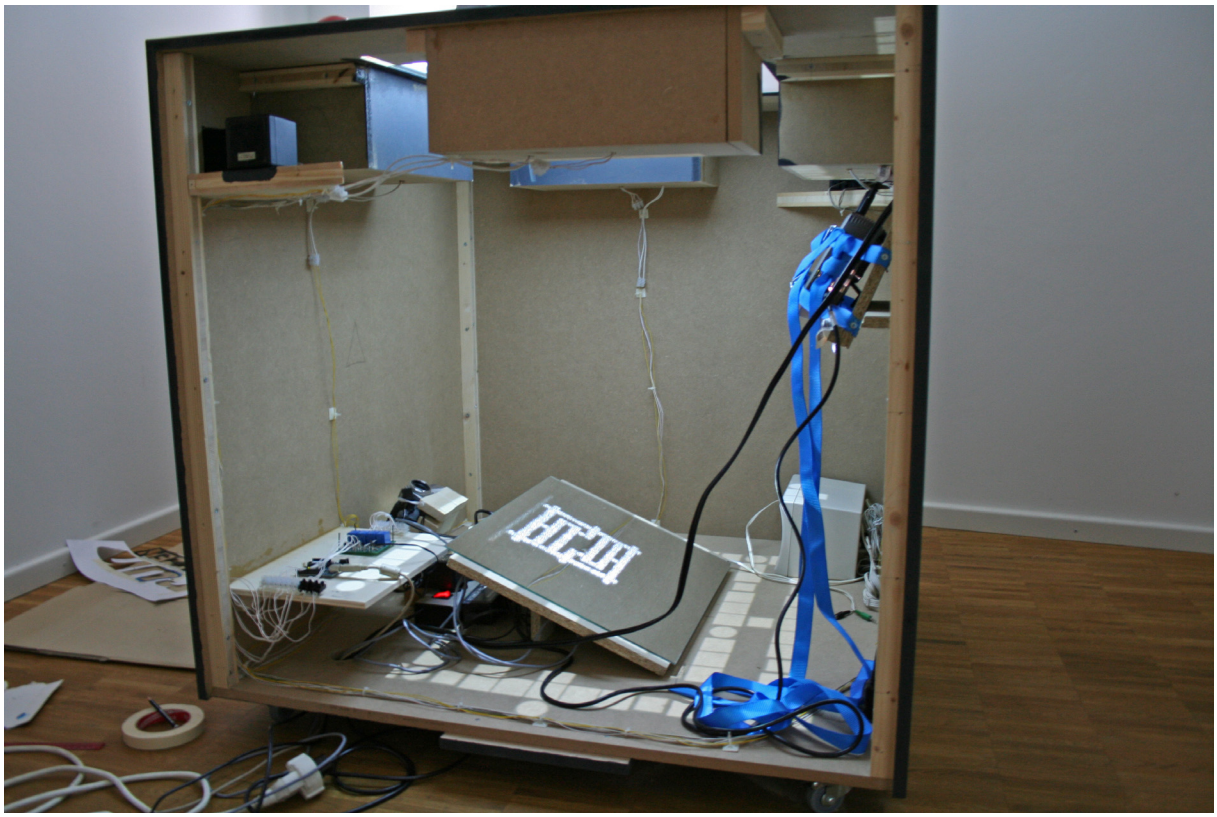


Abbildung 5.8: Innenausbau des Tischs

Am Ende fanden noch einige Ausbauten im Inneren des Würfels statt. Für die Kamera und den Beamer wurden Halterungen angefertigt und ein winkelverstellbarer Spiegel eingebaut. Die Schaltungseinheit (vgl. Kapitel 5.1.3) wurde auf einem Brett befestigt und an die verlegten Kabel zu den Buzzern und Lichtern angeschlossen (Abbildung 5.8).

Während den durchgeführten Testläufen stellte sich heraus, dass die Wärmeentwicklung durch den Beamer und den PC im Würfel sehr hoch ist. Daher wurden nachträglich noch einige Belüftungslöcher direkt hinter dem Beamer in den Würfel gebohrt. Außerdem sorgt ein Ausschnitt im Boden des Quaders für die nötige Frischluftzufuhr von unten.
Final wurden auf den vier Würfelseiten des Spieltischs noch das runKlotzen-Logo in der jeweiligen Spielerfarbe (blau, grün, gelb und rot) aufgemalt.

5.1.2 Spielfiguren

Die Spielfiguren des Spiels runKlotzen bestehen aus 4,5 cm hohen und im Durchmesser 2,5 cm breiten Holzzylindern. Diese wurden in den vier Farben blau, grün, gelb und rot mit Holzlack gestrichen. Das obere Ende der Zylinder wurde mit der grauen Würfelfarbe versehen.
Die Ortsbestimmung der Figuren auf dem Spielfeld geschieht über Farberkennung (vgl. Abschnitt 5.2.2). Zu Beginn des Projekts war vorgesehen, die Unterseite der Spielsteine in deren jeweiliger Farbe einzufärben und diese damit einer bestimmten Farbe zuordenbar zu gestalten. Tests mit einer Kamera und einer milchigen Plexiglasscheibe aber zeigten, dass die Farbqualität am Spielsteinboden durch die Milchglasoptik fast verloren geht und von der Kamera nicht mehr erkannt werden kann. Viele Tests mit reflektierenden Farbfolien, unterschiedlich hellen Farben und Leuchtfarben ergaben, dass nur Leuchtgelb und Leuchtorange zuverlässig von der Kamera erkannt und unterschieden werden kann. Aus diesem Grund haben wir uns entschlossen diese beiden Farben einzusetzen und ein zusätzliches Muster, wie beispielsweise einen schwarzen Kreis in der Mitte des Zylinderbodens, auf zwei der vier Spielerfarben einzufügen. Durch diese Technik können wieder vier Spieler voneinander unterschieden werden:

1. über die Farbe Leuchtgelb und Leuchtorange
2. befindet sich auf der jeweiligen Farbe ein Kreis oder nicht.

Damit ergibt sich als Farbcodierung für die Spielsteine, dass die gelben Figuren einen leuchtgelben Boden erhalten, die roten Figuren einen leuchtorangenen. Die Spielfiguren mit der Farbe blau werden neben der leuchtgelben Farbe mit einem zentrierten schwarzen Kreis versehen, ebenso hält es sich mit den grünen Spielsteinen, die einen leuchtorangefarbenen Boden mit einem zentrierten schwarzen Kreis erhalten. Die Abbildungen 5.9 und 5.10 zeigen die fertig bemalten Spielfiguren.

Abbildung 5.9: Bemalung der Spielsteine

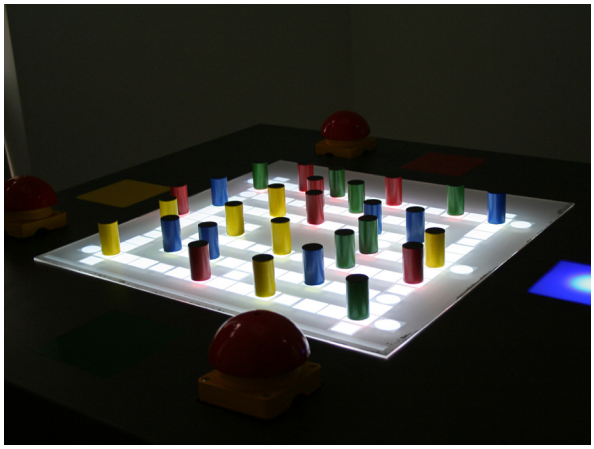


Abbildung 5.10: Spielfiguren auf dem Spielfeld

5.1.3 Schaltung

Die Ansteuerung der Lichter unter den „Homebases“ und die Verarbeitung der Signale, die von den Buzzern gesendet werden, wird von einem Arduino-Board übernommen. Dieses empfängt von der Spielelogik welche „Homebase“ gerade beleuchtet werden soll und sendet an die Spielelogik welche Buzzer gedrückt wurden. Der folgende Plan (Abbildung 5.11) zeigt die bestehende Schaltung.

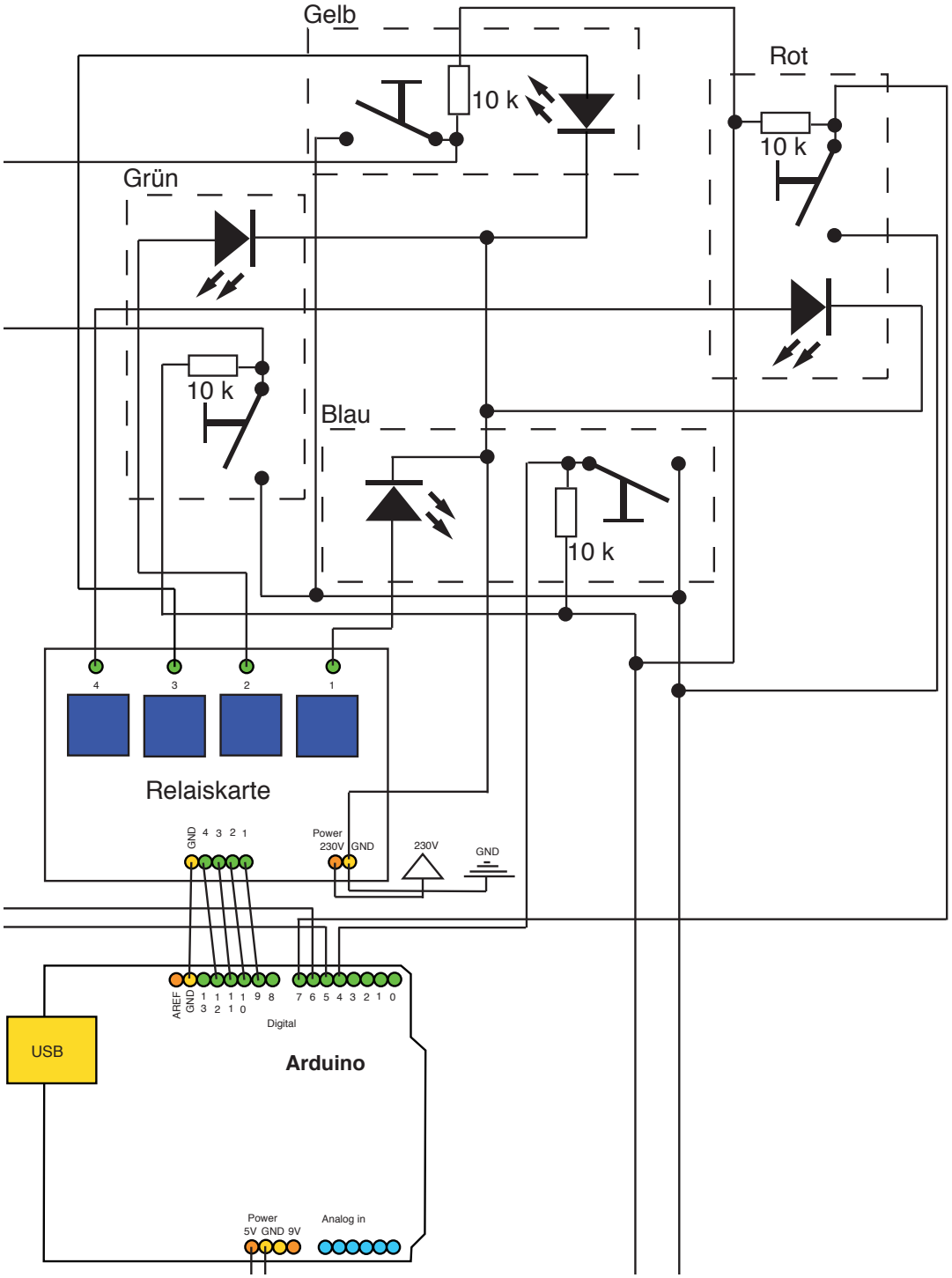


Abbildung 5.11: Schaltplan der Arduino-Steuerung

Da die Spannung des Arduino von 5V für die LED-Lampen nicht ausreicht kommt eine zusätzliche Relaiskarte zum Einsatz. Dieser Baustein hat eine eigene Energieversorgung von 230V und versorgt die Lichter mit Strom. Das Arduino-Board übernimmt lediglich die Schaltung der Lichter. Dazu sind die vier Eingänge der Relais-Karte mit vier Ports des Arduino verbunden. Folgende digitale Ports werden als Ausgang verwendet: Port 9 zur Steuerung der Lampe der blauen „Homebase“, Port 10 für die grüne „Homebase“, Port 11 steuert die Lampe der gelben „Homebase“ und Port 12 ist für die Ansteuerung der roten „Homebase“-Lampe zuständig. Die Lampen sind zu Spielbeginn alle ausgeschaltet und zeigen während des Spiels die Farbe an, die am Zug ist. Als Information, welche Lampe brennen soll, erhält das Arduino-Board von der Spielelogik einen Integer-Wert. Dieser enthält in vier Bit kodiert die Lampenschaltung. Das niederwertigste Bit enthält dabei den Wert für das blaue Licht, das zweite Bit für das grüne usw. (siehe Abbildung 5.12). Dem zufolge würde die Spielelogik beispielsweise den Integer-Wert vier schicken, um das Licht der gelben „Homebase“ anzuschalten.

Abbildung 5.12: Bitcodierung der Lampen

Diese Information kann das Arduino über einfache Bit-Operationen verarbeiten. Der folgende Code zeigt, wie die übertragene Information der Spielelogik (data) durch das Arduino verarbeitet wird. Mit dem Befehl digitalWrite wird auf den Port der gelben „Homebase“ (ledYellow) der angegebene Wert über eine Bitmaske (0100 = 4, data & 4) und eine darauf folgende Bit-Shift-Operation (>> 2) ermittelt und übertragen.

```
digitalWrite(ledYellow, (data & 4) >> 2);
```

Ebenfalls an das Board angeschlossen sind die vier im Spiel integrierten Buzzer, die über, als Ausgang geschaltete Ports angesteuert werden.
Port 4 = Buzzer blau
Port 5 = Buzzer grün
Port 6 = Buzzer gelb
Port 7 = Buzzer rot

Da die als Buzzer umgebaute Grobhandtaster (vgl. Kapitel 5.1.4) beim Drücken oftmals ein längeres Signal als benötigt senden, wird zu einem kleinen Trick gegriffen. Eine zusätzliche Boolean-Variable, als Beispiel hier buzzerYellowPressed, hält fest, ob ein „Buzzer gedrück“-Signal (=1) schon gesendet wurde oder nicht. Ist dies der Fall, wird die Buzzer-Variable (hier valueYellow) auf 0 gesetzt, um ein erneutes senden dieser Information an die Spielelogik zu vermeiden.

```
valueYellow = digitalRead(buzzerYellow);
  if(valueYellow != buzzerYellowPressed)
    buzzerYellowPressed = valueYellow;
  else
    valueYellow = 0;
```

Das Arduino-Board sendet nur Informationen an die Spielelogik, wenn sich diese von 0 unterscheiden, d.h. wenn mindestens ein Buzzer gedrückt wurde. Die Information, welcher Buzzer gedrückt wurde wird wie zuvor bei den Lampen beschrieben in einem Integer-Wert als vier Bit Wert kodiert. Die Reihenfolge entspricht dabei der Farbkodierung der Lampen (siehe Abbildung 5.12).

```
//Information kodieren
int value = valueRed << 3 | valueYellow << 2 | valueGreen << 1 | valueBlue;

if(value != 0)    //mindestens ein Buzzer wurde gedrückt
    Serial.write(value);
```

Der vollständige Arduino-Code befindet sich im Anhang 1.

5.1.4 Buzzer

Die ins Spielkonzept integrierten Buzzer sind eigentlich Grobhandtaster (Not-Aus-Taster) der Firma Moeller. Dennoch sind diese besonderen Schalter ein fester Bestandteil diverser Quiz- und Rateshows aus dem TV und lenken die Aufmerksamkeit der Zuschauer oder Spieler auch optisch auf sich. Nachdem wir die Schalter bekommen hatten, stellten wir fest, dass diese einrasten und nicht wie erwartet zurückspringt. Diese Einrastmechanismus macht an Maschinen durchaus Sinn, wir wollten sie jedoch zweckentfremdet einsetzten und als additive Elemente in das Gesamtkonzept des Spieles runKlotzen integrieren. Die Einrastfunktion der Buzzer zu umgehen gestaltete sich etwas schwieriger als erwartet.
Nach einigen eigenen Versuchen fragten wir einen Elektrotechniker der Hochschule um Rat. Dieser hatte auch noch nie zuvor eine solche Aufgabe übernommen, nahm diese aber aus eigenem Interesse gern an und konnte durch einigen Einfallsreichtum und selbst gebaute Federn helfen, die Schalter zu funktionstüchtigen Buzzern umzubauen.
Für den Umbau mussten die Buzzer geöffnet werden und ein Teil des Einrastmechanismus wurde entfernt. Die ursprünglich eingebauten Federn verfügten nicht über genug Zug, um den Schalter zurückspringen zu lassen. Gerade dieses Tastgefühl ist jedoch wichtig, da ein Spieler das Gefühl eines leichten Widerstandes beim Benutzen des Buzzers braucht, um die Funktionalität der Schalter auch direkt spüren zu können. Aus diesem Grund experimentierten wir mit der richtigen Stärke der Federn (selbstgewickelter stärkerer Draht). Anschließend wurde der Widerstand und das Tastgefühl der Buzzer auch auf alle vier untereinander abgestimmt.
Die Buzzer wurden auf alle vier Seiten in die davor vorgesehen ausgesägten Löcher in den Spieltisch eingebaut und die Verkabelung durch vorher gebohrte Löcher vorgenommen. Um den Schlag der Spieler beim Benutzen der Buzzer abzudämpfen wurde jeder der Schalter mit Schaumstoff und einer Lage einer harten Gummimatte auf ihrer Unterseite versehen und im Anschluss in den Tisch ver-

schraubt. Die Buzzer verfügen nun über genug Flexibilität den Aufschlag der Spieler abzufangen. Die Buzzer fügen sich nun harmonisch in das Designkonzept des Spieltisches ein und stellen, in Kombination mit einem interaktiven Brettspiel, ein Alleinstellungsmerkmal auf dem Spielemarkt dar.

5.1.5 Kamera

Die ersten Versuche das vom Beamer projizierte Bild einzufangen, wurden mit einer Webcam durchgeführt. Der automatische Weißabgleich störte jedoch die Ergebnisse des Farbtrackings enorm. Erst mit einer DV-Kamera bei der der Weißabgleich manuell ausschaltbar ist, konnte ein wesentlich bessere Bildqualität erreicht werden, welches per Firewire-Übertragung direkt in den Computer gespeist werden kann.
Probleme bereitete außerdem die korrekte Positionierung der Kamera innerhalb des Tisches. Bei einer zentrierten Befestigung der Kamera unterhalb des Spielfeldes wurde der Lichtstrahl des Beamers eingefangen und eine störende Lichtreflektion oberhalb des Spielfeldes entstand. Nur durch die Platzierung der Kamera am äußersten Rand des Würfels kann dies umgangen werden.

5.2 Programmierung

Die Programmierung des Spiels runKlotzen ist zum größten Teil in Processing 1.0.3 umgesetzt. Einzelne Teile wurden in Java implementiert und als Bibliothek in den Processing-Code eingebunden.

5.2.1 Umsetzung des Spiels

Grob strukturiert baut sich der runKlotzen-Code in vier Hauptbereiche auf. Diese entsprechen den vier Modi, in denen sich das Spiel befinden kann:
Konfigurationsmodus = modeConfiguration
Modus, um auf Spieler zu warten = modeWaitForPlayer
Initialisierungsmodus für das Spiel = modeInitialiseGame
Spielmodus = modeGame

Der Konfigurationsmodus ist nur beim Programmstart verfügbar. In ihm müssen die notwendigen Einstellungen für die Konfiguration des Systems vorgenommen werden. Dies ist notwendig, da bei jedem Einbau des Beamers und der Kamera, sowie je nach herrschenden Umgebungslichtverhältnissen andere Werte für bestimmte Parameter benötigt werden. Für die Konfiguration öffnet sich ein separates Fenster, welches das momentan von der Kamera aufgenommene Bild darstellt. Über Slider können in dieser Ansicht Einstellungen wie die Positionierung des Spielfelds, Anpassung des überlagernden Grids, die Trapezpunkte (dazu mehr in Kapitel 5.2.4) und die Feinjustierung der Farberkennung (vgl. Kapitel 5.2.2, siehe Abbildung 5.14) vorgenommen werden. Das Spielfeld ist bei der Konfiguration so zu positionieren, dass es sich erstens zentriert in der Mitte der Projektionsfläche befindet und zweitens komplett von der Kamera aufgenommen werden kann. Bei der Grid-Überlagerung (siehe Abbildung 5.13) muss eine möglichst genaue Übereinstimmung der dargestellten Quadrate mit denen des Spielfelds erreicht werden. Nur so kann später im Spiel garantiert werden, dass die Spielfiguren auf den richtigen Feldern des Spielbretts erkannt werden können.

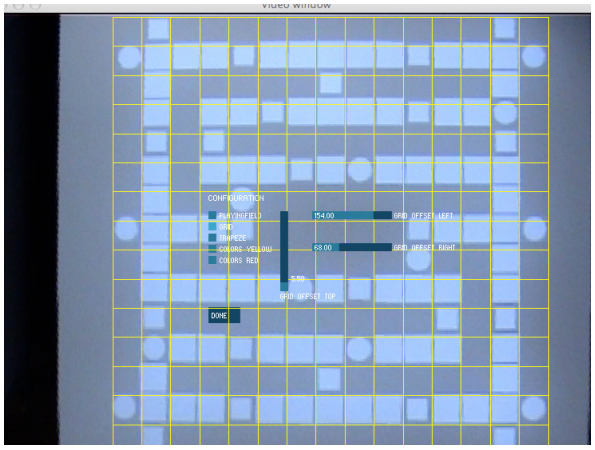


Abbildung 5.13: Grid-Überlagerung im Konfigruationsmodus

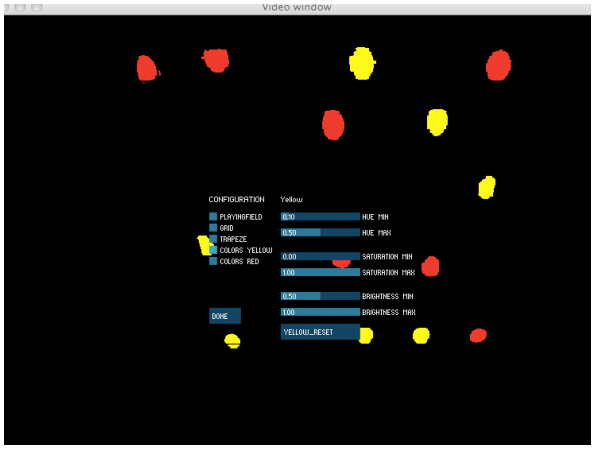


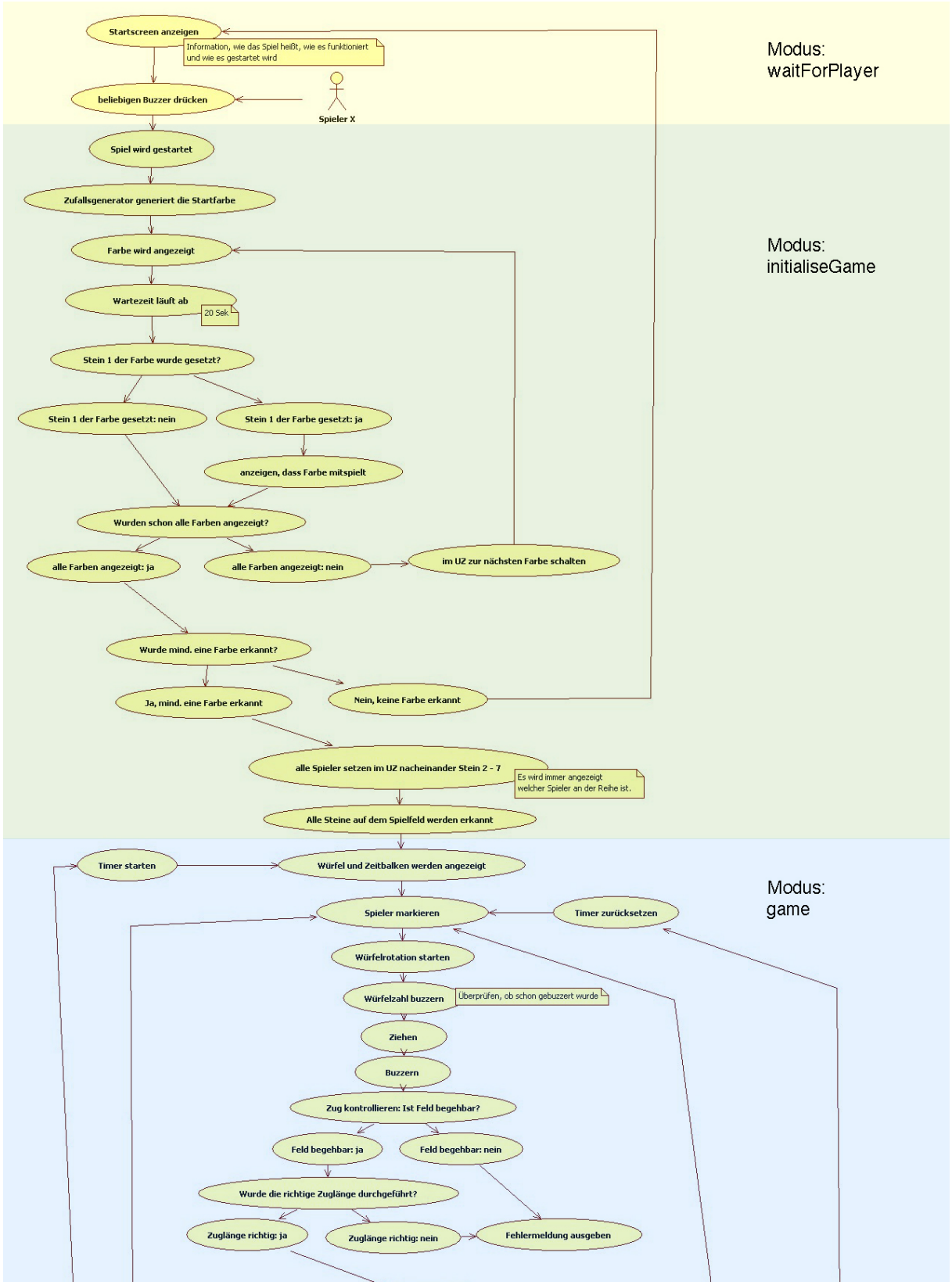
Abbildung 5.14: Farberkennung

Im zweiten Modus befindet sich das System nur, wenn gerade nicht gespielt wird. In dieser Zeit läuft ein kurzer Erklärungsfilm ab, der das Prinzip des Spiels erklärt. Wird in diesem Modus einer oder mehrere Buzzer betätigt, schaltet das System in den dritten Modus, den Initialisierungsmodus, um.

Der Initialisierungsmodus zeigt dem System wie viele Spieler am nächsten Spiel teilnehmen möchten. Dazu ermittelt das Programm zu Beginn per Zufall eine Startfarbe. Der Spieler, der diese Farbe gewählt hat beginnt. Der Teilnehmer hat nun 20 Sekunden Zeit seinen ersten Stein auf dem Spielfeld zu platzieren. Sollte er vor der festgelegten Zeit fertig sein, kann er dies durch Drücken seines Buzzers dem System mitteilen. Im Anschluss daran ist der nächste, im Uhrzeigersinn folgende Spieler an der Reihe. Sollte sich kein Spieler für eine Farbe finden, schaltet das System automatisch nach Ablauf der 20 Sekunden zum Nächsten. Hatten alle vier Farben die Möglichkeit einen Stein zu setzen, ermittelt das Programm aufgrund der gesetzten Spielsteine die Anzahl der Spieler und deren Farbe. Sollte kein Stein während dieser Phase gesetzt worden sein, wird zurück in den zweiten Modus geschaltet. Anderenfalls dürfen die erkannten Mitspieler nun reihum einen Spielstein nach dem anderen auf dem Spielfeld verteilen. Der Spieler, der aktuell am Zug ist wird durch seine beleuchtete „Homebase“ angezeigt. Haben alle Spieler ihre sieben Steine gesetzt, geht das Programm in den vierten Modus, den Spielmodus über.

Beim Start des Spielmodus werden in der Mitte des Spielfelds der Würfel und eine Zeitanzeige angezeigt. Nun findet für jeden Spieler dieselbe Prozedur statt. Der Würfel rotiert und durch Betätigen des Buzzers erhält der Spieler die Würfelzahl, die er mit einem seiner Spielsteine ziehen darf. Hat er seinen Zug vollendet drückt er auf den Buzzer. Das System ermittelt, welche Figur sich bewegt hat und prüft, ob sich diese auf einem Spezialfeld befindet. Außerdem findet eine Überprüfung hinsichtlich der Korrektheit des Zugs statt. Ist das Feld, auf dem sich die Spielfigur befindet nicht begehbar, signalisiert das System dieses und der Spieler muss seinen Zug wiederholen. Befindet sich die Figur auf einem normalen Feld wird mit dem nächsten Spieler fortgefahren. Handelt es sich allerdings bei dem betretenen Feld um ein Zielfeld überprüft das Programm, ob bereits alle Zielfelder des Levels belegt sind. Ist dies nicht der Fall, ist der nächste Spieler an der Reihe. Sind alle Zielfelder belegt findet eine Überprüfung hinsichtlich der Farben statt, die auf den Feldern stehen.

Werden alle Felder von derselben Farbe belegt, ist dieser Spieler der Gewinner. Anderenfalls wird in das nächste Level umgeschaltet und das Spiel fortgeführt. Hat der Spieler seine Figur auf ein Aktionsfeld gesetzt wird in der Mitte des Spielfelds die Rotation der Aktionen gestartet. Durch Buzzern kann der Spieler eine Aktion auswählen. Die gebuzzerte Aktion wird ebenfalls in der Spielfeldmitte angezeigt. Je nach Aktion wird ein spezieller weiterer Programmablauf angestoßen.
Wurde die Aktion Spielerrotation gewonnen hat das keinen Einfluss auf den Programmablauf. Es wird einfach zum nächsten Spieler weitergeschaltet. Handelt es sich um die Aktion Einfrieren sucht das Programm nach einer Veränderung auf dem Spielfeld. Der einzufrierende Spielstein wird vom Spieler umgedreht, sodass das Programm dort keinen Spielstein mehr erkennen kann. Dies ist durch die graue Färbung auf der Oberseite der Spielfiguren möglich. Die so erkannte Figur wird als eingefroren markiert und das Spiel geht weiter. Nach Vollendung einer Runde wird der Spielstein wieder freigegeben. Sollte die erkämpfte Aktion das Wettbuzzern sein, dann zeigt das Spielfeld in der Mitte rotierende Farben an. Alle Spieler dürfen nun gegeneinander buzzern. Der Gewinner dieser Aktion darf einen seiner Spielsteine auf ein Zielfeld stellen. Er beendet diese Aktion wieder mit dem Buzzer. Das System überprüft nun wieder, ob noch Zielfelder frei sind und verfährt wie im oben beschriebenen Fall, wenn ein Zielfeld betreten wurde.
Ein Level ist entweder zu Ende, wenn alle Spielfelder belegt sind oder wenn die Zeit für das Level abgelaufen ist. Auch in diesem Fall werden wieder die Steine auf den Zielfeldern überprüft und je nach Anzahl der vorhandenen Farben fortgefahren. Da das letzte Level nur ein Zielfeld enthält kann durch diesen Mechanismus auch der Gewinner ermittelt werden. Alle Felder, in diesem speziellen Fall das eine, werden von ein und derselben Farbe besetzt. Diese ist automatisch der Gewinner.
Abbildung 5.15 stellt den gesamten Programmablauf im Überblick dar. Der Konfigurationsmodus wird in dieser Grafik nicht gezeigt. Der dazugehörige Programmcode befindet sich auf der beiliegenden CD.


Modus:
waitForPlayer
Modus:
initialiseGame
Modus:
game
Startscreen anzeigen
Information, wie das Spiel heißt, wie es funktioniert und wie es gestartet wird
beliebigen Buzzer drücken
Spieler X
Spiel wird gestartet
Zufallsgenerator generiert die Startfarbe
Farbe wird angezeigt
Wartezeit läuft ab
20 Sek
Stein 1 der Farbe wurde gesetzt?
Stein 1 der Farbe gesetzt: nein
Stein 1 der Farbe gesetzt: ja
anzeigen, dass Farbe mitspielt
Wurden schon alle Farben angezeigt?
alle Farben angezeigt: ja
alle Farben angezeigt: nein
im UZ zur nächsten Farbe schalten
Wurde mind. eine Farbe erkannt?
Ja, mind. eine Farbe erkannt
Nein, keine Farbe erkannt
alle Spieler setzen im UZ nacheinander Stein 2 - 7
Es wird immer angezeigt welcher Spieler an der Reihe ist.
Alle Steine auf dem Spielfeld werden erkannt
Timer starten
Würfel und Zeitbalken werden angezeigt
Spieler markieren
Timer zurücksetzen
Würfelrotation starten
Würfelzahl buzzern
Überprüfen, ob schon gebuzzert wurde
Ziehen
Buzzern
Zug kontrollieren: Ist Feld begehbar?
Feld begehbar: ja
Feld begehbar: nein
Wurde die richtige Zuglänge durchgeführt?
Zuglänge richtig: ja
Zuglänge richtig: nein
Fehlermeldung ausgeben

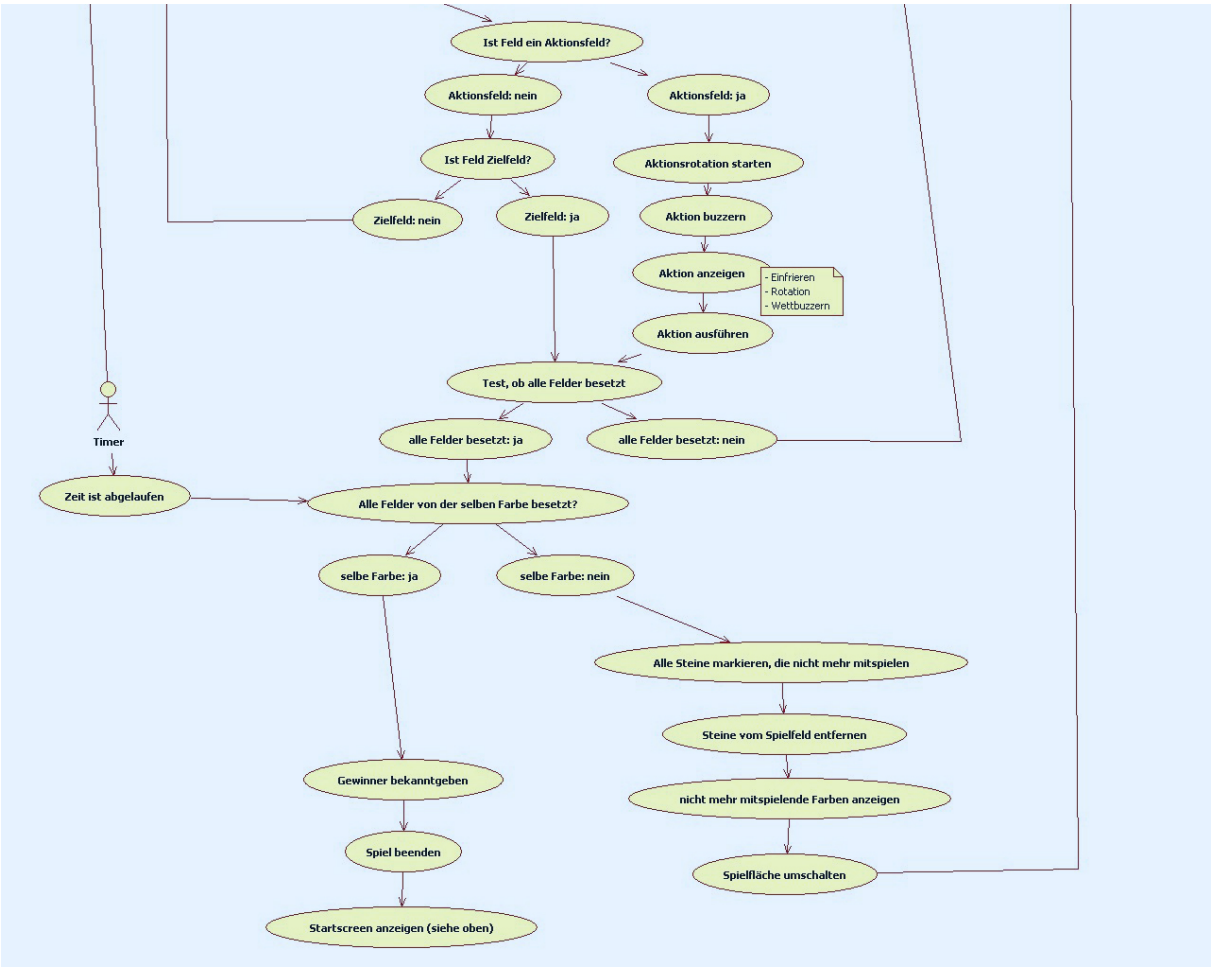


Abbildung 5.15: Programmablaufplan

5.2.2 Farberkennung

Die Farberkennung der einzelnen Spielsteine findet in einer ausgelagerten Java-Klasse statt. Auf Basis des HSV-Farbmodells (Farbton, Sättigung, Helligkeit) erstellt die Methode `cleanColors` in der Klasse `RKImageManipulator` ein Bild, das sich rein aus den gesuchten Farben und schwarz zusammensetzt. Auf der Basis von übergebenen HSV-Bereichen für die Farben Gelb und Orange / Rot entscheidet das Programm für jeden Pixel des übergebenen Bilds, ob er sich im gesuchten Farbbereich befindet. Ist dies der Fall wird je nach erkannter Farbe der Vollton Gelb oder Rot gesetzt. Das folgende Beispiel zeigt die Abfrage, ob es sich bei den Werten des momentan selektierten Pixels um einen Gelbwert handelt. Fällt die Antwort positiv aus, wird an dieser Stelle im Bild ein gelber Pixel gesetzt. Anderenfalls bleibt der Pixel schwarz.

```
if (hue >= hue1Min
                                    && hue <= hue1Max
                                    && saturation >= saturation1Min
                                    && saturation <= saturation1Max
                                    && brightness >= brightness1Min
                                    && brightness <= brightness1Max)
                    {
                          loOutput.setRGB(x, y, Color.yellow.getRGB());
```

Das so entstandene Bild enthält nur noch markierte Positionen an denen sich die gesuchten Farbwerte befinden.

5.2.3 Abbildung auf das Spielfeld

In einem weiteren Schritt werden die erkannten Farben auf Positionen im Spielfeld abgebildet. Hier kommt das im Konfigurationsmodus eingestellte Grid zum Einsatz. Mit Hilfe der einzelnen x- und y-Werte der Grid-Zellen wird das farbreduzierte Bild zellenweise abgesucht. Ein Zähler notiert, wie viele Pixel einer Farbe sich auf dem jeweiligen Feld befinden. Überschreitet die Anzahl der gezählten Pixel einen bestimmten Grenzwert, wird auf diesem Feld ein Spielstein dieser Farbe angenommen. Auf diese Weise wird das gesamte Bild abgesucht und auf eine 15 x 15-Matrix reduziert. Da das Spielfeld ebenfalls durch eine 15 x 15-Matrix repräsentiert wird, lassen sich die Positionen der einzelnen Felder auf dem Spielfeld mit den Positionen der Spielfiguren vergleichen.

```
for (int y=0;y<15;y++){    //y-direction
    for (int x=0;x<15;x++){  //x-direction
      int x1=int (x*rechteckb+gridOffsetLeft);   //grid field start x
      int y1=int (y*rechteckb+gridOffsetTop);    //grid field start y
      int x2=int (x1+rechteckb);                              //grid field end
x
      int y2=int (y1+rechteckb);                        //grid field end y
      for(int k=y1; k<=y2; k++){
        for(int l=x1;l<=x2;l++){
          int cA=colorArray[k][l];        //pixel at x / y
```

```
          switch(cA){
            case 1:
               counterYellow++;                    // count number of yellow
pixels
              break;
            case 2:
               counterRed++;                       //count number of red pi-
xels
          }
        }
        colorfield[y][14-x]=0;
      }
      if(counterRed>grenzwert)
        colorfield[y][14-x]=2;
      if(counterYellow>grenzwert)
        colorfield[y][14-x]=1;
      counterYellow=0;
      counterRed=0;
    }
  }
```

5.2.4 Trapezentzerrung

Durch die Verschiebung der Kamera aus dem Lichtkegel des Beamers heraus erhält die Kamera nur eine schräge Ansicht der Spielfläche (siehe Abbildung 5.16).

Diese Schrägsicht hat eine trapezartige Abbildung des gefilmten Spielfelds zur Folge. Um diese Abbildung für die Auswertung der Positionierung der einzelnen Spielsteine verwenden zu können, muss dieses Trapez zuerst aus dem Bild entfernt werden. Diese Aufgabe übernimmt die Methode cleanTrapezPerspective in der Klasse RKImageManipulator. Die während der Konfiguration eingestellten Werte für die Entzerrung des Trapezes werden zusammen mit dem Kamerabild an die Methode übergeben. Diese erzeugt durch eine perspektivische Transformation und einer Interpolation von Pixeln eine Abbildung der Spielfläche, die kein Trapez mehr enthält. Das so entstandene Bild wird zurückgegeben und für die weitere Verarbeitung im Programm verwendet.

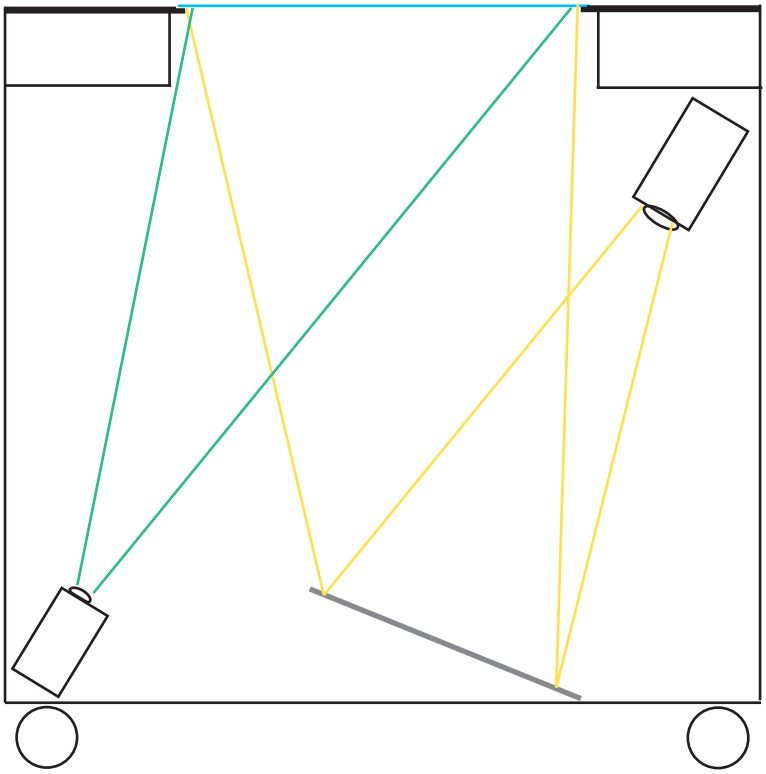


Abbildung 5.16: Positionierung der Kamera Abbildung

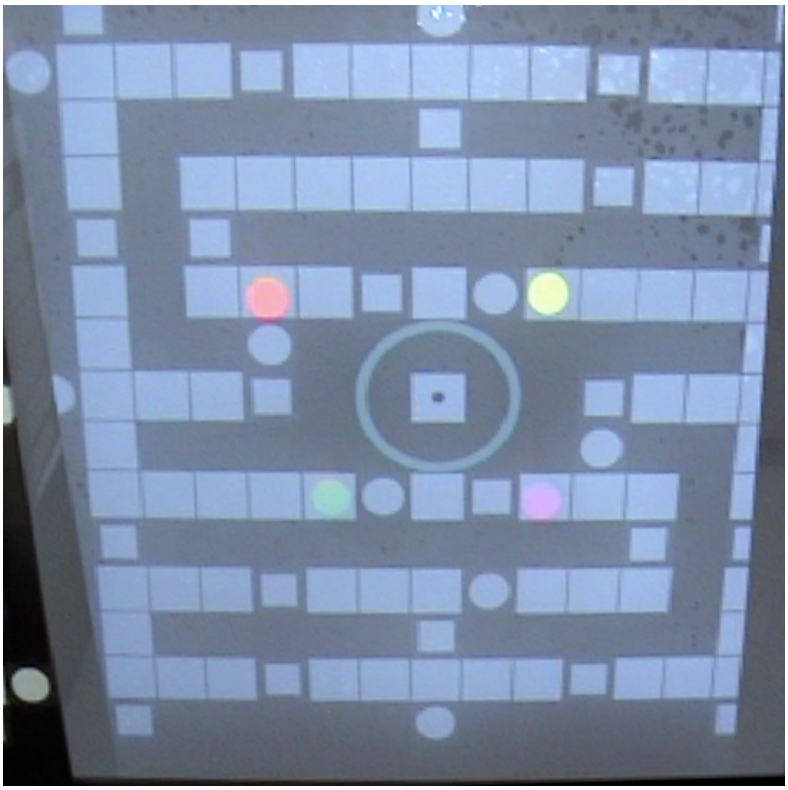


Abbildung 5.17: Trapez im der Darstellung des Spielfelds

5.3 Design

5.3.1 Spielfeld

Die Gestaltung des Spielfelds wird stark durch die technischen Rahmenbedingungen von Beamer und Farbtracking bestimmt. Die Farben der Spielfiguren sind für die Kamera am besten zu erkennen, wenn das volle Licht des Beamers auf sie trifft. Deshalb sind die begehbaren Felder des Spielbretts in weiß gehalten. Die inaktiven im Gegensatz dazu in 70% K als Hell-Dunkel-Kontrast. Aktions- und Zielfelder werden durch einen grauen Rahmen mit 30% K dargestellt, um wiederum für das Farbtracking eine möglichst große Fläche des Feldes weiß zu belassen. Ein Quadrat steht für Aktionsfelder, ein Kreis für Zielfelder, um die Analogie zu den Spielfiguren aufzugreifen – das Runde passt ins Runde. Letztendlich wird das Spielfeld rein in Graustufen visualisiert. Dies kommt darüber hinaus der optischen Wirkung zu Gute, da durch die Spielfiguren die vier Farben Rot, Gelb, Grün und Blau bereits verwendet werden. Eine weitere Farbe würde das Spielfeld unruhiger wirken lassen. Zentral in der Mitte und für jeden Spieler gleich gut sichtbar befinden sich die Zusatzinformationen wie Zeit, Würfel und Icons (siehe Abbildung 5.18).

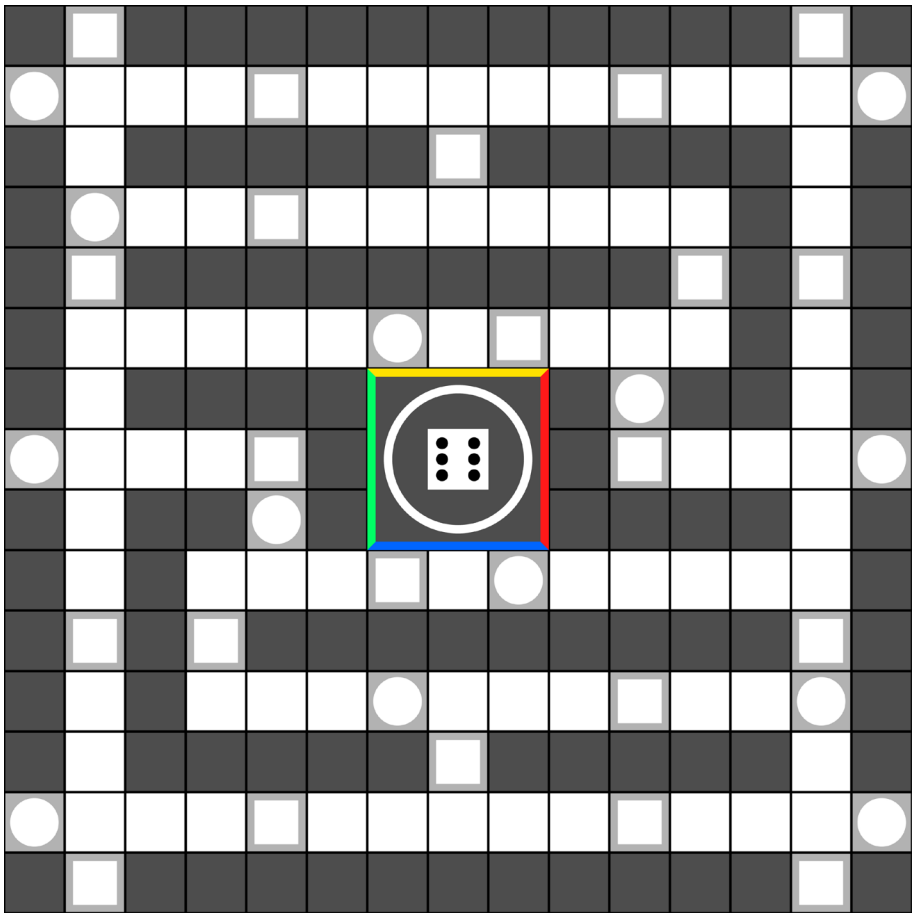
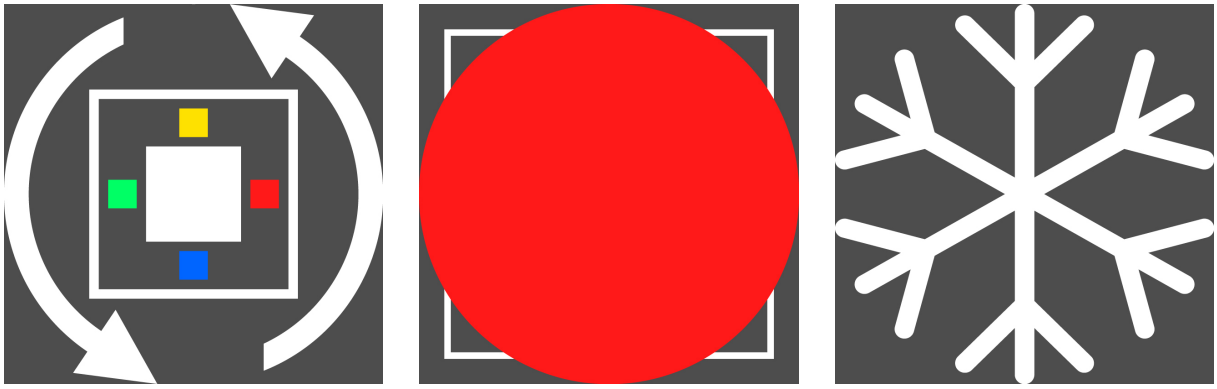


Abbildung 5.18: Spielfeld

5.3.2 Icons

Die Icons werden mittig auf dem Spielfeld dargestellt und müssen für jeden Spieler einfach und gleich gut zu erkennen sein, sprich von vier Seiten. Deshalb verzichten sie auf ein 3D-Optik und stellen ihre Symbolik durch eine Draufsicht dar. Zusätzlich sind die Icons eher detailarm gestaltet, dass ihre Funktion möglichst auf einen Blick wahrgenommen werden kann. Großen Wert wurde auch auf die Gleichwertigkeit aller Icons gelegt, folglich darf keines vom optischen Eindruck her dominieren. Aus diesem Grund werden alle Bildzeichen von einem imaginären Viereck von 60 x 60 Pixeln umrahmt. Schließlich wurde auch auf Verläufe verzichtet, da diese wegen der Leistung des Beamers verloren gehen. Farblich orientieren sich die Icons an der Optik des Spielfeldes und verwenden Rot, Gelb, Grün, Blau, Weiß und Grau. Dies ist möglich, da die Mitte des Spielbretts vom Farbtracking nicht überprüft wird (siehe Abbildung 5.19).



Farbwechsel | Wettbuzzern | Einfrieren

Gewinner | Fehler | Aktionsfeld

Abbildung 5.19: Icons

5.3.3 Logo

Der Name runKlotzen ist ein abgeleitetes Kunstwort von ranklotzen und beschreibt zum einen die dynamische Seite des Spiels (engl. run = rennen), als auch prägnante Elemente – die Klötzchen und der interaktive Tisch. Diese Eigenschaften sind auch im Logo wieder zu finden. Deshalb wird „Klotzen“ in den sehr plakativen Versalien der „Franklin Gothic Heavy“ geschrieben. Zusätzlich wirkt der Schriftzug wie aus einem massiven Klotz herausgestanzt. Für „run“ wird hingegen die dynamisch wirkende Schrift „Eurostile Bold“ verwendet und kursiv gesetzt, um die Bewegung noch zu verstärken. Farblich schlägt „Klotzen“ eine Brücke zum interaktiven Tisch, der ebenfalls in Anthrazit erscheint. „run“ hingegen kann in allen vier Farben der Spielfiguren dargestellt werden. Die Hauptfarbe ist allerdings Rot (siehe Abbildung 5.20).

Abbildung 5.20: Logos

5.3.4 Sound

In einem Spielkonzept wie runKlotzen, welches Brettspiel und digitale Elemente vereint, bedarf es für den Gesamteindruck auch eines akustischen Feedbacks, um die Wirkung zu steigern und das Spielkonzept abzurunden. Spezielle und einzigartige Sounds sind ein Grundelement von Konsolen- und Computerspielen und werden ebenso mit Spieltischen (z.B. Air Hockey) oder typischen Spielautomaten in Spielhallen assoziiert.
Das Spiel runKlotzen bedient sich einiger dieser Elemente und durch den Gebrauch von Buzzern wird zusätzlich der Charakter eines Quizshow-artigen Motivs erhöht.
Damit sollte das Sounddesign für runKlotzen irgendwo zwischen Spielhallenästhetik und Quizshows angesiedelt werden. Zu diesem Zweck wurde auf Plattformen, wie youtube und myvideo, Videos von älteren Quizsendungen und Spielhallenautomaten gesichtet und gegengehört. Einen gemeinsamen Nenner findet man dabei sehr schnell. Die meisten Sounds sind sehr kraftvoll, dynamisch und erklären oder untermalen den Spielverlauf und die Interaktion der Spieler mit dem Automaten.

5. umsetzung

Viele Automatenspiele und Quizshows verwenden kurze eingängige Melodien und Sounds für Fehler, Aktionen oder Rundenwechsel und gewonnene Passagen. Nachdem geklärt war an welcher Stelle und für welche Aktionen Sounds für runKlotzen benötigt werden, begann die Umsetzung.
Die Sounds und Melodien wurden auf Gitarre, Bass oder Keyboard komponiert und anschließend mit Audioprogrammen und digitalen Instrumenten (VST) umgesetzt. Digitale Sounds sind grundlegend besser zu bearbeiten und machen weniger Probleme mit Störgeräuschen beim Endmix.
Grundlegend verwendet runKlotzen Orgelsounds, die bewusst an die bekannten Zwischenspiele von Football und Baseballspielen erinnern sollen und somit auch anfeuernd und dynamisch wirken. Ferner untermalen diese Sounds den „Kampf“ mehrerer Kontrahenten gegeneinander und wirken anstachelnd.
Auch wurden Metaphern, wie etwa zarte Schneeglöckchen für das Einfrieren einer Spielfigur oder ein schriller und nervender, synthetischer Fehlerton, sollte ein Spieler seine Spielfigur falsch gesetzt haben, benutzt. Letzterer Sound funktioniert teils unterbewusst und gibt dem Spieler akustisch das Signal, das eine Spielaktion nicht korrekt ausgeführt wurde.
Quizshows benutzen sehr unterschiedliche Soundvariationen für den Klang eines Buzzers. Bei runKlotzen entschieden wir uns für einen tiefen, vollen synthetischen Klang, der eher an einen neutralen Türklingelsound erinnert und keine weitere Metapher, wie kreischende Leute oder ein Stimmsignal (etwa Ausruf: „Juhu!“) verwendet. Die vier Sounds für die vier im Spielkonzept verwendeten Buzzer unterscheiden sich in ihrer Tonhöhe, um somit die einzelnen Buzzer der jeweiligen Spieler besser unterscheiden zu können.
Die Soundvorschläge wurden beim wöchentlichen Meeting besprochen und nach den gemeinsamen Vorschlägen des Teams angepasst und verändert. Das Fehlen einiger Sounds stellten wir im Verlauf der Umsetzung des Spielkonzeptes fest. Erst später stellte sich heraus, dass man hier und da einige Passagen des Spiels mit einem zusätzlichen Sound besser untermalen könne und somit dem Spieler die notwendigen Spielzüge und Aktionen besser verdeutlicht.
Nachdem alle Sounds vorhanden waren, wurden diese gemixt und anschließend gemastert um Übersteuerungen, Schwankungen der Lautstärke und Intensität anzugleichen. Das Ergebnis ist ein abwechslungsreiches Soundesign, welches doch einheitlich aufeinander eingestimmt ist und auch nach mehrmaligen Spieldurchläufen nicht langweilig wird.

5.4 Website

Die Webseite für das Spiel runKlotzen wollten wir insgesamt sehr übersichtlich und minimalistisch halten. Die Seite soll grundlegend als Informationsseite dienen und bedarf keiner weiteren Features, wie eines Forums oder integrierten Spieles. Die Seite verzichtet bewusst auf Muster oder Hintergrundbilder. Sie sollte das Farb- und Designkonzept des Spieles wiederaufnehmen und das Spiel, die Regeln, das Team und das Projekt vorstellen.
Der Header zeigt das Logo des Spiels und einige Bilder von der Herstellung des Tisches, Umsetzung des Spielprinzips oder der verwendeten Trackingsoftware. Das Layout nimmt dabei ebenfalls die kubistische Form des Spieltisches und der Spielfelder auf, auch wenn die Bilder und Grafiken im rechteckigen 4:3 Format verwendet werden, was besser auf den Benutzer wirkte und auch weniger Platz in der Anordnung verschenkt.
Die Textfelder wurden in zwei rechteckige Bereiche geteilt, wobei die linke Seite vordergründig für die Präsentation von Bildern und die Rechte etwas größere die jeweiligen Texte zum passenden Menüpunkt enthalten. Die graue Farbe orientiert sich an den Designvorgaben des Spielfeldes und des Logos und nimmt diese dominierende Farbe grau abermals auf. Nachdem das Konzept und die Navigation der Webseite fertig waren, wurde diese mit Dreamweaver umgesetzt. Abbildung 5.21 zeigt einen Bildschirmabzug der Website.

Abbildung 5.21: Bildschirmabzug der Website

6. zusammenfassung und ausblick

6.1 Zusammenfassung

Interaktive Tische sind zurzeit sehr populär und bieten eine hervorragende Plattform der Kommunikation, deswegen haben wir uns entschlossen auf dieser Grundlage ein Spielkonzept zu entwickeln. Dieses soll sowohl reale Brettspiele-, als auch interaktive digitale Elemente verknüpfen. Hölzerne Spielfiguren in den vier Grundfarben werden auf einer projizierten Spielfläche gesetzt, wobei das Spiel mehrere Ebenen durchläuft, bis am Ende nur noch eine Spielfigur ein Zielfeld erreichen kann und damit der Gewinner ist.
Einige der populärsten Multitouch-Tische und ähnliche interaktive Spielideen wurden in unsere ausführliche Recherche eingeschlossen. Letztendlich entschlossen wir uns Processing und ein Arduino-Board für die Umsetzung des Konzeptes zu benutzen.
Nach der Entwicklung eines vollständigen Spielprinzips und der Festsetzung aller Spielregeln, konnte die praktische Umsetzung beginnen. In Zusammenarbeit mit der hauseigenen Schreinerei der Hochschule Augsburg wurden die Designentwürfe des Tisches in die Praxis umgesetzt. Dabei wurden Aussparungen für Plexiglasflächen und Halterungen für die verwendeten Komponenten, wie Kamera, Beamer, Spiegel und Audiosystem berücksichtigt. Als Alleinstellungsmerkmal verwendet runKlotzen zusätzlich Buzzer, umgebaute Grobhandtaster der Firma Moeller, wie man sie aus vielen Quizsendungen kennt.
Nach ersten Versuchen mit einer Webcam, griffen wir zu einer DV-Kamera, die aufgrund des manuellen Weißabgleichs bessere Ergebnisse in der Übertragung des aufgenommenen Bildes lieferte. Ein eigens geschriebenes Programm steuert das Spielsystem und reagiert auf die per Farbtracking erkannten Spielfiguren und das Benutzen der Buzzer.
Das Design von Spielfeldern ist durch die Rahmenbedingen von Kamera und Beamer bestimmt, während verwendete Spielicons den Fokus auf die Usability für den Spieler legen. Das Name „runKlotzen“ ist ein zusammengesetztes Kunstwort, welches im Logodesign durch die Verwendung passender Schriftarten „run“ sowie „Klotzen“ grafisch unterstreicht.
Das Sounddesign untermalt die Spielabläufe und orientiert sich grob an bekannten Spielhallensounds und die in Quiz-Shows verwendeten kurzen Melodien.
Die Website dient als Infoseite und greift grafische Elemente des Spielfeldes und des kubistischen Gesamtdesigns auf.
In seiner Gesamterscheinung und Umsetzung, sowie der Integration von Buzzern in das Spielkonzept, stellt runKlotzen ein einzigartiges interaktives Spielvergnügen dar.

6.2 Aktueller Stand

Zur Projektabgabe am 29. Juli 2009 ist das Projekt runKlotzen zum größten Teil funktionsfähig. Das Farbtracking stellt allerdings immer noch die schwerwiegendste Problematik dar. Zum momentanen Zeitpunkt kann das Spiel nur mit zwei Farben gespielt werden, da die Mustererkennung des geplanten Kreises auf den Spielsteinböden (vgl. Kapitel 5.1.2) aus Mangel an Zeit nicht mehr implementiert werden konnte. Auch wechselnde Lichtverhältnisse erschweren eine genaue Erkennung der Farben. Ferner fehlen einige geplante visuelle Feedbacks.
Die Umsetzung des Tisches ist komplett abgeschlossen. Sowohl die für die Befestigung von Beamer, Kamera und Spiegel notwendigen Halterungen sind aufeinander abgestimmt, als auch der Einbau von Lichtern und Buzzern wurde abgeschlossen. Die elektronische Schaltung inklusive Arduino ist komplett funktionsfähig. Das Soundsystem ist ebenfalls zum Einsatz bereit. Auf Seiten der Software wurde die Entwicklung des Konfigurations- und Initialisierungsmodus sowie das Warten auf Spieler vollständig umgesetzt. Die Spielelogik wurde implementiert, ist aber noch nicht zur Genüge getestet. Die Designphase sowie die Entwicklung und Umsetzung der benötigten Icons und Filmsequenzen sind vollendet. Auch alle nötigen Sound-Dateien sind vorhanden. Die Projektwebsite mit aktuellen Informationen, Beschreibungen und Fotos ist unter www.runklotzen.de erreichbar.

6.3 Weiteres Vorgehen

Der Schwerpunkt der weiteren Entwicklung des Projekts runKlotzen sollte den Fokus auf die Erkennung der verschiedenen Spielsteine setzen. Es muss überdacht werden, ob eine optimalere Option neben der Farberkennung bestünde. Auch weitere Tests der Spielelogik sind notwendig. Ist die Entwicklungsphase des Spiels abgeschlossen, sind Usability-Tests mit unterschiedlichen Nutzern durchzuführen, um die Intuitivität zu überprüfen. Die so erlangten Erkenntnisse und eventuell während der Untersuchung aufgetretene Fehler im Programm sollten in einen zweiten Entwicklungszyklus fließen, um das Spiel runKlotzen möglichst benutzerfreundlich und fehlerfrei gestalten zu können.

Anhang 1: Arduino-Code

```
//number of colors:
//blue = 1
//green = 2
//yellow = 3
//red = 4

//buzzers
int buzzerBlue = 4;
int buzzerGreen = 5;
int buzzerYellow = 6;
int buzzerRed = 7;

//values of buzzers
int valueBlue = LOW;
int valueGreen = LOW;
int valueYellow = LOW;
int valueRed = LOW;

//buzzerPressed
int buzzerBluePressed = 0;
int buzzerGreenPressed = 0;
int buzzerYellowPressed = 0;
int buzzerRedPressed = 0;


//LEDs
int ledBlue = 9;
int ledGreen = 10;
int ledYellow = 11;
int ledRed = 12;

//data received from the serial port
int data;

void setup(){
  //set pins as INPUT
  pinMode(buzzerBlue, INPUT);
  pinMode(buzzerGreen, INPUT);
  pinMode(buzzerYellow, INPUT);
  pinMode(buzzerRed, INPUT);
```

```
  //set pins as OUTPUT
  pinMode(ledBlue, OUTPUT);
  pinMode(ledGreen, OUTPUT);
  pinMode(ledYellow, OUTPUT);
  pinMode(ledRed, OUTPUT);

  Serial.begin(9600);    //Start serial communication at 9600bps
}

void loop(){
 toProcessing();
 fromProcessing();

  // wait a bit to not overload the port
  delay(10);
}

void toProcessing(){
  // read the value on all digital input
  valueBlue = digitalRead(buzzerBlue);
  if(valueBlue != buzzerBluePressed)
    buzzerBluePressed = valueBlue;
  else
    valueBlue = 0;

  valueGreen = digitalRead(buzzerGreen);
  if(valueGreen != buzzerGreenPressed)
    buzzerGreenPressed = valueGreen;
  else
    valueGreen = 0;

  valueYellow = digitalRead(buzzerYellow);
  if(valueYellow != buzzerYellowPressed)
    buzzerYellowPressed = valueYellow;
  else
    valueYellow = 0;

  valueRed = digitalRead(buzzerRed);
  if(valueRed != buzzerRedPressed)
    buzzerRedPressed = valueRed;
  else
    valueRed = 0;
```

```
int value = valueRed << 3 | valueYellow << 2 | valueGreen << 1 | valueBlue;

  //send information to processing if at least one buzzer is pressed
  if(value != 0)
    Serial.write(value);
}

void fromProcessing(){
  //If data is available to read at the serial port, store it as data
  if (Serial.available()){
    data = Serial.read();
  }

  //led power on
  digitalWrite(ledBlue, data & 1);
  digitalWrite(ledGreen, (data & 2) >> 1);
  digitalWrite(ledYellow, (data & 4) >> 2);
  digitalWrite(ledRed, (data & 8) >> 3);
}
```