

# Billiard-AI

Ein intelligenter Billardtisch

**BSc Thesis**

Studienrichtung:

Informatik - Computer Perception and Virtual Reality

Autor:

Lukas Seglias, Luca Ritz

Dozent:

Markus Hudritsch

Experte:

Andreas Dürsteler

Datum:

10. Dezember 2021



# Inhaltsverzeichnis

<b>1 Zusammenfassung</b>	<b>1</b>
<b>2 Einführung</b>	<b>3</b>
<b>3 Ziele</b>	<b>5</b>
3.1 Planung . . . . .	6
3.1.1 Meilensteine . . . . .	6
<b>4 Billiard-AI</b>	<b>9</b>
4.1 Vorbereitungen . . . . .	9
4.2 Risikoanalyse . . . . .	10
4.3 Detektion von Snooker-Kugeln . . . . .	11
4.4 Tracking und Stabilisierung der Kugelpositionen . . . . .	12
4.5 Klassifikation . . . . .	14
4.6 Suchalgorithmus . . . . .	16
4.6.1 Kandidatensuche . . . . .	16
4.6.1.1 Expansion einer Kugel . . . . .	17
4.6.1.2 Bewertungsfunktion . . . . .	25
4.6.2 Berechnung der Initialgeschwindigkeit . . . . .	27
4.6.2.1 Reibungsverlust über Bahn . . . . .	27
4.6.2.2 Elastischer Stoß bei Kugelkollision . . . . .	27
4.6.2.3 Bandenkollision . . . . .	28
4.6.3 Modellierung physikalisches System . . . . .	29
4.6.3.1 Ereignisse und ihre Repräsentation . . . . .	30
4.6.3.2 Kantenfunktion . . . . .	31
4.6.3.3 Layer . . . . .	31
4.6.3.4 Beispiel eines Graphen . . . . .	31
4.6.4 Simulation . . . . .	32
4.6.4.1 Reibungsverlust über die Zeit . . . . .	33
4.6.4.2 Elastischer Stoß zweier Kugeln . . . . .	33
4.6.4.2.1 Behandlung Topspin . . . . .	34
4.6.4.3 Bandenreflektion . . . . .	35
4.6.4.4 Kollisionsprüfung . . . . .	36
4.6.4.5 Ereignis - Energie-Transfer über Kugelkollision . . . . .	36
4.6.4.5.1 Performanceverbesserungen . . . . .	37
4.6.4.6 Ereignis - Energie-Transfer über Bandenkollision . . . . .	38
4.6.4.7 Ereignis - Totaler Energieverlust . . . . .	39
4.6.4.8 Ereignis - Verlassen des Systems . . . . .	39
4.6.4.9 Ereignis - Rollen . . . . .	40
4.6.4.10 Bewertungsfunktion . . . . .	40
4.6.5 Animation . . . . .	42
4.6.6 Tiefensuche . . . . .	44
4.6.6.1 Regeln für Snooker . . . . .	44
4.7 Berechnungsprozess . . . . .	46
4.8 Anwendung . . . . .	48
4.9 Infinity-Modus . . . . .	50
4.9.1 Tracking des Spielballs . . . . .	50
4.9.2 Bewegungserkennung . . . . .	51
4.9.3 Verlust von Kugeln . . . . .	51
4.9.4 Neue Kugeln . . . . .	51
4.9.5 Robustheit . . . . .	51
4.9.6 Zeitliche Stabilität . . . . .	51

4.10 Verwandte Arbeiten	52
4.10.1 Automatic pool stick vs. strangers	52
4.10.2 Pool Live AR	52
4.10.3 Innovationsgrad	52
<b>5 Resultate</b>	<b>53</b>
5.1 Klassifikation	53
5.2 Detektion	53
5.3 Bestimmen des Rollreibungskoeffizienten	55
5.4 Energieverlust	56
5.4.1 Energieverlust bei Kugelkollision	57
5.4.2 Energieverlust bei Bandenkollision	57
5.5 Suche	57
5.6 Vergleich Simulation und Realität	59
<b>6 Weitere Arbeiten</b>	<b>61</b>
<b>7 Fazit</b>	<b>63</b>
<b>8 Anhang</b>	<b>73</b>
8.1 Herleitung Startgeschwindigkeit auf Basis bekannter Endgeschwindigkeit unter Einbezug von Reibung	73
8.2 Herleitung Startgeschwindigkeit auf Basis der Zeit und Distanz unter Einbezug von Reibung	74
8.3 Herleitung Initialgeschwindigkeit bei Kugel-Kollision	78
8.4 Herleitung Ereignis - Totaler Energieverlust	79
8.5 Herleitung Ereignis - Kollision dynamischer Objekte	81
8.6 Herleitung Ereignis - Kollision mit Bande	83
8.7 Herleitung Ereignis - Kollision mit Ziel	85
8.8 Herleitung Ereignis - Rollen	87
8.9 Gleitdistanz	89
8.10 Herleitung Beschleunigung	89
8.11 Bestimmung Reibungskoeffizient	90
8.12 Herleitung Linie-Linie-Schnittpunkt	92
8.13 Herleitung Kugeltransformation über Bandenreflektion	93
8.13.1 Den gespiegelten Zielpunkt bestimmen	93
8.14 Simulation Algorithmusverbesserung	97

# 1 Zusammenfassung

TODO: Zusammenfassung



## 2 Einführung

Wie vieles andere ist auch das Erlernen des Billardspiels eine schwierige Sache. Es stellen sich Fragen wie „Welche Kugel soll man anspielen?“, „Wie soll man die Kugel anspielen?“ oder „Wie hält man den Queue richtig?“. Darauffolgend gibt es noch diverse weitere Überlegungen, welche den Profi vom Anfänger unterscheiden. Wie in anderen Spielen auch, ist hier Weitsicht gefragt. Es geht also nicht nur darum, eine Kugel zu versenken, sondern auch den Spielstand so zu verändern, dass optimal weitergespielt werden kann. Das Stichwort ist im Billard vor allem die Platzierung der weissen Kugel.

Diese Arbeit widmet sich nun einem Assistenten, der die Einführung in die schwierige Kunst des Billardspiels vereinfachen soll.



## 3 Ziele

Es wird ein Billardassistent entwickelt, welcher aufgrund des aktuellen Spielstands einen Stoss berechnet, der einerseits einfach durchzuführen ist und andererseits eine optimale Ausgangslage für den nächsten Stoss bieten soll. Es gab bereits einige Vorarbeiten[Luk21a], wie aus dem Kapitel 4.1 zu entnehmen ist.

Die Ziele für diese Arbeit sind nun die Klassifikation der Kugeln, die Entwicklung einer effizienten gerichteten Suche nach einem Stoss, die darauffolgende, möglichst realitätsnahe Simulation sowie einige Verbesserungen bei der Darstellung.

Es werden zwei verschiedene Spielmodi angestrebt. Beim ersten Modus kann der Spieler eine Kugel oder einen Kugeltypen (Farbe) wählen, nach dem gesucht wird. Daraufhin werden ihm alle gefundenen Resultate in geordneter Reihenfolge nach Schwierigkeit dargestellt. Die Suche kann über mehrere Stösse gehen und dabei werden auch die Regeln des Spiels beachtet. Dies ist wichtig, um die optimale Platzierung der weissen Kugel zu eruieren. Nach einem Stoss kann der Spieler wiederum eine Wahl treffen. Der zweite Modus ist der sogenannte „Infinity-Mode“. Die Idee dahinter ist die vereinfachte Einführung des Spielers. So soll das System bei stabilem Zustand automatisch eine Suche nach allen Kugeln starten und das beste Ergebnis anzeigen. Sobald der Spieler den Stoss durchgeführt hat, wird das System wiederum automatisch eine Suche starten. Dadurch kann ein Spieler optimal lernen, welche Stösse einfach sind und wo eine Kugel mit welcher Geschwindigkeit getroffen werden muss, um das gewünschte Ergebnis zu erzielen. Die Regeln spielen in diesem Fall eine untergeordnete Rolle und werden daher ignoriert.

Es ist weiterhin anzumerken, dass es in erster Linie um Snooker-Billard geht. Dies hat mehrere Gründe. Einerseits soll in dieser Arbeit nicht die Klassifikation der Kugeln im Zentrum stehen, sondern die Suche nach einem optimalen Stoss. Es wird angenommen, dass dies mit Snooker-Kugeln einfacher ist als mit Pool-Billard-Kugeln. Andererseits wird das Projekt zusammen mit einem Unternehmen durchgeführt, welches eventuell auch einen kommerziellen Ansatz verfolgen will. Da grössere Turniere wie Weltmeisterschaften in Snooker ausgetragen werden, kam schnell der Wunsch auf, das Hauptaugenmerk darauf zu legen. Nichtsdestotrotz wird die Anwendung so abstrakt gehalten, dass sie mit wenig Aufwand auf Pool-Billard portiert werden könnte. Dies bildet jedoch kein Ziel der Bachelor-Thesis.

## 3.1 Planung

Die initiale Planung beinhaltet eine Auflistung der Tätigkeiten, deren zugewiesenen Meilensteine sowie deren Schätzung in PT (Personen-Tage). Jedem Arbeitspaket wird eine ID zugewiesen, welche bei der Zeiterfassung verlinkt wird. Das Total der zu vergebenden PT beträgt 90.

ID	Name	Meilenstein	Schätzung in PT
T-1	Klassifikation der Kugeln	M-1	6
T-2	Aufsetzen Dokumentation	M-1	1
T-3	Beschreibung Suchalgorithmus	M-1	3
T-4	Implementation Suchalgorithmus	M-1	5
T-5	Beschreibung der physikalischen Eigenschaften für die einfache Suche	M-1	6
T-6	Implementation der einfachen Suche und deren Bewertungsfunktion	M-1	14
T-7	Beschreibung der physikalischen Eigenschaften für die erweiterte Suche	M-2	6
T-8	Implementation der erweiterten Suche und deren Bewertungsfunktion	M-2	8
T-9	Überprüfen/Verbessern der Detektionsgenauigkeit	M-1	6
T-10	Video erstellen	M-3	2
T-11	Plakat schreiben	M-3	2
T-12	Book-Eintrag schreiben	M-3	1
T-13	Präsentation des Finaltags vorbereiten	M-3	2
T-14	Präsentation der Verteidigung vorbereiten	M-3	2
T-15	Finalisieren Dokumentation andere Arbeiten	M-3	4
T-16	Projektmanagement	Kein	4
T-17	Effizienz Erfassung und Steigerung der einfachen Suche	M-1	0
T-18	Effizienz Erfassung und Steigerung der erweiterten Suche	M-2	0
T-19	Dokumentation der Resultate der einfachen Suche	M-1	6
T-20	Dokumentation der Resultate der erweiterten Suche	M-2	2
T-21	Umbau in Unity	M-1	4
T-22	Infinity-Modus	M-1	5
O-1	Suche über mehrere Spielstände	M-2	
O-2	Detektion des Queues in 2D	M-2	
O-3	Detektion des Queues in 3D	M-2	
O-4	Stossberechnung anhand detektiertem Queue in 2D	M-2	
O-5	Stossberechnung anhand detektiertem Queue in 3D	M-2	
O-6	Spielerabhängige Heuristik	M-2	
O-7	Live-Verfolgung und Darstellung der Kugeln	M-2	
Total			90

Tabelle 3.1: Ziele

### 3.1.1 Meilensteine

Es werden drei Meilensteine definiert, welche auch aus optionalen Zielen bestehen können. Werden diese nicht erreicht, so gilt der Meilenstein trotzdem als erreicht. Die Deadlines ergeben sich aus den Schätzungen der zugewiesenen Arbeitspakete.

**Meilenstein 1 - 15.11.2021** Das Ziel ist eine sehr einfache simple Suche. Darunter zu verstehen ist eine Lösung, welche einen direkten Treffer findet (Weiss -> Kugel -> Loch).

*Code Deliverables:*

#### Klassifikation - T-1

Alle Kugeln können entsprechend ihrer Farbe klassifiziert werden.

#### Suchalgorithmus für einfache Suche - T-4, T-6, T-17

Ein direkter Stoss wird in akzeptabler Zeit gefunden.

## **Unity-Umbau - T-21**

Unity ist bereit für den Einsatz. Zum Umbau gehören insbesondere die Farbe der Markierung der Kugeln und deren Bahnen. Weiterhin muss Unity mehrere Suchergebnisse anzeigen können.

## **Infinity-Modus - T-22**

Der Spieler erhält automatisch einen Vorschlag für einen Stoss, wenn die Kugeln auf dem Tisch stillstehen. Sobald der Spieler einen Stoss ausgeführt hat und die Kugeln wieder stillstehen, wird der nächste Stoss vorgeschlagen. Vorgeschlagen werden Stösse, welche eine beliebige Kugel ins Loch spielen.

*Dokumentation Deliverables:*

### **Klassifikation - T-1**

Das Vorgehen der Klassifikation wie deren Resultate und Genauigkeit sind dokumentiert.

### **Suchalgorithmus für Suche - T-3**

Der Algorithmus der Suche ist theoretisch und mit Pseudocode beschrieben. Die theoretische Beschreibung muss nicht gänzlich mit der effektiven Implementation übereinstimmen, da diese auf Performance optimiert wird.

### **Resultate der einfachen Suche - T-19**

In den Resultaten ist die Genauigkeit und Performance des einfachen Suchvorgangs beschrieben.

### **Physik der einfachen Suche - T-5**

Die benötigte Physik der einfachen Suche ist beschrieben.

### **Bewertungsfunktion - T-6**

Die Bewertungsfunktion der einfachen Suche ist dokumentiert.

**Meilenstein 2 - 13.12.2021** Das Ziel ist eine erweiterte Suche, die auch indirekte Stösse über weitere Kugeln oder Banden finden kann. Optional sollen auch mehrere Stösse berücksichtigt werden.

*Code Deliverables:*

### **Suchalgorithmus für erweiterte Suche - T-8, T-18**

Ein indirekter Stoss wird in akzeptabler Zeit gefunden.

### **Suchalgorithmus über mehrere Stösse - O-1**

Es werden mehrere Spielstände bei der Suche berücksichtigt.

### **Queue in 2D detektieren - O-2**

Der Queue wird als 2D-Objekt detektiert.

### **Queue in 3D detektieren - O-3**

Der Queue wird mittels Tiefeninformationen der Kamera als 3D-Objekt detektiert.

### **Stossberechnung anhand detektiertem 2D-Queue - O-4**

Der Stoss wird je nach Haltung des Queues in 2D berechnet. Es wird angenommen, dass der Queue zentral auf die weisse Kugel gerichtet ist.

### **Stossberechnung anhand detektiertem 3D-Queue - O-5**

Der Stoss wird je nach Haltung des Queues in 3D berechnet. Der Queue muss nicht zentral auf die weisse Kugel gerichtet sein.

### **Spielerabhängige Heuristik - O-6**

Je nach Spieler kann eine andere Heuristik zur Bewertung der Stösse eingestellt werden. Durch die Unterscheidung können für professionelle Spieler erfolgsversprechendere schwerer durchzuführende und für Anfänger eher leichtere Stösse gefunden werden.

### **Live-Verfolgung und Darstellung der Kugeln - O-7**

Die Kugeln werden ohne Benutzereingabe getrackt und deren Position über den Projektor dargestellt.

*Dokumentation Deliverables:*

### **Physik der erweiterten Suche - T-7**

Die benötigte Physik der erweiterten Suche ist beschrieben.

### **Resultate der erweiterten Suche - T-20**

In den Resultaten ist die Genauigkeit und Performance des erweiterten Suchvorgangs beschrieben.

**Bewertungsfunktion - T-8**

Die Bewertungsfunktion der erweiterten Suche ist dokumentiert.

**Meilenstein 3 - 17.01.2022** Das Ziel ist der Abschluss aller Arbeiten zu denen auch Plakat, Book-Eintrag oder Video gehören.

*Deliverables:*

**Video - T-10**

Das finale Video ist erstellt.

**Plakat - T-11**

Das Plakat ist erstellt.

**Book-Eintrag - T-12**

Der Book-Eintrag ist erstellt.

**Präsentation für Finaltag - T-13**

Die Präsentation/Ausstellung für den Finaltag ist vorbereitet.

**Präsentation für Verteidigung - T-14**

Die Präsentation für die Verteidigung ist vorbereitet.

**Finalisieren der Arbeiten - T-15**

Die Dokumentation wie auch der Code sind abgeschlossen.

# 4 Billiard-AI

## 4.1 Vorarbeiten

Diese Bachelor-Thesis basiert auf Vorarbeiten einer früheren Projektarbeit. Diese hatte zum Ziel, einige grundlegende Funktionalitäten bereitzustellen. In Abbildung 4.1 ist der Zyklus angegeben, welcher alle Aufgaben zusammenfasst. In einem ersten Schritt wird der aktuelle Spielstand über eine Kamera detektiert. Daraus kann eine textuelle Beschreibung abgeleitet werden. Diese dient wiederum als Input für das Kernstück, die Suche nach einem optimalen Stoss. Als Ausgabe dieses Schrittes erfolgt eine detaillierte Animation des auszuführenden Stosses. Die Animation wird über einen Projektor auf dem Billardtisch dem Spieler zugänglich gemacht. Dieser kann den Stoss ausführen, was zu einer neuen Situation führt und der Zyklus beginnt von vorne.

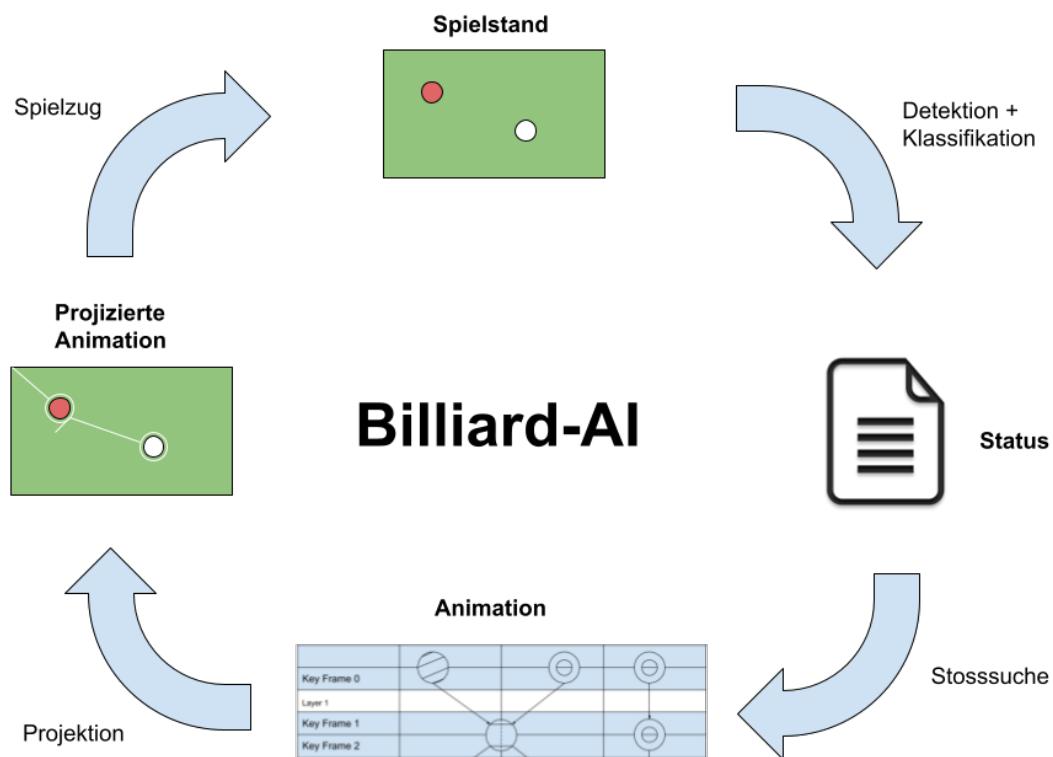


Abbildung 4.1: Billiard-AI-Cycle

Es sind nun die Schritte der Detektion des Spielstands wie auch die Projektion und Darstellung der Animation auf dem Spieltisch bereits umgesetzt. Dabei ging es in erster Linie um die Frage der Repräsentation in geeignetem Koordinatensystem sowie die Messung der Genauigkeit dieser Übersetzung und der darauf basierenden Anzeige über den Projektor.

Die Bachelor-Thesis widmet sich zuerst der Frage nach der Klassifikation in Schritt eins, wie auch der Suche nach dem optimalen Stoss in Schritt zwei.

## 4.2 Risikoanalyse

Es gibt diverse Risiken, die während dieser Arbeit eintreten können. Um das Bewusstsein dafür zu stärken, wird vorgängig eine Risikoanalyse durchgeführt, wobei es um die Identifikation wie auch die Zuordnung deren Auftretenswahrscheinlichkeit und Auswirkungen geht. Weiterhin werden geeignete Massnahmen definiert, die entweder die Eintrittswahrscheinlichkeit reduziert oder bei Auftreten angegangen werden können.

Die Risiken werden im Detail aufgelistet. Die Spalte „WK“ steht für die Eintrittswahrscheinlichkeit, welche in den Wahrscheinlichkeiten „Gering“, „Möglich“, „Wahrscheinlich“, „Sehr Wahrscheinlich“ angegeben wird. Die Spalte „AW“ steht für die Auswirkungen, welche in den Größen „Klein“, „Mittel“, „Gross“ angegeben wird.

ID	Risiko	Massnahme	WK	AW
R-1	Verlust von Programm-Sourcen	Führen eines Repositories auf GIT, welches das Wiederherstellen eines bestimmten Standes erlaubt. Zudem wird jeden Freitag ein Backup des GIT-Standes auf eine externe Festplatte geschrieben, sollte der unwahrscheinliche Fall eintreten, dass GIT nicht mehr verfügbar sein sollte oder seine Bestände verliert.	Möglich	Mittel
R-2	Ausfall der Arbeitsgeräte	Es stehen Ersatzgeräte bereit, welche sofort zum Einsatz kommen könnten.	Möglich	Klein
R-3	Krankheitsausfall der Teammitglieder	Es wird wenn möglich von Zuhause aus gearbeitet, um das Risiko einer Ansteckung zu vermindern.	Möglich	Gross
R-4	Parallele Entwicklung derselben Funktionen	Durch eine anfängliche Planung der Arbeitspakete, wie auch den ständigen Austausch und Einsatz von Pair-Programming an geeigneten Stellen, wird das Risiko stark reduziert. Sollte es trotzdem eintreten, sind die Auswirkungen marginal, da die ständige Kommunikation dies sofort aufdecken würde.	Gering	Klein
R-5	Unterschätzen der Komplexität	Um zumindest ein brauchbares Resultat vorweisen zu können, wurde der erste Meilenstein möglichst simpel gehalten.	Wahrscheinlich	Gross
R-6	Verpassen wichtiger Termine	Es wird ein Kalender mit allen Terminen geführt, welcher mehrmals eine Erinnerung anzeigt.	Gering	Gross

Tabelle 4.1: Risiken

Die identifizierten Risiken werden in der Abbildung 4.2 für eine bessere Übersicht eingetragen.



Abbildung 4.2: Risikoanalyse

### 4.3 Detektion von Snooker-Kugeln

Aufgrund eines Bildes des Billardtisches soll der Spielstand mit der Position aller Kugeln bestimmt werden. Der in dieser Arbeit verwendete Detektionsalgorithmus stammte von der vorherigen Projektarbeit[Luk21b] und funktioniert für Snooker-Kugeln. In der Vorarbeit wurde die Detektion dafür vorgesehen, per Knopfdruck durchgeführt werden konnte. Mit dieser Arbeit wurde darauf aufbauend eine Live-Detektion implementiert, welche kontinuierlich Bilder vom Tisch macht, den Spielstand detektiert und dem Benutzer über den Projektor anzeigt. Dadurch entsteht ein Feedback-Loop, weil die dargestellten Augmentationen im Bild, das verarbeitet wird, ersichtlich sind. Ein Eingabebild für die Detektion ist in Abbildung 4.3 dargestellt und zeigt diesen Feedback-Loop.



Abbildung 4.3: Feedback-Loop in der Detektion

Die dargestellten Augmentationen können zu fälschlicherweise detektierten Kugeln führen. Um diesem Problem entgegenzuwirken wurden die Parameter der Detektion angepasst, damit die Augmentationen grösstenteils entfernt werden können. Da allerdings nicht alle restlos entfernt werden können, besteht ein Restrisiko für fälschlicherweise detektierte Kugeln.

Des Weiteren wurde die Performance der Detektion erhöht, um die Live-Detektion aufzuwerten. In der bisherigen Detektion

wurden drei separate Circle Hough transform[Wik21a] angewendet, eine für jede der drei Gruppen von Kugeln, in die das Bild mittels Segmentation aufgeteilt wurde[Luk21b]. Zwei dieser drei konnten ohne einen bemerkbaren Verlust in der Qualität der Detektion zusammengefasst werden.

## 4.4 Tracking und Stabilisierung der Kugelpositionen

Wie in Kapitel 4.3 beschrieben wurde eine Live-Detektion implementiert, welche kontinuierlich die Bilder der Kamera ausliest, eine Detektion ausführt und anschliessend die Positionen der Kugeln über den Projektor visualisiert. Aufgrund von Rauschen auf den Eingabebildern wird die Position jeder Kugel nicht immer an der exakt gleichen Stelle wie in der vorherigen Detektion erkannt. Dadurch entsteht auf dem Tisch ein unerwünschtes Rauschen in der Position der Kugeln, was sich durch schnelle, kleine Bewegungen der dargestellten Kreise bemerkbar macht. Dieses Problem ist in Abbildung 4.4 visualisiert. Diese Unruhige Darstellung gilt es zu verbessern, damit der Spieler während des Stosses nicht davon abgelenkt wird.



Abbildung 4.4: Visualisierung des Rauschens in den detektierten Positionen einer roten Kugel über 5 Frames. Die schwarzen Kreise visualisieren die im entsprechenden Frame detektierte Position, die grauen Kreise die zuvor detektierten Positionen.

Der Lösungsansatz ist die Betrachtung des Spielstandes über die Zeit, um die detektierten Positionen abzuglättten und das Rauschen zu unterdrücken. Dies erfordert ein Tracking, welches die detektierten Kugeln von einem Frame  $F_{t-1}$  mit den detektierten Kugeln von Frame  $F_t$  verbindet. Es muss demnach bestimmt werden, wo eine Kugel aus Frame  $F_{t-1}$  in Frame  $F_t$  ist.

Um dies zu erreichen wird jede Kugel  $K_t$  im aktuellen Frame  $F_t$  geprüft und diejenige Kugel  $K_{t-1}$  in Frame  $F_{t-1}$  gesucht, die der Kugel  $K_t$  in der euklidischen Distanz am nächsten ist. Das bedeutet, es wird zu jeder aktuellen Kugel die Vorgängerkugel aus dem vorherigen Frame gesucht.

Für diese Zuweisung sind bewegte Kugeln problematisch. Im Fall dass sich Kugel  $K_t$  weit von ihrer Position in Frame  $F_{t-1}$  entfernt hat, könnte es zu einer Verwechslung kommen, weil die richtige Kugel nicht mehr die Nächstgelegene ist. Daher darf die Distanz der nächsten Kugel aus Frame  $F_{t-1}$  eine Maximaldistanz  $D$  nicht überschreiten. Diese Maximaldistanz  $D$  muss kleiner als der Radius der Kugeln gewählt werden, damit möglichst keine Verwechslungen auftreten. Außerdem muss diese Maximaldistanz grösser als das durchschnittliche Rauschen in der detektierten Position sein, damit das Tracking bei stillstehenden Kugeln erfolgen kann.

Diese Suche nach der nächstgelegenen Kugel des vorherigen Frames wird für jede Kugel des aktuellen Frames durchgeführt. Kugeln des vorherigen Frames, welche einer aktuellen Kugel zugewiesen wurden, werden nicht erneut zugewiesen. Kugeln, für die keine Vorgängerkugel gefunden werden konnte, werden als *nicht getrackt* markiert.

Für jede getrackte Kugel können die detektierten Positionen über ein Zeitfenster von  $N$  Frames aufgezeichnet werden. Diese History der Positionen kann anschliessend genutzt werden, um einen *gleitenden Durchschnitt*[Wik21e] der Position zu berechnen. Der gleitende Durchschnitt führt bei stillstehenden Kugeln zu einer deutlichen Reduktion des Rauschens. Bei bewegten Kugeln ist zu beachten, dass diese nicht zwingend über alle Frames getrackt werden können, weil sie zwischen den Frames zu weit gerollt sein könnten. Die History wird für Kugeln, die nicht mehr getrackt werden konnten, gelöscht. Dadurch ist die History leer, wenn eine Kugel wieder getrackt werden konnte und wird wieder gefüllt, solange das Tracking erfolgreich ist. Die Berechnung des gleitenden Durchschnitts auf einigen wenigen Positionen aus der History ist suboptimal, weil zu wenige Datensätze vorhanden sind, um eine effektive Rauschunterdrückung zu erreichen. Um dieses Problem zu lösen wird die Durchschnittsposition einer getrackten Kugel erst verwendet, wenn diese mindestens  $L < N$  Frames getrackt wurde. Sofern eine Kugel beispielsweise erst 3 Frames lang getrackt wurde, wird noch kein gleitender Durchschnitt berechnet. Bei stillstehenden Kugeln ist diese Bedingung unproblematisch, weil diese bei kleinem  $L$  schnell erfüllt ist.

Bei bewegten Kugeln ist die Durchschnittsposition weiterhin problematisch, weil diese zwischen den detektierten Positionen liegt und dadurch hinter der tatsächlichen Position zurückliegt, siehe Abbildung 4.5.

Diese Problematik wird entschärft, indem die Durchschnittsposition  $P_{avg}$  mit der aktuellen Position  $P_{current}$  aufgrund ihrer Distanz  $d$  zusammengerechnet wird [Unk21g]. Sofern die Distanz der beiden Positionen gross ist, soll die aktuelle Position

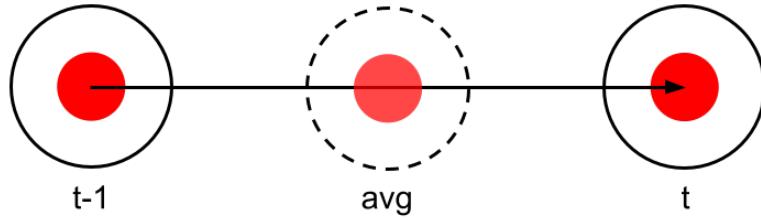


Abbildung 4.5: Detektierte Positionen derselben Kugel zum Zeitpunkt  $t - 1$  und  $t$ . Der Durchschnitt dieser beiden Positionen liegt hinter der aktuellen Position.

verwendet werden, weil in diesem Fall die Durchschnittsposition zeitlich verzögert hinter der Position liegt, siehe Abbildung 4.5. Sofern die Distanz klein ist, dann kann die Durchschnittsposition verwendet werden, um das Rauschen zu unterdrücken. Der Faktor  $f \in [0; 1]$ , mit dem die beiden Positionen kombiniert werden, wird über die Maximaldistanz  $D$  des Trackings wie folgt definiert:

$$f = \min\left(\frac{d}{D}, 1\right) \quad (4.1)$$

Das Minimum verhindert, dass  $f > 1$  wird, wenn die Distanz  $d > D$ . Anschliessend kann der Faktor  $f$  verwendet werden, um die geglättete Position  $P_{new}$  einer getrackten Kugel zu berechnen:

$$P_{new} = f \cdot P_{current} + (1 - f) \cdot P_{avg} \quad (4.2)$$

Für Kugeln, welche nicht getrackt werden konnten, wird die im aktuellen Frame detektierte Position verwendet.

Das Tracking von bewegten Kugeln bedingt, dass sich die Kugeln zwischen zwei aufeinanderfolgenden Frames nicht zu weit bewegen, um noch immer als dieselbe Kugel identifiziert zu werden. Dies ist bei kleinen Geschwindigkeiten unproblematisch, allerdings treten bei höheren Geschwindigkeiten *motion blur* auf und die Kugeln "teleportieren" von einem Ort zum anderen.

Damit ist eine Rauschunterdrückung in der Anzeige der detektierten Positionen aller Kugeln erreicht, ohne dabei eine sichtbare Latenz zu verursachen. Es gilt zu beachten, dass das hier vorgestellte Tracking lediglich zum Zweck hat, die dargestellten Positionen zu stabilisieren und nicht schnell bewegende Kugeln über mehrere Frames tracken zu können.

## 4.5 Klassifikation

In Kapitel 4.3 wurde beschrieben, wie die Positionen aller Kugeln mithilfe eines Bilds des Spielstands detektiert werden. Um den Spielstand zu verstehen ist es relevant zu wissen, welche Kugel welche Farbe hat. Im Snooker gibt es die Farben Weiss, Rot, Gelb, Pink, Grün, Blau, Braun und Schwarz. Deshalb muss jede detektierte Kugel anhand eines Bildausschnitts in eine dieser Klassen eingeteilt werden. Der Ablauf der Klassifikation der Snooker-Kugeln wird nachfolgend beschrieben.

Zunächst liegt zu jeder Kugel deren Position in Pixel- sowie Modell-Koordinaten vor und der Radius der Kugeln ist in Pixel bekannt[Luk21c]. Aufgrund dieser Informationen wird von jeder detektierten Kugel ein Bildausschnitt des RGB-Bildes rund um die Kugelposition definiert, der für die Klassifikation verwendet werden kann. Der Bildausschnitt ist quadratisch, wodurch in den Ecken des Bildes noch das Grün des Billardtisches ersichtlich ist.

Dieses Grün könnte die Klassifikation verfälschen, da damit mehr grüne Pixel auf dem Bild sind und eine Kugel eher als grün klassifiziert werden könnte. Um dieses Problem zu umgehen wird der Pixelradius der Kugel mit einem Faktor von 0.5 skaliert, um den Bildausschnitt zu verkleinern und den grünen Tisch so zu entfernen.

Als Nächstes wird zur Rauschunterdrückung ein Gauss-Filter mit der Kernelgrösse von 5x5 auf den Bildausschnitt angewendet. Anschliessend wird das RGB-Bild in den HSV-Farbraum[Wik21c] konvertiert, um die Farbanalyse zu vereinfachen. Aufgrund des HSV-Bildes wird pro Kanal (Hue, Saturation und Value) ein Histogramm berechnet und der häufigste Wert ermittelt. Damit wurde die Information des Bildausschnitts auf drei Zahlen, eine pro HSV-Kanal, reduziert. Diese drei Werte können als Position des Bildausschnitts im HSV-Raum interpretiert werden.

Pro Kugelfarbe kann definiert werden, in welche HSV-Bereiche diese Kugeln gehören. Bspw. gehören Rot, Pink, Braun eher in den roten Bereich des Hue-Kanals. Blau, Gelb und Grün sind vom roten Bereich abgrenzbar. Die weisse und schwarze Kugel sind am ehesten im Value-Kanal klassifizierbar.

Für eine genauere Unterscheidung wurden Bilder von verschiedenen Spielständen gemacht und anhand dieser die genauen Parameter für die Klassifikation gefunden. In diesen Trainingsbildern ist eine Kugel jeder Klasse an unterschiedlichen Stellen platziert worden, um die ungleichmässige Ausleuchtung des Billardtischs zu berücksichtigen. In Abbildung 4.6 sind zwei Trainingsbilder pro Klasse aufgeführt.

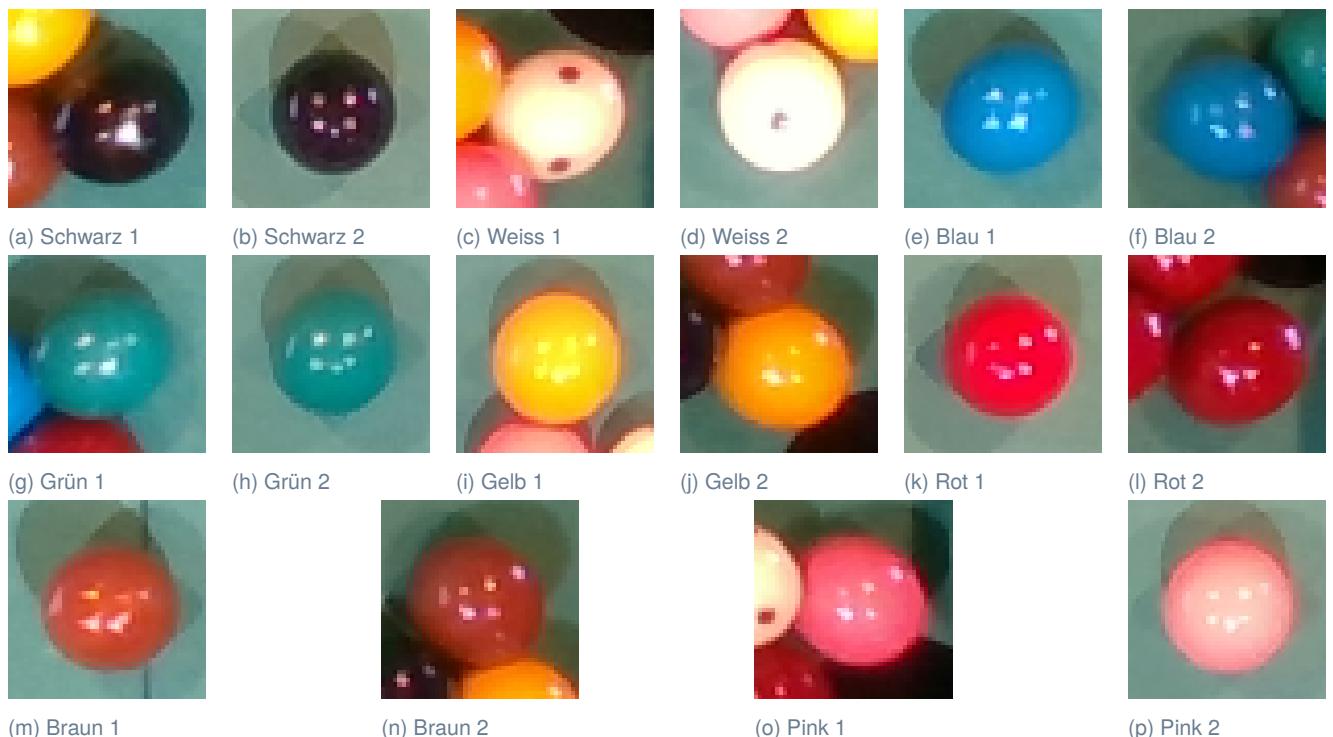


Abbildung 4.6: Auswahl aus den Trainingsbildern

Das Resultat der Analyse dieser Daten sind Wertebereiche für die HSV-Kanäle pro Klasse, wodurch eine grobe Klassifikation, ermöglicht wird. Diese Wertebereiche bieten keine eindeutige Klassifikation, sondern werden verwendet, um die potentiellen Klassen einer Kugel zu finden. So kann es sein, dass eine blaue Kugel aufgrund ihres Hue-Kanals klar als Blau erkannt wird. Es kann aber auch sein, dass eine braune Kugel sowohl Rot, Pink und Braun als potentielle Klassen erhält.

Es muss demnach noch eine genauere Unterscheidung gemacht werden, sobald die potentiellen Klassen bekannt sind. Anhand der Trainingsbildern wurden die Durchschnittswerte der häufigsten Werte der HSV-Kanäle pro Klasse berechnet. Diese Durchschnittswerte bilden die Cluster-Zentren jeder Klasse und können für die Klassifikation verwendet werden. Dazu wird die Distanz der HSV-Position des zu klassifizierenden Bildausschnitts zu jedem Cluster-Zentrum bestimmt. Der Cluster mit der kleinsten Distanz gibt dem Bildausschnitt seine Klasse.

Sofern ein Bildausschnitt mehrere potentielle Klassen in der Prüfung durch Wertebereiche erhalten hat, wird die Distanz zwischen dessen HSV-Position und dem Cluster-Zentrum jeder der potentiellen Klassen verglichen. Die Klasse mit der kleinsten Distanz zur HSV-Position des Bildausschnitts wird angenommen. Demnach ist diese Klassifikation in der Idee ein *k-nearest neighbors* Algorithmus[Wik21d], wobei keine Trainingsdatensätze als Nachbaren verwendet werden, sondern die bereits gefundenen Cluster-Zentren.

Sofern ein Bildausschnitt keine potentiellen Klassen erhalten hat, wird diesem die Klasse *unbekannt* zugewiesen.

Mit der hier beschriebenen Klassifikation ist die Farbe jeder Kugel nach deren Detektion bekannt und es sind alle Informationen für die weitere Verwendung in der Suche und Darstellung vorhanden.

## 4.6 Suchalgorithmus

Die Suche wird über zwei Schritte durchgeführt. Der erste Schritt besteht aus der Suche nach einem Lösungskandidaten, wobei vom Ziel aus ein Stoß gesucht wird, welcher das Potenzial hat, eine Kugel in diesem Ziel zu versenken. Das Resultat dieses Schrittes ist lediglich der Geschwindigkeitsvektor der weißen Kugel. Ob dieser Stoß tatsächlich das Resultat zur Folge hat, welches er voraussagt, wird im zweiten Schritt geprüft. Der Geschwindigkeitsvektor der weißen Kugel kann mit unterschiedlichem Betrag in einen Simulationsschritt eingegeben werden. Der Simulationsschritt wird als Resultat ein physikalisches System wie in Kapitel 4.6.3 ergeben.

Im Nachfolgenden wird auf die verschiedenen Schritte und deren Funktionsweise sowie die optimale parallele Durchführung der Berechnungen eingegangen.

### 4.6.1 Kandidatensuche

Die Kandidatensuche, im Folgenden Suche genannt, wird über eine klassische Graphensuche durchgeführt, wobei der vollständige Graph alle möglichen Stöße enthält. Für die Suche gibt es zwei Möglichkeiten, entweder wird bei der weißen Kugel gestartet und von dort ein Stoß gesucht, welcher eine andere Kugel ins Loch spielt, oder es wird bei einem oder mehreren Löchern gestartet und von dort ein Stoß gesucht, welcher von der weißen Kugel ausgehend eine andere Kugel ins Loch spielt.

Nachfolgend wird die Suche beschrieben, welche beim Loch, dem Ziel, startet, eine einzulochende Kugel findet und anschließend den Stoß bis zur weißen Kugel zurück sucht. Dementsprechend ist der Root-Knoten des Suchbaumes das zu treffende Ziel (Loch). Da ein handelsüblicher Billardtisch mehrere Löcher hat, muss pro Loch eine separate Suche durchgeführt werden.

Bei der Durchführung eines Expansionsschrittes werden ausgehend von einem Knoten im Suchbaum dessen Nachfolger-Knoten ermittelt. Diese stellen im Fall vom Root-Knoten Kugeln dar, welche in dieses Loch gespielt werden könnten. Aus diesen Kugeln werden Kugel-Knoten gebildet, welche diese Kugeln entweder auf direktem Wege oder indirekt über die Bande in das Loch spielen lassen sollen. Ausgehend von diesen Kugel-Knoten, werden deren Nachfolger-Knoten in weiteren Expansionsschritten ermittelt, welche wiederum Kugel-Knoten darstellen. Diese Kugel-Knoten stellen dann Kugeln dar, welche die Kugel des Vorgänger-Kugel-Knotens entweder auf direktem Wege oder indirekt über die Bande treffen sollen. Sofern ein Kugel-Knoten die weiße Kugel darstellt, so ist dieser Kugel-Knoten ein Endzustand und damit ist der Stoß über die Kette von Nachfolger- zu Vorgänger-Kugel-Knoten definiert.

Zur Veranschaulichung des Prinzips folgt ein Beispiel. Es wird vereinfacht angenommen, dass der Tisch nur ein Loch hat. Für mehrere Ziele ergeben sich mehrere Suchbäume, einen pro Loch. In Abbildung 4.7 erfolgt die Eingabe des Suchalgorithmus in Form des Root-Knotens. Es wird nur das zu treffende Ziel definiert. Auf der rechten Seite des Tisches ist der Suchbaum dargestellt.

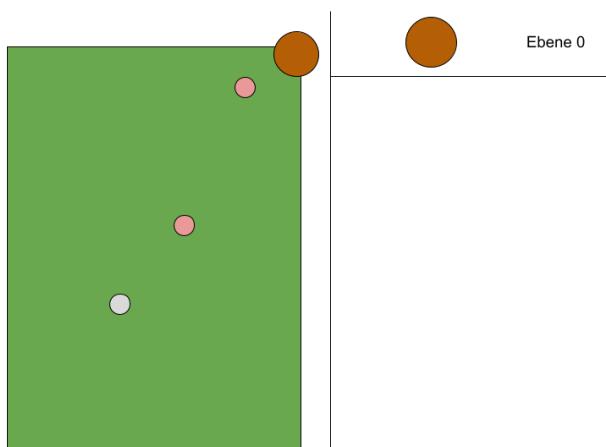


Abbildung 4.7: Kandidatensuche 1

In einem zweiten Schritt wird die einzulochende Kugel definiert. Es kommen lediglich die beiden roten Kugeln in Frage. Nachfolgend wird der Pfad weiter betrachtet, bei dem die rote Kugel, welche näher beim Loch ist, gewählt wurde. Abbildung 4.8 zeigt, dass der Suchbaum um einen Knoten erweitert wurde.

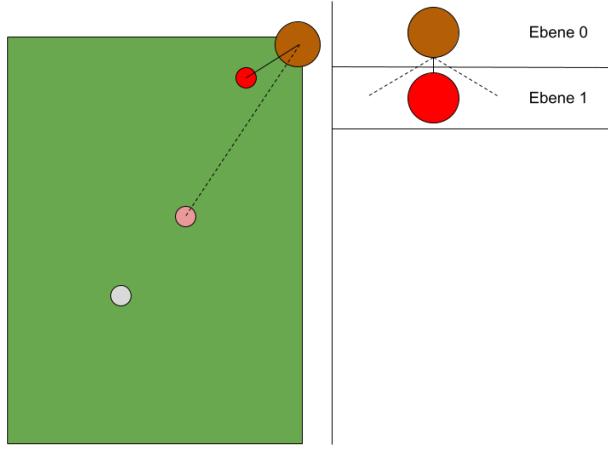


Abbildung 4.8: Kandidatensuche 2

In Abbildung 4.9 erfolgt der letzte Schritt. Hier sind verschiedene Optionen möglich, bspw. könnte die weiße Kugel direkt oder über die Bande an die zuvor gewählte rote Kugel gespielt werden. Es könnte aber auch die andere rote Kugel an die zuvor gewählte rote Kugel gespielt werden. Hier wird der Fall betrachtet, dass die weiße Kugel indirekt über eine Bande an die zuvor gewählte rote Kugel gespielt wird.

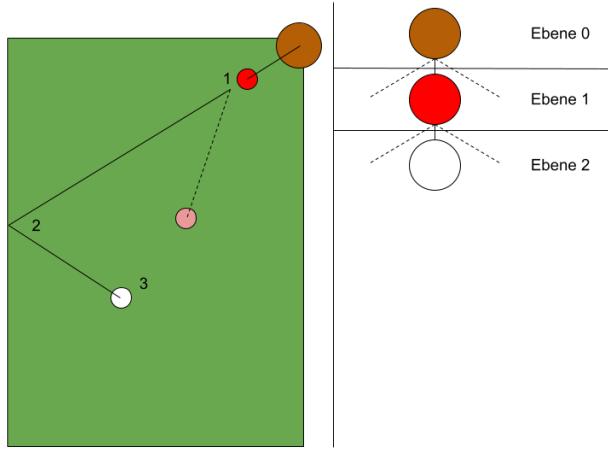


Abbildung 4.9: Kandidatensuche 3

Algorithmus 1 verdeutlicht den Ablauf der „Expand-Funktion“. Zuerst wird eine leere Liste namens „nodes“ angelegt. Diese wird danach mit Nodes gefüllt, welche entweder durch einen Stoss über eine weitere Kugel oder indirekt über die Bande Zustände kommen. Die Nodes bilden das Ergebnis der Funktion.

```
Function expand(node: Node, constantObjects: list) —> list[Node]
  nodes ← list()
  nodes ← append(expandBalls(node, constantObjects), nodes)
  nodes ← append(expandBank(node, constantObjects), nodes)
  return nodes
```

**Algorithmus 1:** Algorithmus zur Durchführung eines Expansionsschritts bei der Kandidatensuche

#### 4.6.1.1 Expansion einer Kugel

Bei einer Expansion einer Kugel wird der nächste Zielpunkt berechnet. Das Prinzip wird in Abbildung 4.10 veranschaulicht. Es gibt einen bereits bekannten Zielpunkt  $T$ , wo die Kugel hingespult werden muss. Weiterhin ist die aktuelle Position  $S$  der Kugel bekannt. Dazwischen kann der Vektor  $\vec{d}$  gebildet werden.

$$\vec{d} = S - T \quad (4.3)$$

Um die Kugel zum gewünschten Ziel zu befördern, wird der neue Zielpunkt  $Z$  mithilfe des Vektors  $d$  und dem bekannten Kugelradius  $r$  berechnet.

$$Z = S + 2 \cdot r \cdot \hat{d} \quad (4.4)$$

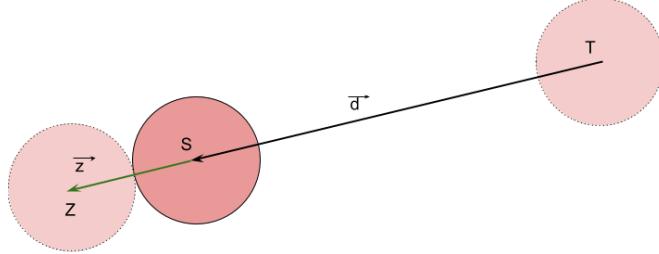


Abbildung 4.10: Kugelexpansion

Eine Kugel kann ebenso über eine oder mehrere Bänder expandiert werden. Dies geschieht über einen geometrischen Ansatz durch eine Spiegelung[DJo13] des Zielpunkts an der entsprechenden Bande, über welche gespielt werden soll. In Abbildung 4.11 wird ein Suchschritt über eine Bande visualisiert. Es wird ein Punkt  $A$  an einer Bande gesucht, zu welchem die Kugel  $B$  gespielt werden muss, um die Kugel  $C$  zu treffen. Dieser Bandenkollisionspunkt  $A$  wird mithilfe eines Spiegelpunkts  $\bar{C}$  des Punktes  $C$  an der Bande berechnet. In der Abbildung 4.11 sind zwei Beispiele ersichtlich. Im ersten Fall wird für die

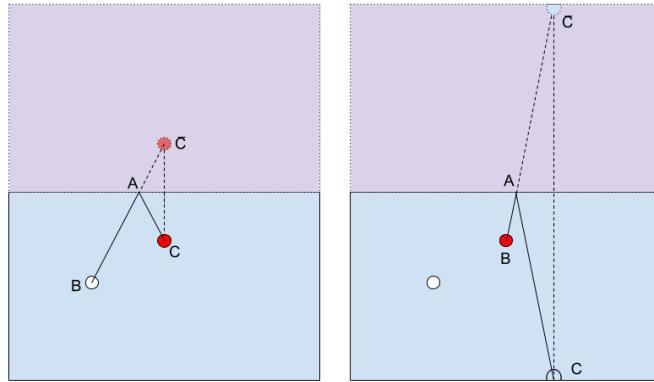


Abbildung 4.11: Tiefe über Bande erreichen mittels Reflektion

weiße Kugel ein Kollisionspunkt an der oberen Bande bestimmt, um die rote Kugel zu treffen. Im zweiten Fall wird für die rote Kugel ein Kollisionspunkt an der oberen Bande bestimmt, um ins Ziel zu treffen.

Gegeben sind die folgenden Angaben (Parameter werden gross geschrieben, Variablen dagegen klein):

$$B = \begin{pmatrix} B_X \\ B_Y \end{pmatrix}, C = \begin{pmatrix} C_X \\ C_Y \end{pmatrix}, \bar{C} = \begin{pmatrix} \bar{C}_X \\ \bar{C}_Y \end{pmatrix}, \quad (4.5)$$

Der Schnittpunkt kann über zwei Geraden berechnet werden, die über die Parameterform gegeben sind. Eine dieser Geraden liefert der gespiegelte Punkt  $\bar{C}$  mit  $B$ .

$$\vec{I} = \vec{\bar{C}} - \vec{B} \quad (4.6)$$

$$L_1 = \vec{B} + \lambda_1 \cdot \vec{I} \quad (4.7)$$

Die andere Gerade ist über die Bande (Rail) gegeben, wobei  $R_1$  der Startpunkt und  $R_2$  der Endpunkt der Bande ist:

$$R_1 = \begin{pmatrix} R_{1X} \\ R_{1Y} \end{pmatrix}, R_2 = \begin{pmatrix} R_{2X} \\ R_{2Y} \end{pmatrix} \quad (4.8)$$

$$\vec{r} = \vec{R}_2 - \vec{R}_1 \quad (4.9)$$

$$L_2 = \vec{R}_1 + \lambda_2 \cdot \vec{r} \quad (4.10)$$

Der Kollisionspunkt  $A$  lässt sich über  $\lambda_1$  der Gleichung 4.11 mithilfe der Linie  $L_1$  bestimmen<sup>1</sup>.

$$\lambda_1 = \frac{r_x \cdot B_y - r_y \cdot B_x + R_{1,x} \cdot r_y - R_{1,y} \cdot r_x}{l_x \cdot r_y - l_y \cdot r_x} \quad (4.11)$$

Es gilt zu beachten, dass es sich bei der Billardkugel nicht um einen unendlich kleinen Punkt handelt und daher deren Radius berücksichtigt werden muss. Das Problem wird in Abbildung 4.12 veranschaulicht. Einerseits wird als Zielposition  $C$  nicht

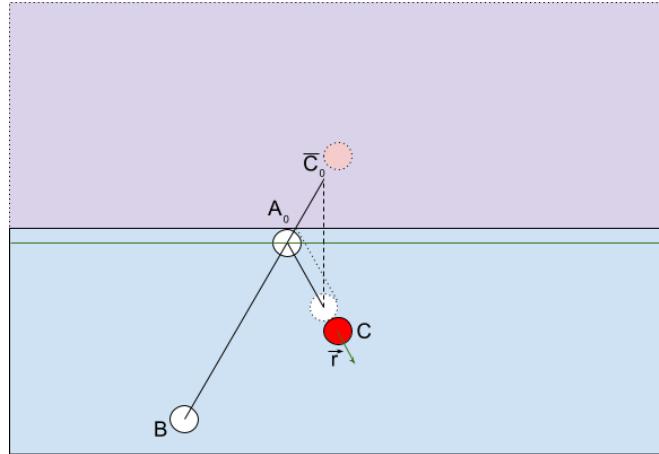


Abbildung 4.12: Berücksichtigung des Kugelradius bei Bandenreflektion

die Kugelposition selbst, sondern ein um den Radius verschobener Punkt in Gegenrichtung zur gewünschten Laufrichtung  $\vec{r}$  der Kugel  $C$  definiert. Andererseits wird die Bande, an welcher gespiegelt wird, um den Radius der Kugel in Richtung des Zentrums verschoben. Die verschobene virtuelle Bande ist grün eingezzeichnet.

Durch Studium mehrerer Beispiele wird ein allgemeiner Algorithmus hergeleitet. Hierbei wird zuvor genanntes Problem mit dem Kugelradius vernachlässigt, da dieses durch die beschriebenen Positionskorrekturen behandelt werden kann. Der Algorithmus kann unabhängig von dieser Korrektur beschrieben werden.

In einem ersten Schritt werden die Banden definiert, über welche der Zielpunkt gespiegelt werden soll. Dies sind in diesem Beispiel die linke und die obere Bande, dementsprechend werden die grün markierten Spiegel zuerst links und dann oben angewendet. Es resultieren die Punkte  $\bar{C}_0$  und  $\bar{C}_1$ . Die Abbildung 4.13 zeigt die Situation mitsamt der gespiegelten Punkte.

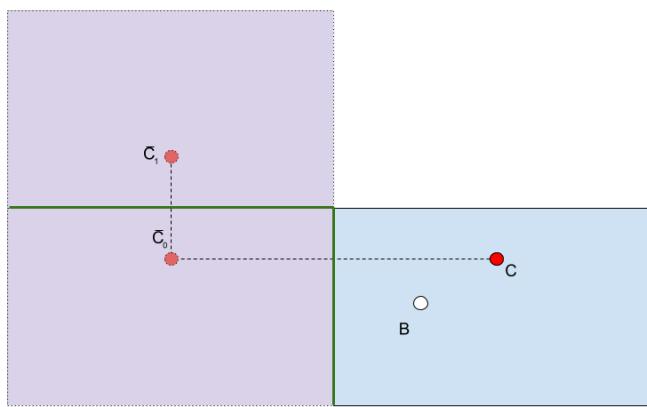


Abbildung 4.13: Zweifaches Bandenspiel - A

Die Abbildung 4.14 zeigt das Vorgehen, Nachdem die Spiegelungen durchgeführt wurden. Es wird der letzte Spiegelpunkt  $\bar{C}_n$ , in diesem Fall  $\bar{C}_1$ , beibehalten, die Vorherigen werden verworfen. Vom Startpunkt  $B$  aus wird eine Verbindung zum Spiegelpunkt  $\bar{C}_1$  gezogen und alle Bandensegmente auf Schnittpunkte geprüft, was schliesslich im Punkt  $A_0$  resultiert. Dies ist der

<sup>1</sup>Für Herleitung, siehe Anhang 8.13

erste Kollisionspunkt mit der Bande und kann der Resultatsmenge hinzugefügt werden. Anschliessend wird der Spiegelpunkt  $\bar{C}_1$  an der Bande gespiegelt, an welcher der Kollisionspunkt liegt, in dem Fall an der Linken. Diese Spiegelung resultiert im Spiegelpunkt  $\bar{C}_0$ .

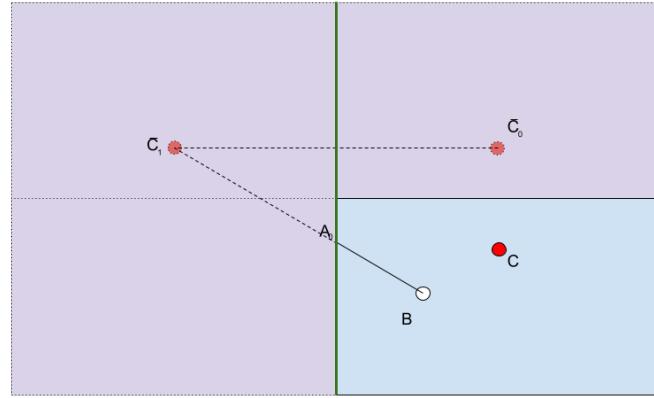


Abbildung 4.14: Zweifaches Bandenspiel - B

In der Abbildung 4.15 wird der darauffolgende Schritt gezeigt. Es gibt zum Vorhergehenden keine wesentlichen Unterschiede. Es wird wiederum nur der letzte Spiegelpunkt  $\bar{C}_0$  beibehalten, zu welchem vom neuen Startpunkt  $A_0$  eine Verbindung gezogen wird. Diese Halbgerade wird wiederum auf Schnittpunkte mit allen Bandensegmenten geprüft, was im Kollisionspunkt  $A_1$  resultiert. Dieser Kollisionspunkt wird der Resultatsmenge hinzugefügt und der Punkt  $\bar{C}_0$  wird an der Bande mit dem Kollisionspunkt gespiegelt, in dem Fall der Oberen. Diese Spiegelung überführt den Punkt  $\bar{C}_0$  auf den ursprünglichen Punkt  $C$ , was das Ende des Algorithmus bedeutet.

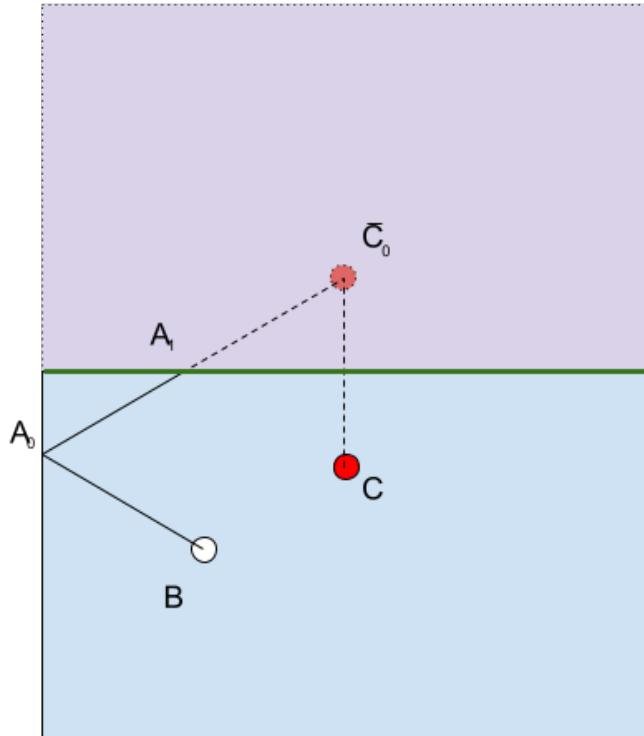


Abbildung 4.15: Zweifaches Bandenspiel - C

Die Abbildung 4.16 zeigt das endgültige Resultat, wenn der letzte Kollisionspunkt  $A_1$  mit dem Zielpunkt  $C$  verbunden wird.

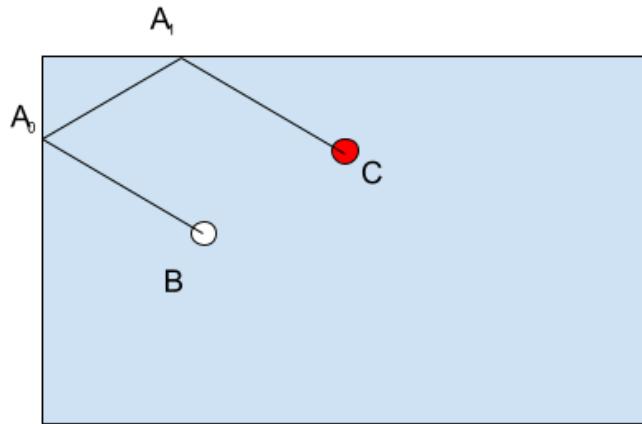


Abbildung 4.16: Zweifaches Bandenspiel - D

Dasselbe Prinzip kann auch auf ein Spiel über drei Banden angewendet werden. Die Abbildung 4.17 veranschaulicht ein Solches. Es wird zuerst an der linken, dann an der oberen und anschliessend an der rechten Bande gespiegelt. Es wird in einem Schritt wiederum der letzte Spiegelpunkt und dessen Startposition betrachtet, mit welchen der Kollisionspunkt an einer Bande gefunden werden kann. Danach kann der Spiegelpunkt an dieser Bande gespiegelt werden und der nächste Schritt kann starten. Dies wird so oft wiederholt, wie es Banden zum spiegeln hat.

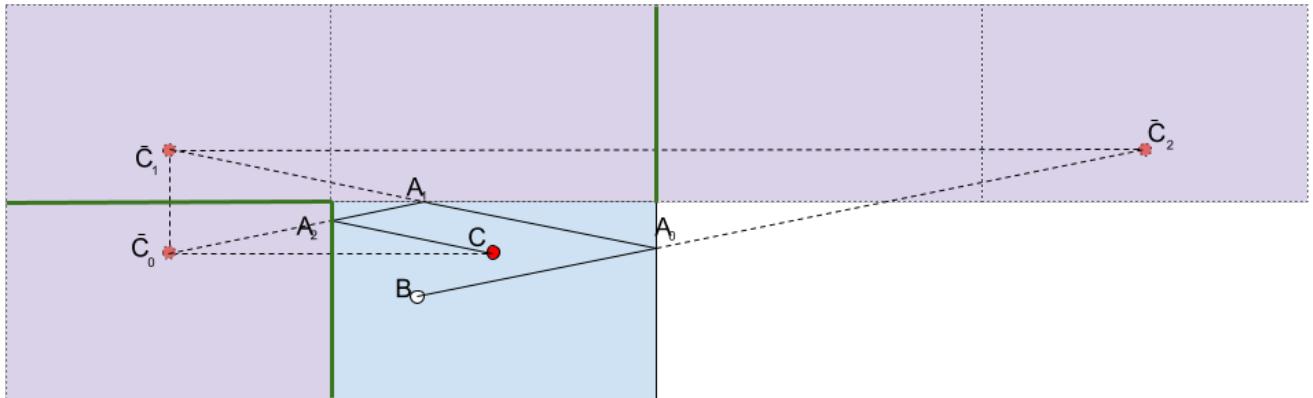


Abbildung 4.17: Dreifache Reflektion an Banden

Eine Spiegelung kann durch die Anwendung der bandenspezifischen Transformationsmatrix  $M$  erzielt werden, wobei  $\vec{s}$  für den Spiegelvektor steht<sup>2</sup>. Hierbei steht der Punkt  $C$  für den Zielpunkt, den es zu treffen gilt.

$$M = \begin{pmatrix} s_x & 0 & R_x^1 - s_x \cdot R_x^1 \\ 0 & s_y & R_y^1 - s_y \cdot R_y^1 \\ 0 & 0 & 1 \end{pmatrix} \quad (4.12)$$

$$\bar{C} = M \cdot C \quad (4.13)$$

<sup>2</sup>Für Herleitung, siehe Kapitel 8.13.1

Weiterhin stellt sich die Frage, an welcher Bande überhaupt gespiegelt werden darf. Dazu wurden Definitionsbereiche für jede Bande erstellt, wie in Abbildung 4.18 ersichtlich ist. Punkte auf der roten Seite liegen ausserhalb des Definitionsbereichs, Punkte im grünen Bereich sind spiegelbar. Des Weiteren ist die Bandennormale vorhanden, welche zum Ursprung des Koordinatensystems in der Tischmitte zeigt. Um für einen Punkt  $C$  herauszufinden, ob dieser spiegelbar ist, muss demnach nur geprüft werden, ob er im Definitionsbereich liegt. Dazu wird der Ursprung des Koordinatensystems zum Start der Bande verschoben und es wird die elementweise Multiplikation des verschobenen Punktes  $C'$  mit der Bandennormale durchgeführt. Die Elemente des Vektors werden geprüft, ob sie grösser oder gleich dem Nullvektor sind. Dies resultiert in einem Vektor, welcher eine 0 speichert, wenn das Element kleiner und 1 wenn es grösser ist. Abschliessend wird die quadrierte Länge des resultierenden Vektors gebildet, diese Länge muss 2 entsprechen, in dem Fall liegt der Punkt im Definitionsbereich.

$$C' = C \cdot T^0 \quad (4.14)$$

$$\bar{C} = C' \cdot \hat{n} \quad (4.15)$$

$$\vec{r} = \bar{C} \geq \vec{0} \quad (4.16)$$

$$I = \vec{r} \cdot \vec{r} \quad (4.17)$$

$$p = I == 2 \quad (4.18)$$

Wird ein Punkt geprüft, der auf derselben Höhe wie eine Bande liegt, muss zusätzlich die jeweilige Komponente des Punktes  $\bar{C}$  grösser als 0 sein, bei der die Komponente des Normalenvektors ungleich 0 ist.

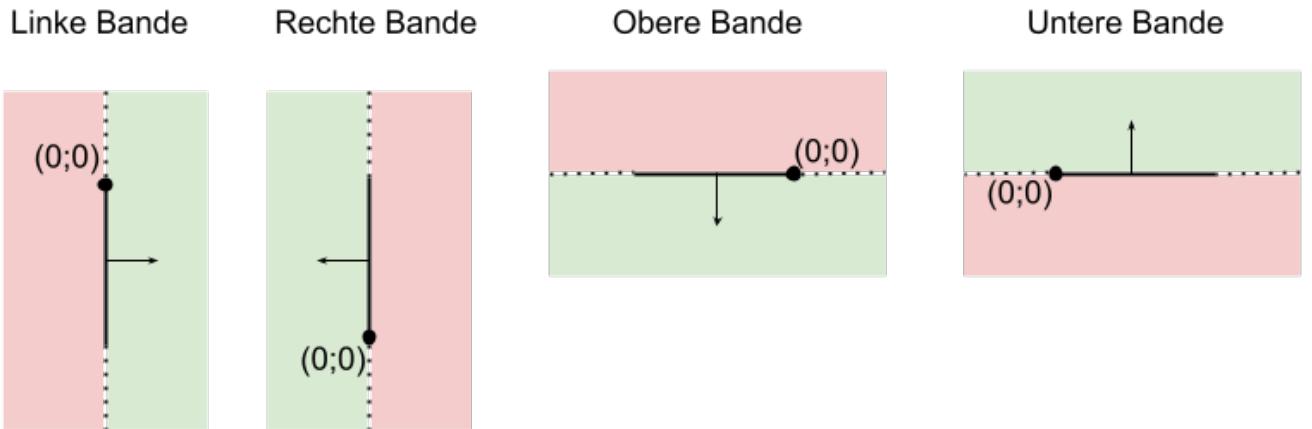


Abbildung 4.18: Definitionsbereich einer Bandenspiegelung

Algorithmus 2 und 3 zeigt, wie ein Stoss über die Bande gefunden werden kann. Dazu werden in einem ersten Schritt alle möglichen Kombinationen der Banden gebildet, was auch den letzten Spiegelpunkt  $\bar{C}_n$  generiert. Danach können die Zielpositionen bestimmt werden, wobei auch geprüft wird, ob der Weg zwischen den Positionen passierbar ist. Sollte dies nicht der Fall sein, wird dieser Lösungskandidat verworfen.

```

Function expandByRail(B: vec2, C: vec2, reflections: int, rails: Rail[]) —> Node[]
    nodes ← Node[]
    combinations ← combos(reflections, C, rails)
    for combination in combinations do
        targets ← target(B, C, combination.second, rails, combination.first)
        if ! empty(targets) then
            | nodes ← append(physicalEvents(B, C, targets), nodes)
        end
    end
    return nodes

Function combos(reflections: int, target: vec2, rails: Rail[]) —> (Rail[], vec2)[]

    combinations: (Rail[], vec2, bool)[] ← []
    for rail in rails do
        | combinations ← append(([rail], reflect(target, rail), canReflect(target, rail)), combinations)
    end

    for index ← 1 < reflections do
        for railIndex ← 0 < length(rails) do
            rail ← rails[railIndex]
            if combinations[railIndex].third and canReflect(combinations[railIndex].second, rail) then
                | combinations[railIndex].first ← append(rail, combinations[railIndex].first)
                | combinations[railIndex].second ← reflect(combinations[railIndex].second, rail)
            end
            else
                | combinations[railIndex].third ← false
            end
        end
    end
    results: (Rail[], vec2)[] ← []
    for railIndex ← 0 < length(rails) do
        if combinations[railIndex].third then
            | results ← append((combinations[railIndex].first, combinations[railIndex].second), results)
        end
    end
    return results

Function canReflect(C: vec2, rail: Rail) —> bool
    moved ← C - rail.start
    reflected ← moved · rail.normal
    possible ← reflected ≥ 0
    return possible · possible = 2

Function reflect(C: vec2, rail: Rail) —> vec2
    reflected: vec3 ← vec3(C, 1.0)
    return vec2(rail.M · reflected)

```

**Algorithm 2:** Algorithmus zur Berechnung eines Stosses über die Bande - Teil 1

```

Function target(B: vec2, C: vec2, reflected: vec2, rails: Rail[], combo: Rail[]) —> vec2[]
  targets <- vec2[]
  start <- B
  while ! empty(combo) do
    comboRail <- pop(combo)
    target <- ( $\infty, \infty$ )
    for rail in rails do
      railIntersection <- halfLineLineSegmentIntersection(start, (reflected - start), rail.start, rail.end)
      if railIntersection and start <> railIntersection then
        target <- railIntersection
        reflected <- reflect(reflected, rail)
        break
      end
    end
    if ! shotPossible(start, target) then
      return
    end
    targets <- append(target, targets)
    start <- target
  end
  If! shotPossible(start, C) return targets

```

**Algorithm 3:** Algoritmus zur Berechnung eines Stosses über die Bande - Teil 2

Für die Expansion einer Kugel, sei es über eine weitere Kugel oder über eine Bande, muss geprüft werden, dass keine weitere Kugel im Weg ist. Dies wird über die Berechnung des Distanzvektors zwischen dem Vektor  $\vec{d}$  und der zu prüfenden Kugel erzielt. Dasselbe Prinzip wird in Kapitel 4.6.4.5.1 beim Fall einer dynamisch-statischen Kugelkollision erläutert.

Sobald die weisse Kugel expandiert wird, muss sichergestellt werden, dass diese auch an der entsprechenden Position mit dem Queue getroffen werden kann. Dies wird, wie in Abbildung 4.19 veranschaulicht, durch einen Vektor  $\vec{s}$  mit einer bestimmten Länge  $s$  entgegen der Rollrichtung der Kugel sowie einem Mindestabstand  $e$  zu diesem Vektor sichergestellt. Die Länge von  $e$  wird auf 10[mm] gewählt. Von jeder Kugel aus werden die Abstände gemessen, liegt die Kugel zwischen  $W$  und  $W + \vec{s}$ , so wird der Abstand senkrecht zum Vektor  $\vec{s}$  berechnet, zu sehen bei der Kugel  $K_1$ . Liegt die Kugel wie  $K_2$  vorne oder hinten, wird der Abstand zum Punkt  $W$  oder  $W + \vec{s}$  berechnet. Für die so erhaltenen Distanzen  $d_i$  muss gelten, wobei  $r$  für den Kugelradius steht:

$$d_i \geq e + r \quad (4.19)$$

Liegt die Kugel in Richtung der Rollrichtung, wurde die Bedingung bereits geprüft, ansonsten dürfte die weisse Kugel nicht expandiert worden sein. Daher muss dieser Fall nicht speziell behandelt werden.

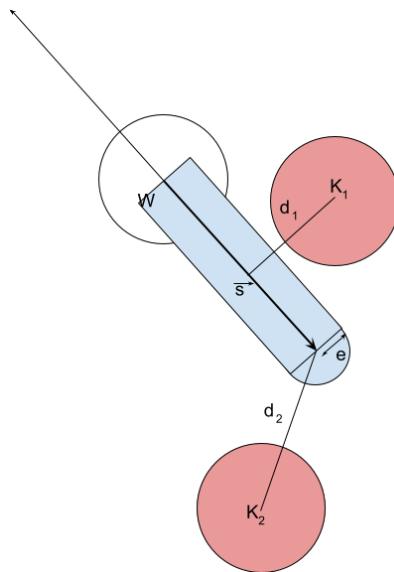


Abbildung 4.19: Kugelexpansion - Platz für Queue

#### 4.6.1.2 Bewertungsfunktion

Um die Suche zu vereinfachen und in eine spezifische Richtung zu lenken, wo die besten Resultate zu erwarten sind, ist es unerlässlich eine Bewertung des Stosses durchzuführen. Die Heuristik wurde so definiert, dass sie sich auf den aktuell expandierten Knoten beschränkt. Die Kosten für eine Expansion werden über die Pfade aufsummiert. Anhand der Summen kann jeweils der kostengünstigste Pfad evaluiert und verfolgt werden. Das Prinzip wird in Abbildung 4.20 veranschaulicht. Die Knoten werden je nach Bedeutung mit unterschiedlichen Farben markiert. Blau sind sie, wenn sie bereits expandiert wurden. Grün, wenn der Knoten im nächsten Schritt expandiert wird, da er die geringsten Pfadkosten aufweist und rot, wenn der Knoten nicht in Frage kommt aufgrund zu hoher Kosten. Dieses Vorgehen entspricht demjenigen des Dijkstra-Algorithmus [Unk21c].

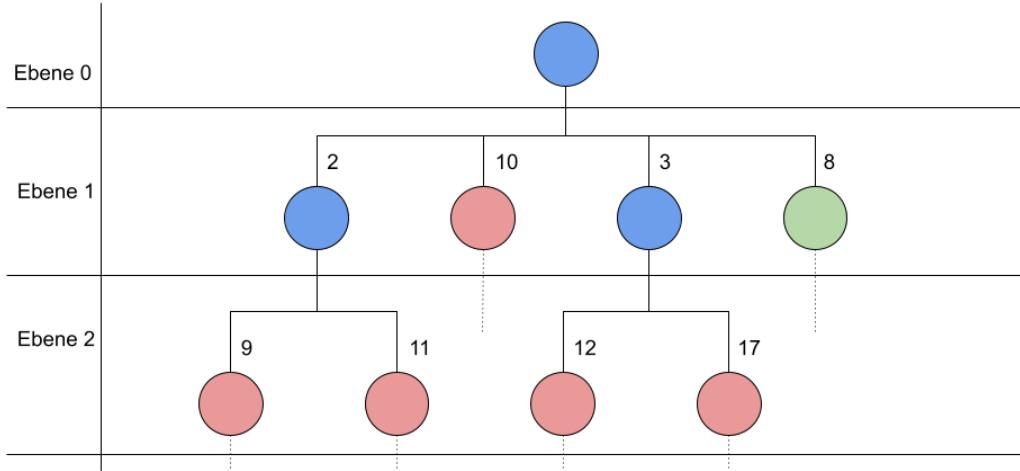


Abbildung 4.20: Bewertung eines Suchbaums

Die Kosten werden auf Basis dreier Kriterien<sup>3</sup> gebildet. Das Erste bildet die Berücksichtigung der Distanz, welche eine Kugel zurücklegt. Das Zweite behandelt den Winkel, in welcher ein Zielpunkt getroffen werden muss. Das Dritte fügt den Kosten einen Wert für jede Indirektion hinzu. Jeder dieser Werte liegt zwischen 0 und 1. Die ersten beiden Kriterien sind in Abbildung 4.21 veranschaulicht. Es werden zwei Expansionen gezeigt. Die relevanten Informationen  $d$  für die Distanz sowie  $\alpha$  für den Winkel weisen einen entsprechenden Index auf, welcher den Expansionsschritt markiert. Beim ersten Expansionsschritt bildet der Zielpunkt den Elternknoten. Der Winkel  $\alpha_1$  ist definiert durch die Rollrichtung der Kugel und einer Normalen auf den Zielpunkt. Die Normalen zeigen jeweils zum Ursprung in der Mitte des Tisches. Die Distanz  $d_1$  ist definiert über die Länge des zurückzulegenden Weges. Ist in einem Expansionsschritt die weiße Kugel betroffen, werden die Distanzkosten mit den Distanzkosten des vorangegangenen Expansionsschritts gewichtet. Dies stellt sicher, dass der Weg, welcher die nächste Kugel nehmen wird, möglichst kurz sein muss, damit sich ein allfälliger Fehler beim Stoss der weißen Kugel nicht zu stark akkumuliert. Im zweiten Fall wird der Winkel  $\alpha_2$  durch die Rollrichtung der ersten und der Rollrichtung der zweiten Kugel definiert.

Um die beiden Größen vergleichen zu können, müssen sie in dieselbe Größenordnung gebracht werden. Aus diesem Grund werden sie durch die maximal möglichen Werte normiert[Mül21]. Für die Distanz ist dies die Diagonale über den Tisch, für den Winkel wird ein maximal möglicher Wert von  $87^\circ$  gewählt, dies ist bereits bei der Suche berücksichtigt. Es gilt die Annahme, dass je kürzer der Weg und je kleiner der Winkel, desto einfacher der Stoss.

$$d_{krit} = \frac{d_i}{d_{max}} \quad (4.20)$$

$$\alpha_{krit} = \frac{\alpha}{\alpha_{max}} \quad (4.21)$$

Aktuell fließt der Winkel  $\alpha_{krit}$  zu stark in die Bewertung ein. Daher wird dieser durch eine kubische Bézier-Kurve[Unk21b] gewichtet. Die Parameter lauten wie folgt.

$$P_0 = (0 \quad 0) \quad (4.22)$$

$$P_1 = (1 \quad 0) \quad (4.23)$$

$$P_2 = (0.5 \quad 1) \quad (4.24)$$

<sup>3</sup>Die Kriterien betreffend der Distanz wie auch des Winkels werden wie in Publikation [Smi06] verwendet.

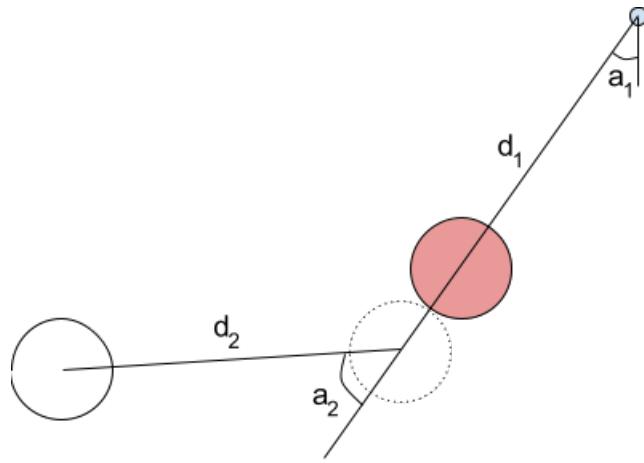


Abbildung 4.21: Expansionskosten eines Knotens

$$P_3 = \begin{pmatrix} 1 & 1 \end{pmatrix} \quad (4.25)$$

$$P = \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix} \quad (4.26)$$

$$T = \begin{pmatrix} t^3 & t^2 & t & 1 \end{pmatrix} \quad (4.27)$$

$$M = \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \quad (4.28)$$

$$\alpha_{krit}^G = f(t = \alpha_{krit}) = T \cdot M \cdot P \quad (4.29)$$

Nach dieser Gewichtung resultiert ein Punkt im zweidimensionalen Raum, wovon die y-Komponente als  $\alpha_{krit,y}^G$  verwendet wird. Die Kurve ist in Abbildung 4.22 visualisiert. Es wird deutlich, dass kleinere Winkel einen eher geringen Einfluss auf die Kosten haben. Ab einem Winkel von  $50^\circ$ , welcher auf der Grafik 4.22 als Punkt  $D_1$  markiert ist, beginnt die Kurve bis etwa  $80^\circ$  stark zu steigen. Ab dort flacht sie wiederum ab, bis die Kosten von  $90^\circ$  schliesslich den Wert 1 erreichen.

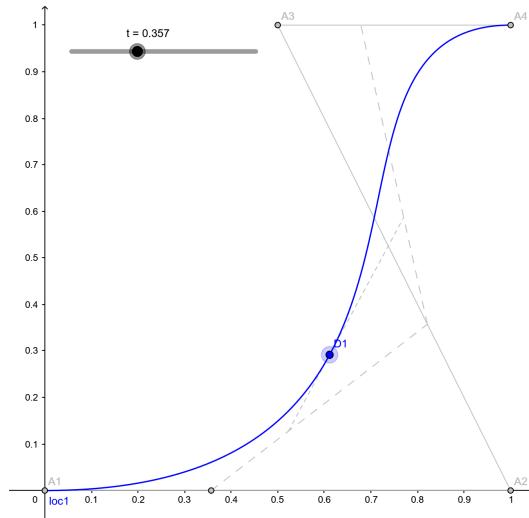


Abbildung 4.22: Gewichtung der Winkelkosten

Das Kriterium der Indirektion über Kugeln wird wiederum über einen maximal möglichen Wert gelöst. Es wird eine maximale Indirektion  $K_{I,max}$  angegeben und die Anzahl der Vorkommnisse  $K_{I,n}$  wird durch die Konstante dividiert. Dadurch werden die

Kosten erhöht, je grösser der Indirektionsgrad ist.

$$K_{I,krit} = \frac{K_{I,n}}{K_{I,max}} \quad (4.30)$$

Das Kriterium der Indirektion über Banden wird ebenso über einen maximal möglichen Wert gelöst. Es wird eine maximale Indirektion  $K_{B,max}$  angegeben und die Anzahl der Vorkomnisse  $K_{B,n}$  wird durch die Konstante dividiert. Dadurch werden die Kosten erhöht, je grösser der Indirektionsgrad ist.

$$K_{B,krit} = \frac{K_{B,n}}{K_{B,max}} \quad (4.31)$$

Die endgültigen Kosten werden über die Addition aller Kriterien gebildet.

$$K = d_{krit} + \alpha_{krit,y}^G + K_{I,krit} + K_{B,krit} \quad (4.32)$$

## 4.6.2 Berechnung der Initialgeschwindigkeit

Sofern ein Stosskandidate wie in Abschnitt 4.6.1 beschrieben gefunden wurde, muss berechnet werden, welche Geschwindigkeit die weisse Kugel erhalten muss, damit die gewünschte Kugel ins Loch gespielt wird. Relevant dafür ist der Reibungsverlust wenn die Kugel rollt und die Kollision mit anderen Kugeln.

Wenn eine Kugel in ein Loch rollen soll, kann eine minimale Endgeschwindigkeit angenommen werden und aufgrund der zurückzulegenden Strecke die Startgeschwindigkeit bestimmt werden, dies wird in Abschnitt 4.6.2.1 beschrieben. Sofern diese Kugel über den Spielball ins Loch gespielt werden soll, dann muss der Spielball mit einer gewissen Geschwindigkeit die Kugel treffen, damit diese die gewünschte Startgeschwindigkeit erhält. Dies wird in Abschnitt 4.6.2.2 beschrieben.

### 4.6.2.1 Reibungsverlust über Bahn

Das Ziel eines Billiardstosses ist es, eine Kugel von einem Punkt A zu einem Punkt B rollen zu lassen. Dabei findet Rollreibung statt, welche das Abbremsen der Kugel verursacht. Es ist relevant zu wissen, welche Startgeschwindigkeit  $v_1$  eine Kugel am Punkt A haben muss, um mit der Endgeschwindigkeit  $v_2$  am Punkt B anzukommen. Beispielsweise wenn eine Kugel X in eines der Löcher gespielt werden soll, wobei die Kugel eine gewisse minimale Endgeschwindigkeit haben soll, damit sie tatsächlich ins Loch rollt.

Es kann die nachfolgende Formel angewendet werden<sup>4</sup>:

$$\vec{v}_1 = \sqrt{\frac{49 \cdot \mu_g \cdot (|\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot \Delta s)}{49 \cdot \mu_g + 24 \cdot (\mu_r - \mu_g)}} \cdot \frac{\vec{v}_2}{|\vec{v}_2|} \quad (4.33)$$

Sei  $S$  die Startposition der Kugel,  $E$  die Endposition der Kugel und  $V$  der gewünschte Betrag der Endgeschwindigkeit, dann lässt sich die Endgeschwindigkeit  $\vec{v}_2$  wie folgt berechnen:

$$\vec{\Delta s} = E - S \quad (4.34)$$

$$\hat{\Delta s} = \frac{\vec{\Delta s}}{|\vec{\Delta s}|} \quad (4.35)$$

$$\vec{v}_2 = V \cdot \hat{\Delta s} \quad (4.36)$$

### 4.6.2.2 Elastischer Stoss bei Kugelkollision

Um eine Kugel T in eine gewünschte Richtung, wie etwa zum Loch, zu rollen, muss sie von einer anderen Kugel A, evtl. vom Spielball, angestoßen werden. Es gilt herauszufinden, wo die Kugel T getroffen werden muss und welche Geschwindigkeit die Kugel A zum Kollisionszeitpunkt haben muss, um die Kugel T in die gewünschte Richtung mit der geforderten Geschwindigkeit rollen zu lassen.

Die Situation mit der Kugel T, an der Position  $C$  und der Kugel A, an der Position  $A$  ist in Abbildung 4.23 dargestellt. Die Kugel A muss zum Punkt  $B$  rollen, wo sie auf Kugel T prallt und ein elastischer Stoss[Unk21f] stattfindet. Während die Kugel die

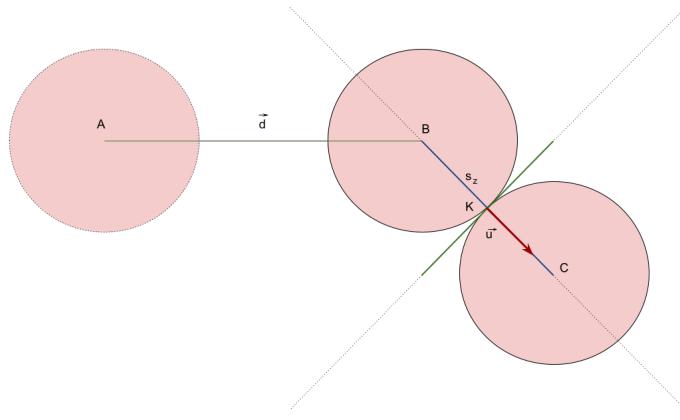


Abbildung 4.23: Kollisionspunkt zweier Kugeln

Distanz  $|\vec{d}|$  zurücklegt, verliert sie an Geschwindigkeit aufgrund von Reibung, diese wird in Abschnitt 4.6.2.1 behandelt. Hier ist lediglich relevant, welche Geschwindigkeit die Kugel A am Punkt  $B$  haben muss.

Sei  $\vec{u}$ , die gewünschte Richtung und Geschwindigkeit der Kugel T nach der Kollision, dann ist der Zielpunkt  $B$  der Kugel A wie folgt definiert<sup>5</sup>:

$$B = C - 2 \cdot r \cdot \hat{u} \quad (4.37)$$

Sei  $\hat{d}$  der Richtungsvektor der Länge 1 zwischen den Punkten  $A$  und  $B$ ,  $E_v$  die Energieverlustkonstante in Prozent, dann gilt für die Geschwindigkeit  $\vec{v}_1$  der Kugel A bei der Kollision<sup>6</sup>:

$$\hat{u} = \frac{\vec{u}}{|\vec{u}|} \quad (4.38)$$

$$\vec{v}_1 = \frac{\vec{u} \cdot \hat{d}}{\hat{d} \cdot \vec{u}} \cdot \frac{1}{1 - E_v} \cdot \hat{d} \quad (4.39)$$

Anschliessend kann die Startgeschwindigkeit der Kugel A an der Position A berechnet werden, weil die gewünschte Endgeschwindigkeit und der zurückzulegende Weg bekannt ist<sup>7</sup>.

#### 4.6.2.3 Bandenkollision

Hierbei wird der Geschwindigkeitsvektor berechnet, der vor einer Bandenkollision gilt. Es wird von einer perfekten Spiegelung ausgegangen. Der an der Bande eingehende Geschwindigkeitsvektor  $\vec{v}_0$  kann demnach berechnet werden, wenn die Bandennormale  $\vec{N}$  sowie die gewünschte Engeschwindigkeit  $\vec{v}$  bekannt ist.

$$\vec{v}_0 = \vec{v} - 2 \cdot \vec{N} \cdot (\vec{v} \cdot \vec{N}) \quad (4.40)$$

Der einkommende Geschwindigkeitsvektor  $\vec{v}_0$  muss noch um den Energieverlust an der Bande vergrössert werden:

$$\vec{v}_0 = |\vec{v}_0 \cdot \frac{1}{1 - E_v}| \cdot \hat{v}_0 \quad (4.41)$$

<sup>4</sup>Die Herleitung findet sich im Anhang 8.1

<sup>5</sup>Die Herleitung findet sich im Anhang 8.3

<sup>6</sup>Die Herleitung findet sich im Anhang 8.3

<sup>7</sup>siehe Kapitel 4.6.2.1

#### 4.6.3 Modellierung physikalisches System

Ein Stoss kann für die Simulation mithilfe eines Modells beschrieben werden. Dieses Modell beschreibt den Ablauf eines Stosses durch Ereignisse, welche den beteiligten Objekten widerfahren. Die Kugeln, Banden und Löcher sind die Objekte dieses Modells. Jedes Objekt dieses Modells hat einen veränderlichen Energiewert, bspw. sind Kugeln in Bewegung oder sie sind ruhend. Eine Bande oder ein Loch bewegen sich nie, d.h. deren Energiewert ist konstant 0. Kollisionen zweier Kugeln oder einer Kugel und der Bande sind Beispiele von Ereignissen, welche Veränderungen der Energiezustände der Objekte bewirken.

Objekte, die einen Energiewert besitzen, welcher sich über die Zeit oder durch Interaktionen mit anderen Objekten verändert, werden nachfolgend variabel genannt. Sofern ein variables Objekt einen Energiewert grösser 0 hat, dann wird es weiter als dynamisch bezeichnet. Sobald es den Energiewert 0 erreicht, ist es als statisches variables Objekt kategorisiert. Banden und Löcher, welche einen fixen Energiewert besitzen, werden als konstant beschrieben.

Zwischen den Ereignissen kann eine Energieabnahme stattfinden. Beispielsweise unterliegt eine rollende Kugel der Rollreibung und verliert dadurch bis zum nächsten Ereignis Energie. Diese Energieabnahme erfolgt durch die Anwendung der Kantenfunktion auf der Kante zwischen zwei Ereignissen.

Diese Nomenklatur ist in Abbildung 4.24 ersichtlich.

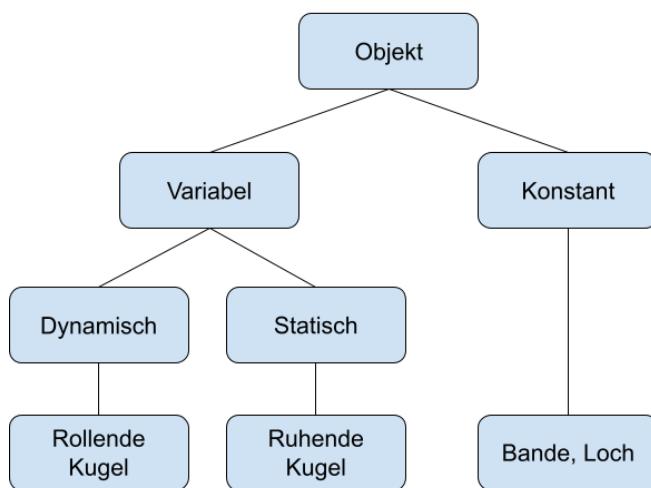


Abbildung 4.24: Typen von Objekten

#### 4.6.3.1 Ereignisse und ihre Repräsentation

Das Ziel ist der Aufbau eines graphenähnlichen Konstrukts, welches aus Layern besteht und Zustandsübergänge variabler Objekte durch Knoten repräsentiert. Einige dieser Knoten treten bei Ereignissen wie einer Kollision oder wenn eine Kugel in ein Loch rollt auf. Andere Knoten dienen der reinen Abbildung des variablen Objekts innerhalb des Layers. Nachfolgend werden die Knoten definiert.



**Energy-Input-Node:** Dieser Node beschreibt das Auftreten eines Energie-Inputs von Aussen. Beispielsweise wenn die weisse Kugel mit dem Queue gestossen wird.



**Energy-Transfer-Node (Collision-Node):** Dieser Node beschreibt die Übergabe von Energie auf die beteiligten Objekte. Ein Beispiel ist die Kollision zweier Kugeln, wobei diese Energie austauschen.

Der Energy-Transfer-Node wird in drei Schichten aufgeteilt. Bei der Input-Schicht wird die Kantenfunktion (siehe S. 31) angewendet. Diese berücksichtigt den Energieverlust bis zum Auftreten des Ereignisses. Die mittlere Schicht beschreibt die Übertragefunktion der beteiligten Objekte. Die dritte Schicht repräsentiert das Resultat, also den Status der Objekte nach der Energieübergabe.

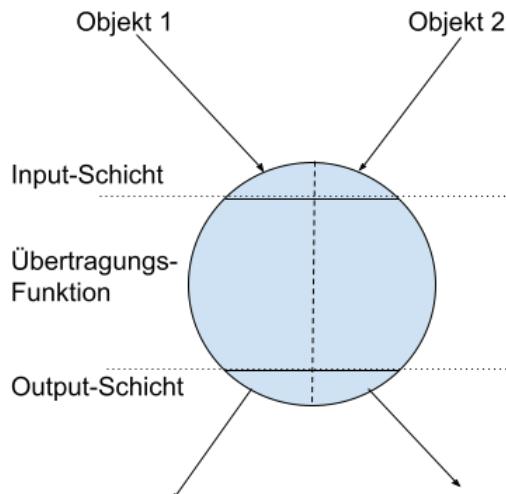


Abbildung 4.25: Der Energy-Transfer-Node



**No-Energy-Node:** Dieser Node wird eingesetzt, sobald der Energiewert eines variablen Objekts auf 0 sinkt und somit auch den Übergang von dynamisch zu statisch repräsentiert.



**Out-Of-System-Node:** Dieser Node wird eingesetzt, sobald ein variables Objekt das System verlässt, indem beispielsweise eine Kugel in ein Loch rollt.



**Cutting-Node:** Dieser Node ist ein spezieller Energy-Transfer-Node. Er wird bei variablen Objekten eingesetzt, die nicht an einem Ereignis beteiligt sind, wenn ein solches auftritt. Zum Ereigniszeitpunkt wird ein neuer Layer (siehe S. 31) geschaffen, welcher die Objekte, die am Ereignis beteiligt sind, in einem Energy-Transfer-Node festhält. Weiterhin wird der Status aller dynamischen Objekte ebenfalls zu diesem Zeitpunkt festgehalten. Der Cutting-Node beschränkt sich auf ein Input- sowie Output-Objekt und beinhaltet als Energieübertragungsfunktion die Identitätsfunktion.

#### 4.6.3.2 Kantenfunktion

Die Kantenfunktion beschreibt eine Energieabnahme über die Zeit oder den Weg.

#### 4.6.3.3 Layer

Ein Layer beinhaltet sämtliche Veränderungen der variablen Objekte und beschreibt den Status aller Objekte zu diesem Zeitpunkt. Sobald ein Ereignis auftritt wird ein neuer Layer im Modell eingefügt.

#### 4.6.3.4 Beispiel eines Graphen

Das beschriebene Datenmodell kann als Graph<sup>8</sup> visualisiert werden. Als Beispiel wird die Idee eines Billardstosses in Abbildung 4.26 hinzugezogen. Auf dem Tisch liegt eine weisse wie auch zwei rote Kugeln. Die zweite rote Kugel wird an keiner Interaktion beteiligt sein, weswegen sie ihren Zustand nie verändert. Es ist ersichtlich, dass bei jedem Ereignis für diese Kugel ein „Out-of-energy-Node“ eingefügt wird. Das erste Event beschreibt die Kollision des dynamischen Objekts „weisse Kugel“ sowie des statischen Objekts „rote Kugel“. Danach verliert die weisse Kugel sämtliche Energie und wechselt in den Status „Out-of-energy“. Da die rote Kugel zu diesem Zeitpunkt dynamisch ist, wird für sie ein „Cutting-Node“ eingefügt. Zuletzt rollt die rote Kugel in das Loch, für sie wird ein „Out-of-system-Node“ erstellt und das System hat sämtliche Energie verloren. Ein Endzustand wurde erreicht, da alle Kugeln nun statisch sind.

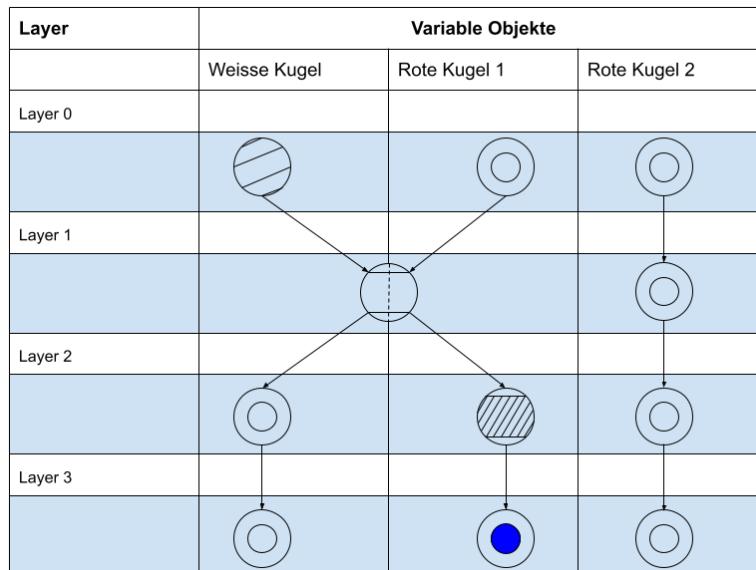


Abbildung 4.26: Beispiel für ein Resultat des Algorithmus 4

In Algorithmus 4 wird die Grundidee erläutert. Als Input für die Funktion „simulate“ dient der erste Layer, welcher mehrere variable Objekte beinhalten kann, wobei mindestens ein Objekt dynamisch (energiereich) sein sollte. Dieser Layer wird direkt dem erzeugten System hinzugefügt und dieses wird solange bearbeitet, bis alle variablen Objekte statisch sind (das System hat keine Energie mehr). In jedem Schleifendurchlauf wird das nächste Event berechnet. Die Events können diverser Natur sein. Es werden Kollisionen mit dynamischen, statischen wie auch konstanten Objekten geprüft. Bei der Kollision mit konstanten Objekten können die Ereignisse „Energy transfer“ oder „Out of System“ auftreten. Weiterhin wird geprüft, ob ein dynamisches Objekt seine Energie durch die Kantenfunktion verliert. Aufgrund eines Ereignisses wird zunächst ein neuer Layer generiert, dann für die am Ereignis beteiligten Objekte den entsprechenden Node und zuletzt für die anderen variablen Objekte entweder

<sup>8</sup>Aus effizienzgründen wird auf einen Graphen in der Implementation verzichtet. Das Modell setzt sich aus verschiedenen Layern zusammen, bei welchen die Informationen der Events für jedes variable Objekt separat und mit Zugriffszeit  $O(1)$  abgelegt werden.

ein „Cutting-Node“ oder ein „Out-of-energy-Node“ eingefügt.

```

struct {
    timestamp: datetime
    me: string
    partner: string
    partnerType: [BALL, TARGET, RAIL]
    event: [OutOfEnergy, Collision]
} Event

Function simulate(start: Layer, constantObjects: list) —> System
    system ← System()
    system ← appendLayer(system, start)
    while ! system.isStatic() do
        nextEvent ← nextEvent(system.lastLayer(), constantObjects)
        layer ← atMoment(system.lastLayer(), nextEvent)
        system ← appendLayer(system, start)
    end
    return system

Function nextEvent(layer: Layer, constantObjects: list) —> Event
    nextEvent: Event ← none
    for object in layer.dynamicObjects() do
        nextEvent ← min(nextEvent, outOfEnergy(object))
        nextEvent ← min(nextEvent, collision(object, layer.dynamicObjects()))
        nextEvent ← min(nextEvent, collision(object, layer.staticObjects()))
        nextEvent ← min(nextEvent, collision(object, constantObjects))
        // ... more events to check (eq. rolling)
    end
    return nextEvent

Function atMoment(layer: Layer, event: Event) —> Event
    nextLayer ← layer() for node in layer.objects() do
        if node.first == event.me or node.first == event.partner then
            | nextLayer ← append(nodeForEvent(node, event), nextLayer)
        end
        else
            if node.second.hasEnergy() then
                | nextLayer ← append(cuttingNode(node, event.timestamp), nextLayer)
            end
            else
                | nextLayer ← append(node, nextLayer)
            end
        end
    end
    return nextLayer

```

**Algorithm 4:** Algorithmus zum Aufbau eines physikalischen Systems

#### 4.6.4 Simulation

Sobald ein möglicher Lösungskandidat anhand der in Abschnitt 4.6.1 beschriebenen Suche gefunden und dessen Initialgeschwindigkeit nach Abschnitt 4.6.2 berechnet wurde, wird eine Simulation durchgeführt, um die Lösung definitiv zu bestätigen. Durch das Anwenden verschiedener Anfangsgeschwindigkeiten der weissen Kugel können in diesem Schritt mehrere Situationen evaluiert werden.

Die Simulation wird durch die Definition eines physikalischen Systems wie in Kapitel 4.6.3 durchgeführt. Hierbei gelten die Zuordnungen wie sie nachfolgend beschrieben werden.

##### Ereignisse

**Energy-Input-Node** Wird modelliert über die Eingabe der Energie der weissen Kugel. Ein spezifischer Node zur Modellierung wird nicht implementiert, es wird der Energy-Transfer-Node verwendet, wobei nur der Output-Wert relevant ist.

**Energy-Transfer-Node** Tritt bei der Kollision zwischen zweier Kugeln oder einer Kugel mit der Bande auf.

**No-Energy-Node** Tritt auf, wenn eine Kugel vom dynamischen in den statischen Zustand wechselt (ausrollt). In jedem Layer, wo eine Kugel statisch ist, wird sie durch diesen Node modelliert.

**Out-of-System-Node** Sobald eine Kugel mit dem Zielkreis kollidiert, tritt dieses Ereignis auf. Dem System wird die Energie entzogen und die Kugel ist nicht mehr verfügbar.

**Kantenfunktion** Die Kantenfunktion zwischen den Übergängen innerhalb des Layers bildet der Reibungsverlust der Kugel über eine bestimmte Zeit oder einen bestimmten Ort.

**Dynamische/Statische Objekte** Im Billiard gibt es nur die Kugeln als statische und/oder dynamische Objekte.

**Konstante Objekte** Die konstanten Objekte bilden die Banden wie auch die Ziele.

Es wird ungefähr der Pseudoalgorithmus wie in 4 angewendet, optimal auf das Problem „Billiard“ abgestimmt. Es folgen die physikalischen Berechnungen zur Durchführung der Simulation.

#### 4.6.4.1 Reibungsverlust über die Zeit

Die Geschwindigkeit einer Kugel wird durch die Reibung über die Zeit reduziert. Dazu wird die Formel der gleichförmig beschleunigten Bewegung verwendet, wobei  $\vec{v}_0$ ,  $\vec{a}$  sowie  $t$  gegeben sind.

$$\vec{v} = \vec{a} \cdot t + \vec{v}_0 \quad (4.42)$$

#### 4.6.4.2 Elastischer Stoss zweier Kugeln

Die Kollision zweier Kugeln wird im folgenden als zweidimensionaler elastischer Stoss angesehen. Dabei ist es wichtig, zwei Komponenten beim Zusammenprall zu betrachten. Dies ist einerseits das Liniensegment  $s_z$  zwischen den Mittelpunkten der Kugeln sowie die orthogonal dazu stehende Gerade  $s_t$ . Die Kugel, welche Energie mitbringt, übergibt der anderen Kugel Energie in Richtung von  $s_z$ . Die übrig gebliebene Energie zeigt in Richtung von  $s_t$  [Unk21f]. In Abbildung 4.27 ist ein solcher Stoss dargestellt. Kugel eins bringt dabei die Geschwindigkeit  $\vec{v}_1$  und Kugel zwei die Geschwindigkeit  $\vec{v}_2$  mit. Ein Anteil der Geschwindigkeit von Kugel eins wird der Kugel zwei in Richtung von  $s_z$  übergeben. Dasselbe gilt für die Kugel zwei. Sie übergibt einen Teil ihrer Geschwindigkeit an Kugel eins ebenfalls in Richtung von  $s_z$ . Die verbleibende Energie der Kugeln zeigt jeweils in Richtung von  $s_t$ .

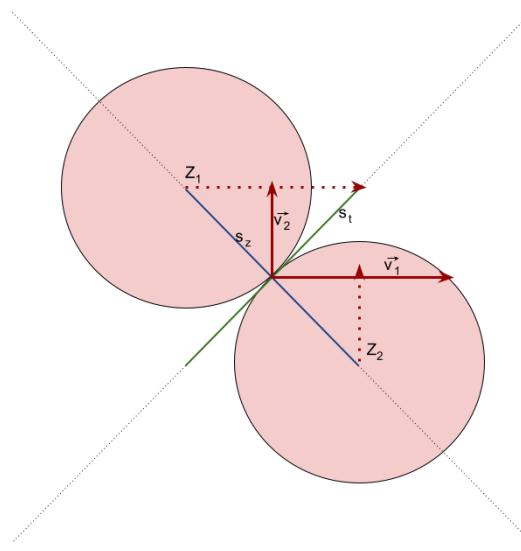


Abbildung 4.27: Elastischer Stoss zweier Kugeln

Die initialen Geschwindigkeiten  $\vec{v}_1^i$  und  $\vec{v}_2^i$  werden aufgrund eines Energieverlusts um eine Konstante  $E_v$ , welche in Prozent angegeben wird, reduziert. Dies ist notwendig, da in der Realität durch Reibung die Energie nicht vollständig weitergegeben wird.

$$\vec{v}_1 = \vec{v}_1^i \cdot (1 - E_v) \quad (4.43)$$

$$\vec{v}_2 = \vec{v}_2^j \cdot (1 - E_v) \quad (4.44)$$

$$(4.45)$$

Um nun die neuen Geschwindigkeiten  $\vec{u}_1$  und  $\vec{u}_2$  zu berechnen, müssen die initialen Geschwindigkeiten  $\vec{v}_1$  sowie  $\vec{v}_2$  auf die Komponenten in Richtung von  $s_z$  und  $s_t$  aufgeteilt werden.

$$\vec{z} = \vec{Z}_2 - \vec{Z}_1 \quad (4.46)$$

$$\hat{z} = \frac{\vec{z}}{|\vec{z}|} \quad (4.47)$$

Von Interesse ist dabei nur  $\hat{z}$ . Mittels diesem Vektor kann die Komponente in Richtung von  $s_z$  beider Geschwindigkeiten ausgerechnet werden.

$$v_{1,z} = (\vec{v}_1 \cdot \hat{z}) \cdot \hat{z} \quad (4.48)$$

$$v_{2,z} = (\vec{v}_2 \cdot \hat{z}) \cdot \hat{z} \quad (4.49)$$

Mittels dieses neuen Vektors in  $s_z$  Richtung kann der Vektor in  $s_t$  Richtung berechnet werden.

$$\vec{v} = \vec{v}_t + \vec{v}_z \quad (4.50)$$

$$\vec{v}_t = \vec{v} - \vec{v}_z \quad (4.51)$$

Daraus folgt für die jeweiligen Kugeln:

$$v_{1,t} = \vec{v}_1 - v_{1,z} \quad (4.52)$$

$$v_{2,t} = \vec{v}_2 - v_{2,z} \quad (4.53)$$

Die neuen Geschwindigkeitsvektoren setzen sich wie beschrieben aus den jeweiligen Komponenten zusammen. Das Resultat lautet wie folgt:

$$\vec{u}_1 = \vec{v}_{2,z} + \vec{v}_{1,t} \quad (4.54)$$

$$\vec{u}_2 = \vec{v}_{1,z} + \vec{v}_{2,t} \quad (4.55)$$

#### 4.6.4.2.1 Behandlung Topspin

Da eine Kugel auch einen Topspin aufweist und kaum Reibung mit der kollidierenden Kugel hat, hat dieser Topspin im Falle einer Kugelkollision einen Einfluss auf den weiteren Verlauf der vorher dynamischen Kugeln. Dieser Einfluss scheint konstant oder zumindest in kleinem Masse variabel, also praktisch unabhängig von der eingehenden Geschwindigkeit zu sein und fällt demnach vor allem bei kleinen Geschwindigkeiten ins Gewicht. Es werden zwei Ansätze erläutert. Die erste Variante behandelt den Topspin als ablenkende Kraft in eingehender Richtung mit konstanterem Betrag. Die zweite Variante berücksichtigt zusätzlich die Variabilität des Topspins. In Abbildung 4.28 wird der Einfluss des Topspins bei grossen wie auch kleinen Geschwindigkeiten erläutert.

Bei Variante eins wird der Einfluss des Topspins berücksichtigt, indem bei den vorher dynamischen Kugeln ein Vektor mit konstanterem Betrag in der einkommenden Geschwindigkeitsrichtung auf den durch die elastische Kollision berechneten Vektor  $\vec{v}_{k,t}$  aufaddiert wird. Demnach gilt für den neuen Vektor  $\vec{v}_{k,t,n}$  der Kugeln, wobei  $A$  für die konstante Geschwindigkeit und  $\hat{v}_k$  für den normierten Vektor  $\vec{v}_k$  der Kugel  $k$  steht.

$$\vec{v}_{1,t,n} = \vec{v}_{1,t} + A \cdot \hat{v}_1 \quad (4.56)$$

$$\vec{v}_{2,t,n} = \vec{v}_{2,t} + A \cdot \hat{v}_2 \quad (4.57)$$

Dieser Vektor  $\vec{v}_{k,t,n}$  wird nun anstelle von  $\vec{v}_{k,t}$  verwendet, wobei die maximale Energie des Vektors auf die Eingangsenergie beschränkt wird.

$$\vec{u}_1 = \vec{v}_{2,z} + \min(\vec{v}_{1,t,n}, |\vec{v}_1| \cdot \hat{v}_{1,t,n}) \quad (4.58)$$

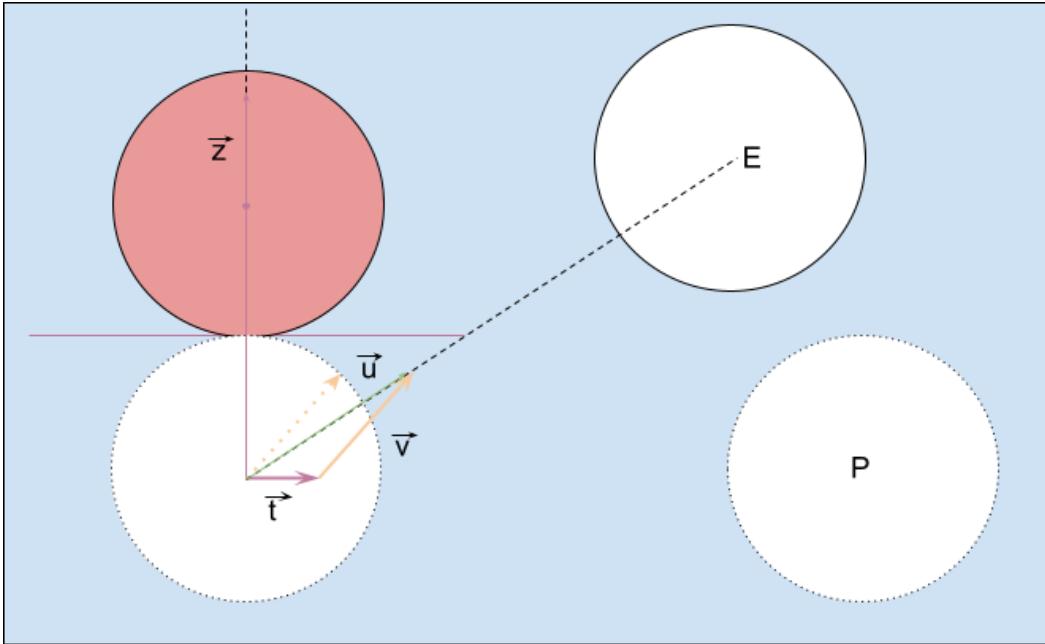
$$\vec{u}_2 = \vec{v}_{1,z} + \min(\vec{v}_{2,t,n}, |\vec{v}_2| \cdot \hat{v}_{2,t,n}) \quad (4.59)$$

Variante zwei verfolgt einen ähnlichen Ansatz, jedoch wird bei der Berechnung von  $\vec{v}_{k,t,n}$  anstelle des normierten der initiale Geschwindigkeitsvektor verwendet, welcher mit einem Prozentsatz skaliert wird. Dadurch wird die Variabilität berücksichtigt. Es gilt, dass  $A$  zwischen 0 und 1 liegt.

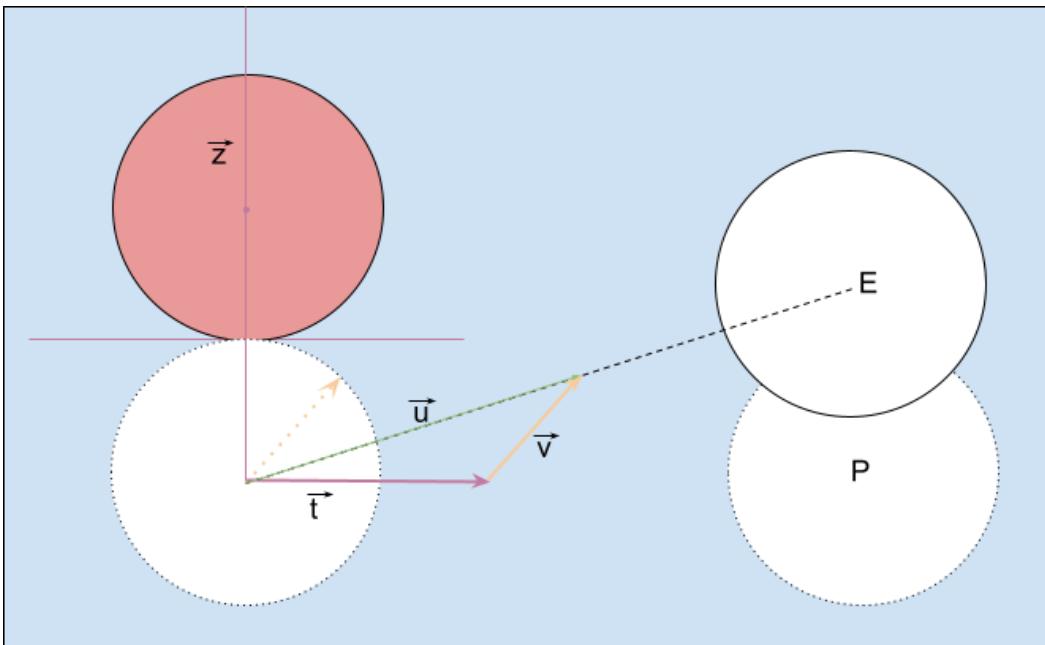
$$\vec{v}_{1,t,n} = \vec{v}_{1,t} + A \cdot \vec{v}_1 \quad (4.60)$$

$$\vec{v}_{2,t,n} = \vec{v}_{2,t} + A \cdot \vec{v}_2 \quad (4.61)$$

Die Berechnung von  $\vec{u}_1$  sowie  $\vec{u}_2$  geschieht auf dieselbe Art und Weise wie bei Variante eins.



(a) Topspin-Einfluss bei kleiner Geschwindigkeit



(b) Topspin-Einfluss bei grosser Geschwindigkeit

Abbildung 4.28: (a) Eine einkommende Geschwindigkeit  $\vec{v}$  wird auf die Komponenten  $\vec{z}$  und  $\vec{t}$  aufgeteilt. Ohne Berücksichtigung des Topsins wird die Position der weissen Kugel bei  $P$  berechnet, die effektive Position ist bei  $E$ . (b) Bei grösserer Geschwindigkeit ist die Abweichung zwischen  $P$  und  $E$  kleiner.

#### 4.6.4.3 Bandenreflektion

Sofern eine Kugel an eine Bahn stösst, so wird diese abgelenkt. In dem hier beschriebenen Modell wird der Drall[Unk21e], welcher die Bahn einer Kugel nach Kollision mit der Bahn ablenken würde, ignoriert. Das bedeutet, es wird davon ausgegangen, dass der Ausfallswinkel nach der Bandenreflektion gleich dem Einfallsinkel sei. Dazu kann die folgende Formel[Bou00] verwendet werden, wobei  $I$  der einfallende und  $R$  der ausgehende Weg der Kugel und  $N$  der Normalenvektor der Bahn sind:

$$R = I - 2 \cdot N \cdot (I \cdot N) \quad (4.62)$$

Da bei einer Bandenkollision Energie verloren geht, wird dieses Verhalten mit der Konstanten  $E_v$  modelliert. Diese gibt den

Verlust in Prozent an.

$$R = (I - 2 \cdot N \cdot (I \cdot N)) \cdot \frac{1}{1 + E_v} \quad (4.63)$$

#### 4.6.4.4 Kollisionsprüfung

Während der Simulation ist es notwendig zu prüfen, welche Kugeln mit welchen kollidieren könnten. Um zu wissen, ob eine Kugel eine Strecke zurücklegen kann, ohne mit einer anderen Kugel zu kollidieren, muss für jede andere Kugel geprüft werden, ob diese im Weg liegt. Daher sollte dieser Test effizient sein. Für den Test wird die zurückzulegende Strecke als Liniensegment zwischen Punkt  $A$  und Punkt  $B$  und die Position einer zu testenden Kugel  $C$  als Punkt verstanden. Anschliessend wird der Abstand zwischen dem Punkt  $C$  und dem Liniensegment  $AB$  geprüft, ob dieser kleiner als der Kugeldurchmesser ist. Sofern dies der Fall ist, liegt die Kugel an der Position  $C$  im Weg und es würde eine Kollision stattfinden, sofern die Ausgangskugel die Strecke  $AB$  rollt. Dies wird für jede Kugel geprüft.

#### 4.6.4.5 Ereignis - Energie-Transfer über Kugelkollision

Dieses Ereignis beschreibt die Kollision mit einer anderen Kugel zum Zeitpunkt  $t$ . Bekannt sind dabei die Beschleunigung  $\vec{a}$ <sup>9</sup>, die Geschwindigkeit  $\vec{v}$  und der Ort  $\vec{s}$  beider Kugeln. Diese werden entsprechend indexiert.

Weiterhin sind folgende Abstraktionen bekannt:

$$\vec{\Delta a} = \vec{\Delta a}_1 - \vec{\Delta a}_2 \quad (4.64)$$

$$\vec{\Delta v} = \vec{\Delta v}_1 - \vec{\Delta v}_2 \quad (4.65)$$

$$\vec{\Delta s} = \vec{\Delta s}_1 - \vec{\Delta s}_2 \quad (4.66)$$

Es entsteht ein Polynom vierten Grades<sup>10</sup>. Um diese Gleichung zu lösen bedarf es der entsprechenden Lösungsformel [Unk21h]:

$$ax^4 + bx^3 + cx^2 + dx + e = 0 \quad (4.67)$$

$$x_{1,2} = -\frac{b}{4a} - S \pm \frac{1}{2} \sqrt{-4S^2 - 2p + \frac{q}{S}} \quad (4.68)$$

$$x_{3,4} = -\frac{b}{4a} + S \pm \frac{1}{2} \sqrt{-4S^2 - 2p - \frac{q}{S}} \quad (4.69)$$

$$p = \frac{8ac - 3b^2}{8a^2} \quad (4.70)$$

$$q = \frac{b^3 - 4abc + 8a^2d}{8a^3} \quad (4.71)$$

$$S = \frac{1}{2} \sqrt{-\frac{2}{3}p + \frac{1}{3a}(Q + \frac{\Delta_0}{Q})} \quad (4.72)$$

$$Q = \sqrt[3]{\frac{\Delta_1 + \sqrt{\Delta_1^2 - 4\Delta_0^3}}{2}} \quad (4.73)$$

$$\Delta_0 = c^2 - 3bd + 12ae \quad (4.74)$$

$$\Delta_1 = 2c^3 - 9bcd + 27b^2e + 27ad^2 - 72ace \quad (4.75)$$

Wobei die Koeffizienten folgendermassen lauten:

$$a = \frac{1}{4} \cdot (\vec{\Delta a} \cdot \vec{\Delta a}) \quad (4.76)$$

$$b = \vec{\Delta a} \cdot \vec{\Delta v} \quad (4.77)$$

$$c = \vec{\Delta a} \cdot \vec{\Delta s} + \vec{\Delta v} \cdot \vec{\Delta v} \quad (4.78)$$

$$d = 2 \cdot (\vec{\Delta v} \cdot \vec{\Delta s}) \quad (4.79)$$

<sup>9</sup>Die Beschleunigung  $\vec{a}$  ist durch die Herleitung in Kapitel 8.10 gegeben.

<sup>10</sup>Für Herleitung, siehe Anhang 8.5

$$e = \vec{\Delta s} \cdot \vec{\Delta s} - D^2 \quad (4.80)$$

Durch Lösen des Polynoms werden alle Zeitpunkte  $x_{1-4} = t_{1-4}$  bestimmt, wobei der relevante Zeitpunkt  $t$  der Erste ist.

$$t = \min(t_1, t_2, t_3, t_4) \quad (4.81)$$

#### 4.6.4.5.1 Performanceverbesserungen

Das Lösen dieses Gleichungssystems erfordert viel Rechenzeit, was die Frage nach Performanceverbesserungen aufwirft. Diese werden durch Vorbedingungen erzielt, welche geprüft werden. Es findet zu Beginn eine Fallunterscheidung statt. Wenn die zweite Kugel statisch ist, so wird geprüft, ob die Distanz  $d$  zwischen der Position  $P_2$  der zweiten Kugel  $K_2$  und der halben Geraden  $g$  definiert durch die Position  $P_1$  der ersten Kugel  $K_1$  wie auch deren Geschwindigkeitsvektor  $\vec{v}$  kleiner oder gleich dem Kugeldurchmesser ist. Nur in diesem Fall kann es eine Kollision geben. Sämtliche Angaben beziehen sich auf die Grafik 4.29.

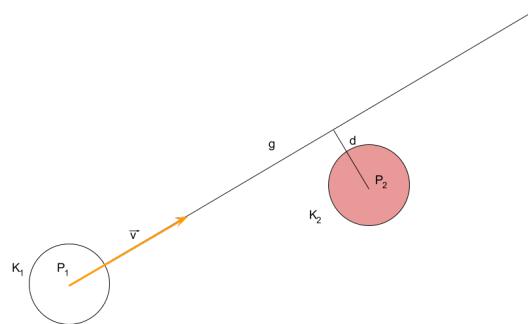


Abbildung 4.29: Vorbedingung einer Prüfung auf Kollision zwischen Kugeln bei statischer Beteiligung

Sobald beide Kugeln dynamisch sind reicht eine einfache Prüfung auf die Distanz nicht mehr. Es muss ein Schnittpunkttest der beiden halben Geraden, definiert durch die beiden Geschwindigkeitsvektoren, durchgeführt werden. Hierbei ist der Durchmesser der Kugel mitzubeachten, dies wird in Abbildung 4.30 dargestellt.

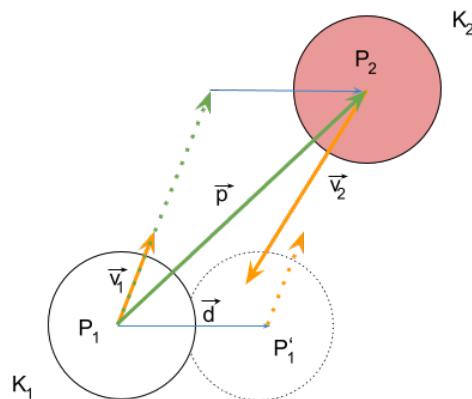


Abbildung 4.30: Vorbedingung einer Prüfung auf Kollision zwischen dynamischen Kugeln

Daher wird der Vektor  $\vec{p}$  zwischen den Positionen  $P_1$  und  $P_2$  gebildet (grün).

$$\vec{p} = P_2 - P_1 \quad (4.82)$$

Der erhaltene Vektor wird auf den Geschwindigkeitsvektor  $\vec{v}_1$  projiziert (grün gestrichelt).

$$\vec{p}' = (\vec{p} \cdot \hat{v}_1) \cdot \hat{v}_1 \quad (4.83)$$

Danach wird der Zwischenvektor  $\vec{d}$  gebildet (blau).

$$\vec{d} = \vec{p} - \vec{p}' \quad (4.84)$$

Die Kugel  $K_1$  wird um diesen Vektor verschoben, es resultiert  $P'_1$ .

$$P'_1 = P_1 + \vec{d} \quad (4.85)$$

Von dieser Position aus kann nun ein Schnittpunkttest zwischen den beiden halben Geraden, durch die Geschwindigkeitsvektoren definiert, durchgeführt werden.

Zudem gibt es zwei Spezialfälle zu beachten. Der erste beschreibt die Handhabung bei paralleler Fortbewegung der Kugeln. Da dies ein sehr seltener Fall ist und wohl kaum auftreten wird, wird die Berechnung zugelassen, wenn der Vektor  $\vec{d}$  kleiner oder gleich gross wie der Durchmesser ist. Die Situation wird in Abbildung 4.31 erläutert, die Berechnungen für den Vektor  $\vec{d}$  sind äquivalent zu den vorhergehenden Berechnungen.

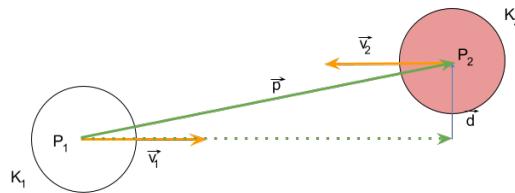


Abbildung 4.31: Vorbedingung einer Prüfung auf Kollision zwischen dynamischen parallel velaufenden Kugeln

Der letzte Spezialfall befasst sich mit der Tatsache, dass ein Schnittpunkttest, so wie er unter Berücksichtigung der obigen Fälle durchgeführt wird, kein Ergebnis findet, wenn der Schnittpunkt hinter einer der Geraden liegt. Diese Situation kann in Abbildung 4.32 entnommen werden. Es ist ersichtlich, dass der Geschwindigkeitsvektor  $\vec{v}_1$  einen grösseren Betrag aufweist als der Geschwindigkeitsvektor  $\vec{v}_2$ . Um diesem Umstand Rechnung zu tragen, wird die Startposition der Geraden um den Durchmesser nach hinten verschoben.

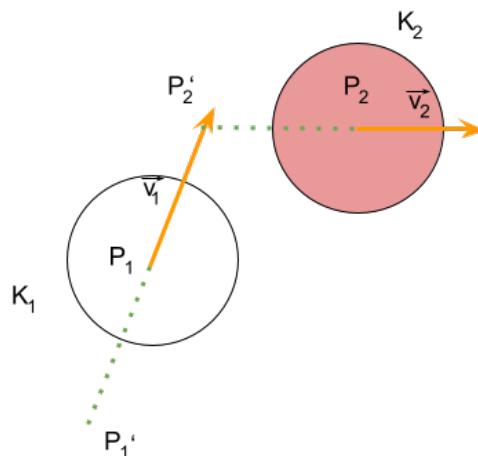


Abbildung 4.32: Vorbedingung einer Prüfung auf Kollision zwischen dynamischen hintereinander velaufenden Kugeln

#### 4.6.4.6 Ereignis - Energie-Transfer über Bandenkollision

Dieses Event beschreibt die Kollision mit der Bande. Es soll der Zeitpunkt  $t$  der Kollision mit der Bande festgestellt werden. Der Algorithmus funktioniert so, dass zuerst geprüft wird, ob eine Kollision stattfinden kann. Dies erfolgt über einen Schnittpunkt-Test zwischen einer Linie und einem Liniensegment. Eine Bande kann als Liniensegment zwischen dem Startpunkt  $R_1$  und  $R_2$  betrachtet werden. Diese Punkte müssen demnach bekannt sein. Weiterhin ist der Geschwindigkeitsvektor  $\vec{v}$  und die Position der Kugel  $C$  bekannt. Aufgrund dieser Informationen kann eine Linie definiert werden.

Die Punkte  $R_1$  und  $R_2$  werden um den Kugelradius zur Tischmitte verschoben, damit dem Kugelradius Rechnung getragen wird und dieser nicht weiter betrachtet werden muss. Daraus ergeben sich  $R'_1$  und  $R'_2$ <sup>11</sup>.

<sup>11</sup>Für Herleitung, siehe Anhang 8.6

Erster Schritt - Prüfe, ob Kollision möglich:

Dazu wird festgestellt, ob ein Schnittpunkt zwischen der Lauflinie der Kugel und dem Liniensegment der Bande existiert<sup>12</sup>. Sofern kein Schnittpunkt vorliegt, können weitere Tests abgebrochen werden, da keine Kollision mit der Bande stattfinden wird.

Zweiter Schritt - Ort der Kollision bestimmen:

Betrachte dazu eine der Gleichungen mit zugehörigem  $\lambda$ :

$$s(\vec{\lambda}_1) = \vec{R}'_1 + \lambda_1 \cdot \Delta \vec{R}' \quad (4.86)$$

$$s(\vec{\lambda}_2) = \vec{C} + \lambda_2 \cdot \vec{v} \quad (4.87)$$

Dritter Schritt - Zeitpunkt der Kollision bestimmen:

Dies kann über die folgenden Gleichungen gelöst werden:

$$\Delta t_1 = \frac{-v + \sqrt{v^2 + 2 \cdot a \cdot \Delta s}}{a} \quad (4.88)$$

$$\Delta t_2 = \frac{-v - \sqrt{v^2 + 2 \cdot a \cdot \Delta s}}{a} \quad (4.89)$$

Bevor die Lösung zu  $t_1$  und  $t_2$  berechnet wird, muss jedoch der Radikand auf folgende Eigenschaft geprüft werden:

$$0 \leq v^2 + 2 \cdot a \cdot \Delta s \quad (4.90)$$

Nur in diesem Fall gibt es eine Lösung und somit an dieser Position eine Kollision. Ansonsten hat die Kugel vorher schon ihre Gesamtenergie verloren und steht still. Es wird das Minimum der beiden Zeiten  $\Delta t_1$  und  $\Delta t_2$  verwendet.

Probleme hat dieser Ansatz ergeben, wenn die Kugel z.B. nahe der Bande rollt oder teilweise auch aufgrund der Ungenauigkeiten beim Rechnen mit Fließkommazahlen. So wurde die Kollision mit derselben Bande mehrfach aufeinanderfolgend detektiert und die Kugeln konnte so durch die Bande hindurchrollen. Das Problem wurde behoben, indem die letzte Bande, mit welcher die Kugel kollidierte, von der Ereignisberechnung ausgeschlossen wurde. Die Bande wird erst wieder miteinbezogen, wenn die Kugel eine andere Bande oder eine andere Kugel getroffen hat.

#### 4.6.4.7 Ereignis - Totaler Energieverlust

Dieses Ereignis trifft ein, sobald eine Kugel stillsteht. Der Zeitpunkt wird über die Formel 4.91 beschrieben<sup>13</sup>.

$$t = \max\left(\frac{-v_{0,x}}{a_x}, \frac{-v_{0,y}}{a_y}\right) \quad (4.91)$$

#### 4.6.4.8 Ereignis - Verlassen des Systems

Der Ereigniszeitpunkt erfolgt über zwei Schritte. Im ersten Schritt wird der Kollisionspunkt der Kugel mit dem Zielkreis bestimmt. Anhand dieser Information kann die zurückgelegte Distanz der Kugel bestimmt werden. Es gelten die folgenden Definitionen:

$\vec{C}$  = Zielmittelpunkt

$\vec{p}$  = Position der Kugel

$\hat{v}$  = Normierte Geschwindigkeit der Kugel

Die nachfolgende Formel definiert, mit welchem Faktor der Geschwindigkeitsvektor multipliziert werden muss, damit er die Länge der zurückzulegenden Distanz zum Kollisionsort mit dem Ziel erreicht<sup>14</sup>. Zu beachten ist, dass / nur gültig ist, wenn es grösser als 0 ist.

$$I_1 = \frac{-(2 \cdot \hat{v} \cdot (\vec{p} - \vec{C})) + \sqrt{4 \cdot (\vec{p} - \vec{C}) \cdot (\vec{p} - \vec{C}) \cdot r^2}}{2} \quad (4.92)$$

<sup>12</sup>Siehe Anhang 8.6

<sup>13</sup>Die Herleitung erfolgt in Kapitel 8.4. Die Beschleunigung  $\ddot{a}$  ist durch die Herleitung in Kapitel 8.10 gegeben.

<sup>14</sup>Für Herleitung, siehe Anhang 8.7.

$$l_2 = \frac{-(2 \cdot \hat{v} \cdot (\vec{p} - \vec{C})) - \sqrt{4 \cdot (\vec{p} - \vec{C}) \cdot (\vec{p} - \vec{C}) \cdot r^2}}{2} \quad (4.93)$$

$$l = \max(\min(l_1, l_2), 0) \quad (4.94)$$

$$d(l) = l \cdot \hat{v} \quad (4.95)$$

Nachdem die Distanz bestimmt wurde, kann der Kollisionszeitpunkt berechnet werden<sup>15</sup>. Zu beachten ist, dass keine Lösung existiert, wenn die Diskriminante kleiner denn 0 ist. Die Kugel rollt zu langsam und der Reibungsverlust ist zu gross, als dass sie den Zielkreis erreichen könnte.

$$t_1 = \frac{-|\vec{v}| + \sqrt{|\vec{v}|^2 + 2 \cdot |\vec{a}| |d(l)|}}{|\vec{a}|} \quad (4.96)$$

$$t_2 = \frac{-|\vec{v}| - \sqrt{|\vec{v}|^2 + 2 \cdot |\vec{a}| |d(l)|}}{|\vec{a}|} \quad (4.97)$$

$$t = \min(t_1, t_2) \quad (4.98)$$

#### 4.6.4.9 Ereignis - Rollen

Wenn eine Kugel zentral angestossen wird, dann gleitet sie einen Moment ohne zu rollen. Die Eigenschaft des Rollens wird ebenso als Ereignis geführt. Der Zeitpunkt, wann dieses Ereignis eintritt, kann über die Formel<sup>16</sup> 4.99 berechnet werden, wobei  $\mu$  für den Gleitreibungskoeffizienten steht.

$$t = \frac{v_0}{\frac{7}{2} \cdot g \cdot \mu} \quad (4.99)$$

Dieser Zeitpunkt wird einmal beim Anstossen einer Kugel berechnet und zwischengespeichert. Würde dieses Ereignis nach jedem Vorherigen neu berechnet werden, so würde der Rollzeitpunkt immer weiter in der Zukunft liegen. Dieses Zwischen-speichern wäre nicht nötig, würde die Simulation sämtliche Ereignisse vorausberechnen, zwischenspeichern, abarbeiten und bei Bedarf invalidieren<sup>17</sup>.

Weiterhin wird angenommen, dass gleitende Kugeln nach einer Kollision mit einer anderen Kugel weitergleiten. Es ist auch bekannt, dass die Kugel nach der Kollision mit der Bande sofort zu rollen beginnt[Dan13].

#### 4.6.4.10 Bewertungsfunktion

Wie auch die Suche nach einem guten Lösungskandidaten muss das Endresultat der Simulation bewertet werden. In dem Fall geht es um die Bestimmung der Belohnung eines Stosses oder umgekehrt die Bestrafung in Form von Kosten. In einem ersten Schritt wird die Belohnung berechnet, welche sich an der Anzahl und dem Typen der eingelochten Kugeln richtet. Es gibt die Schwarze (7), Pinke (6), Blaue (5), Braune (4), Grüne (3), Gelbe (2) und Rote (1) Kugel. Die Punkte sind jeweils hinter dem Typen aufgeführt. In einem Stoss wird eine maximal mögliche Punktzahl von 7 definiert. Sollten mehr Punkte erzielt werden, wird dies im Weiteren keinen Einfluss mehr haben, dazu später mehr. Dieser Score  $S$  wird durch eine Normierung der erzielten Punkte  $p$  anhand der maximal möglichen Punkte  $p_{max}$  verkleinert. Zumeist wird er zwischen 0 und 1 liegen, er kann aber auch grösser werden.

$$Sp = \frac{p}{p_{max}} \quad (4.100)$$

Anschliessend wird der Score auf eine Bestrafung/Kosten umgerechnet. Hierbei wird ignoriert, dass mehr Punkte wie die maximal möglichen erzielt wurden. Dies geschieht, in dem das Minimum von 1 und dem Score  $S$  genommen wird.

$$K_p = 1 - \min(S, 1) \quad (4.101)$$

<sup>15</sup>Für Herleitung, siehe Anhang 8.7. Die Beschleunigung  $\vec{a}$  ist durch die Herleitung in Kapitel 8.10 gegeben.

<sup>16</sup>Die Herleitung erfolgt in Kapitel 8.8

<sup>17</sup>Eine mögliche Verbesserung des Algorithmus wird in Kapitel 8.14 beschrieben.

Des weiteren soll die Stossstärke, gemessen an der Startgeschwindigkeit des Spielballs, in die Bewertung einfließen. Der Grund ist, dass bei stärkeren Stößen ein Präzisionsfehler wahrscheinlicher ist und daher schwachere Stöße zu bevorzugen sind. Die Kosten der Stossstärke werden wie folgt mithilfe der Stossgeschwindigkeit  $v$  und einer Maximalgeschwindigkeit von  $V_{max} = 5 \frac{m}{s}$  in einem Wert zwischen 0 und 1 ausgedrückt.

$$K_v = \frac{v}{V_{max}} \quad (4.102)$$

Die Gesamtkosten bildet anschliessend die gewichtete Summe beider Kosten  $K_p$  und  $K_v$ . Das Gewicht wurde als  $\alpha = \frac{1}{2}$  definiert.

$$K = \alpha \cdot K_p + (1 - \alpha) \cdot K_v \quad (4.103)$$

Dieser so erhaltene Wert ist noch nicht aussagekräftig genug. Er soll in Relation zu der Schwierigkeit des Stosses, also den Suchkosten, angegeben werden können. Die Belohnung soll zum Beispiel nur 50% Gewicht haben. Um dies zu erreichen, werden die definitiven Kosten über eine solche Normierung erreicht, wobei  $K$  die nicht normierten Kosten zwischen 0 und 1,  $g$  die Gewichtung in Prozent gegenüber der Schwierigkeit und  $K_s$  die Suchkosten sind.

$$K_{norm} = K \cdot g \cdot K_s \quad (4.104)$$

#### 4.6.5 Animation

Um eine möglichst optimale Benutzerfreundlichkeit zu gewährleisten, soll der berechnete Stoß über eine Animation auf dem Tisch dargestellt werden. Dies wird durch eine Übersetzung des physikalischen Simulationsmodells, wie in Kapitel 4.6.3 beschrieben, erzielt.

Die Animation besteht aus Keyframes, zwischen welchen interpoliert werden kann. Das grundlegende Prinzip wird in Abbildung 4.33 visualisiert. Auf der linken Seite wird der Zeitstrahl angegeben. Die Zeit schreitet in vertikaler Richtung nach unten voran. Der rote Pfeil markiert den aktuellen Zeitpunkt, dieser liegt zwischen Keyframe 1 und Keyframe 2. Der Status zu diesem Zeitpunkt kann durch Interpolation zwischen den genannten Keyframes berechnet werden.

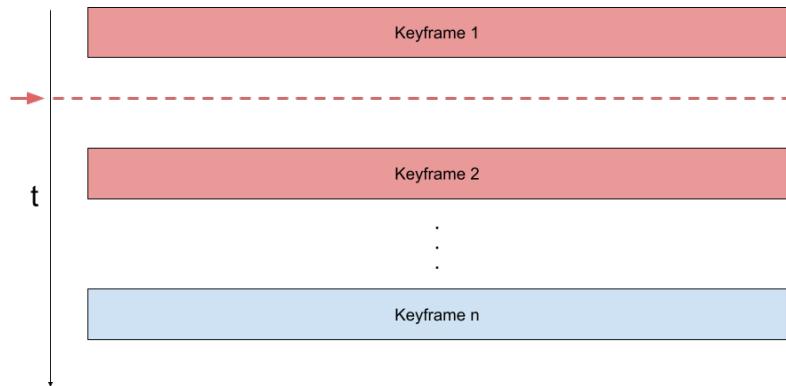


Abbildung 4.33: Keyframe Animation

Das Animationsmodell wird anhand des Simulationsmodells abgeleitet. Dazu kann Abbildung 4.34 betrachtet werden. Wie in Kapitel 4.6.3 erklärt, besteht das physikalische Simulationsmodell aus in Layern gruppierten Knoten. Diese Knoten haben jeweils eine Eingabe- wie auch eine Ausgabeschicht.

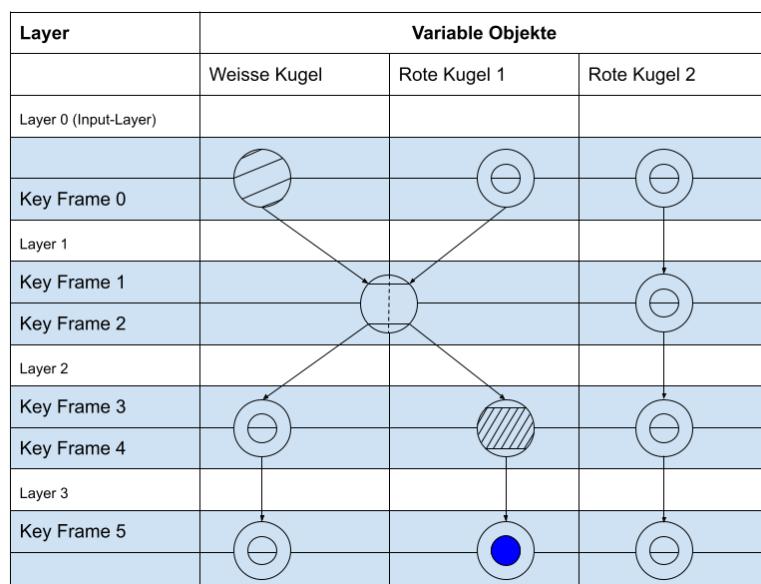


Abbildung 4.34: Simulationsmodell mit Keyframes

Jede Eingabe- wie auch Ausgabeschicht bildet so im Animationsmodell ein Keyframe. Die Besonderheit liegt darin, dass nicht wie bei einer nach Lehrbuch aufgebauten Keyframe-Animation mit Fortlaufen der Zeit die Keyframes durchiteriert<sup>18</sup> werden. Die Keyframes, welche aufgrund des Simulationsmodells generiert werden, werden paarweise als Fenster betrachtet. Dieses Paarfenster wird bei Erreichen des Endkeyframes auf das nächste Paar verschoben. Die Funktionsweise ist in Abbildung 4.35 beschrieben. Dort werden die verschiedenen Keyframefensterpaare unterschiedlich rot markiert. Die Zeit schreitet wiederum

<sup>18</sup>Durchiterieren bedeutet, dass wenn ein Keyframe vorher das Ende markierte, wie es das Keyframe 2 in Abbildung 4.33 tut, dann wird das Keyframe 2 mit Fortlaufen der Zeit zum Startkeyframe und das nächstfolgende zum neuen Ende.

in vertikaler Richtung nach unten voran, der rote Pfeil auf der linken Seite zeigt eine mögliche aktuelle Position innerhalb einer Animation. Demnach wäre das zweite Keyframefenster mit den Keyframes 2 und 3 aktiv.

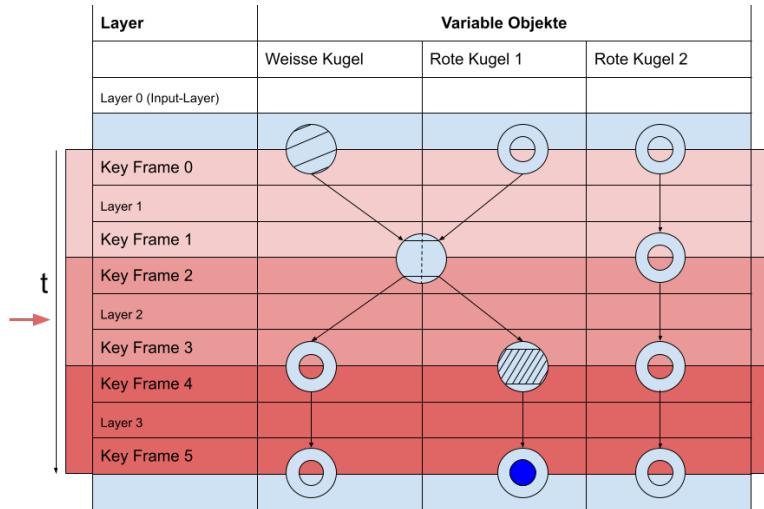


Abbildung 4.35: Simulationsmodell mit paarweisen Keyframefenstern

Die Begründung der etwas spezielleren Handhabung liegt an der Tatsache, dass bei einem physikalischen Ereignis wie einer Kugelkollision eine Veränderung des Status geschieht. So werden im Wesentlichen die Magnituden wie auch die Richtungen der Geschwindigkeitsvektoren verändert. Würde nur ein Keyframe bei einem solchen Ereignis generiert werden, dann könnte nicht mehr zwischen den Geschwindigkeitsvektoren aufgrund der geänderten Richtung interpoliert werden.

Die so generierte Animation kann als Gruppierung von einzelnen Keyframeanimationen, wie in Abbildung 4.33 angegeben, mit jeweils nur zwei Keyframes angesehen werden.

## 4.6.6 Tiefensuche

Um dem Spieler einen möglichst optimalen Stoss vorzuschlagen, sollte nicht nur der aktuelle Spielstand, sondern auch die zukünftigen Spielstände und deren mögliche Stösse berücksichtigt werden. Ein Stoss ist nur gut, wenn er für die Darauffolgenden eine ebenso gute, wenn nicht bessere, Ausgangslage bietet. Ein guter Stoss in der aktuellen Situation kann evtl. zu schwierigeren Stößen führen, weil der Spielball schlecht platziert wurde. Daher wurde beim Lösungsdesign des Suchalgorithmus darauf geachtet, dass eine Suche über mehrere Stösse ermöglicht wird. Es handelt sich immer noch um eine Graphensuche, deren Suchraum jedoch erweitert wird. Ein korrespondierender Suchbaum wird in Abbildung 4.36 veranschaulicht. Die orangefarbenen Knoten bilden die Root-Knoten, die Repräsentation der Ziellöcher. Die grünen Knoten sind Expansionsknoten. Diese stehen z.B. für einen Weg über eine Kugel oder eine Bande. Die beiden Knoten sind bereits aus Kapitel 4.6.1 bekannt. Sobald die weiße Kugel expandiert wird, ist eine Kandidatensuche beendet und es startet eine Simulation. Die Simulation wird beim Durchführen des Suchalgorithmus ebenfalls als Knoten in diesem Suchbaum angesehen, hier blau eingefärbt. Nach der Berechnung eines Simulationsknotens kann die Suche bei genügender Tiefe beendet oder wie in Ebene 3 fortgeführt werden. Das Resultat des Simulationsknotens bildet den neuen Spielstand und es werden neue Root-Knoten darauf basierend expandiert. Die Suche kann so zukünftige Stösse in die Bewertung des aktuellen Stosses einbeziehen. Das Resultat der Tiefensuche sind nicht mehr nur einzelne Stösse, sondern Abfolgen von Stößen, welche nacheinander ausgeführt werden könnten.

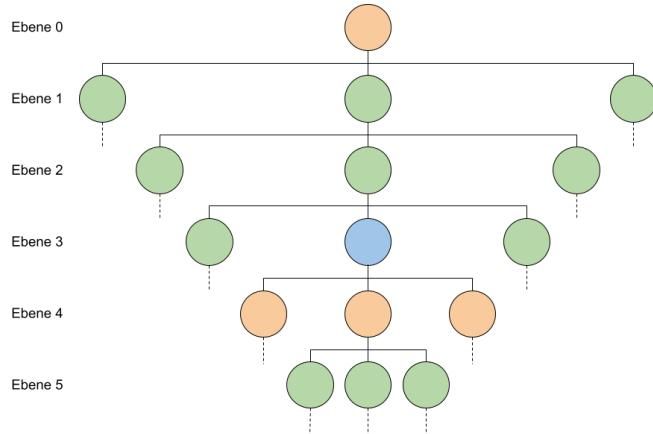


Abbildung 4.36: Suchbaum der Tiefensuche

### 4.6.6.1 Regeln für Snooker

Wird über mehrere Stösse gesucht, sind gewisse Regeln des Spiels einzuhalten. Bei Snooker werden abwechselungsweise rote und farbige Kugeln versenkt, solange noch rote Kugeln auf dem Tisch sind. Die farbigen Kugeln haben unterschiedliche Werte: Gelb(2), Grün(3), Braun(4), Blau(5), Pink(6), Schwarz(7)[Unk].

Gibt es noch eine oder mehrere rote Kugeln zu spielen, wird die versenkte farbige Kugel, ausser Rote, wieder auf ihrem Spot (Aufsetzmarke) platziert. Ist der Spot nicht frei, so wird sie auf den nächst höherwertigen Spot gelegt. Die Wertung der Spots entspricht der Wertung der Kugeln, welche bei der Ausgangsstellung platziert wurden. Die Ausgangsstellung ist in Abbildung 4.37 veranschaulicht. Wenn alle Spots belegt sind, wird die Kugel so nah wie möglich an ihrem ursprünglichen Spot in Richtung der Kopfbande<sup>19</sup> platziert[Unk].

Diese möglichst nahe Platzierung wurde über eine Suche nahe des Spots in einem diskretisierten Raum implementiert. Abbildung 4.38 veranschaulicht eine solche Situation, in der die Position  $s$  besetzt ist und daher eine möglichst nahe Platzierung erforderlich ist. Es wird angenommen, dass sich die Kopfbande auf der rechten Seite befindet. Demnach wird zuerst eine Position in dieser Richtung geprüft, danach eine Position in senkrechter Richtung zur Kopfbande und anschliessend die Position entgegengesetzt zur Kopfbande. Dies ist in Abbildung 4.38 in den Schritten 1a bis 1d zu sehen. Die Fringe  $f$  beinhaltet potentielle Positionen für die zu platzierende Kugel, die noch auf Verfügbarkeit geprüft werden müssen. Zu Beginn enthält  $f$  nur die belegte Position  $s$ . Ist eine Platzierung nicht möglich, werden die Positionen  $a_1$  bis  $a_4$  als neue Fringe betrachtet. Danach werden die Positionen innerhalb der Fringe wiederum zuerst Richtung Kopfbande, danach in senkrechter Richtung zu dieser und anschliessend noch entgegengesetzt auf Verfügbarkeit geprüft. Zu sehen in den Schritten 2a bis 2d. Dies wird fortgeführt, solange die Kugel nicht platziert werden konnte und die möglichen Positionen nicht zu weit vom Spot entfernt sind oder ausserhalb des Tisches liegen. Wenn die Kugel nicht platziert werden konnte, wird dies als Fehler betrachtet und die Suche wird abgebrochen.

<sup>19</sup>Die Kopfbande ist diejenige, welcher der schwarzen Kugel bei der Ausgangsstellung am nächsten ist.

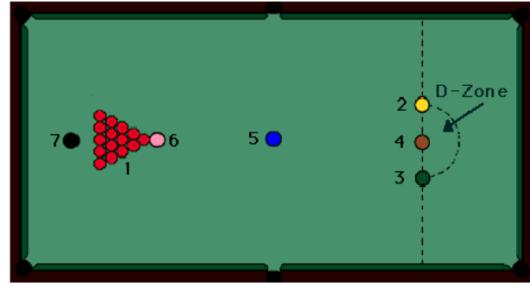


Abbildung 4.37: Ausgangslage bei Snooker[Unk]

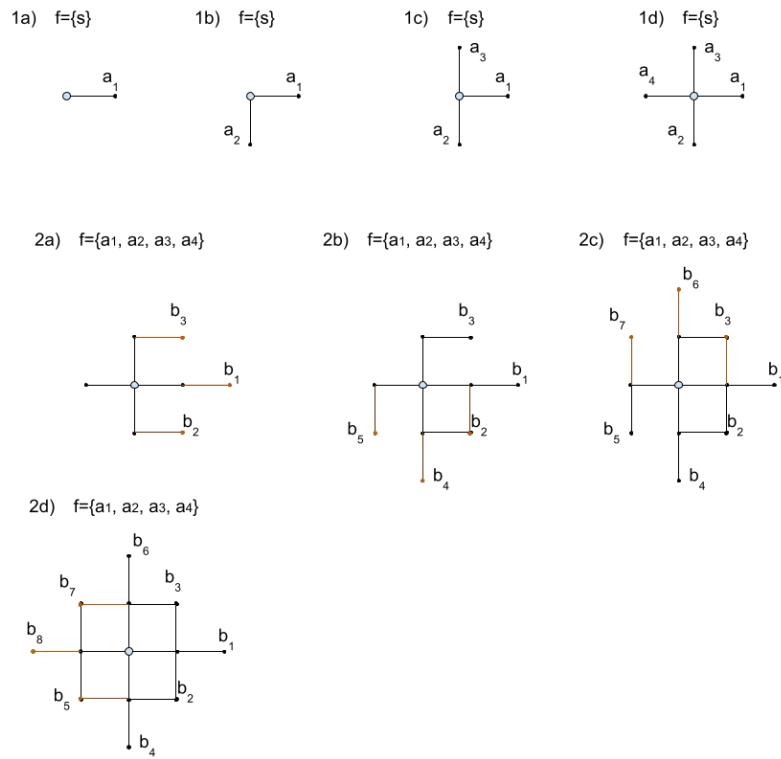


Abbildung 4.38: Replatzierung einer Kugel nahe am Spot  $s$

Gibt es keine roten Kugeln mehr, werden die farbigen Kugeln in der Reihenfolge ihrer Punkte aufsteigend, also Gelb, Grün, Braun, Blau, Pink, Schwarz versenkt und das Spiel ist beendet[Unk].

## 4.7 Berechnungsprozess

Die Berechnung eines optimalen Stosses wird aufgrund der vielen Möglichkeiten sehr zeitintensiv, weswegen die expandierten Teilschritte parallel gerechnet werden.

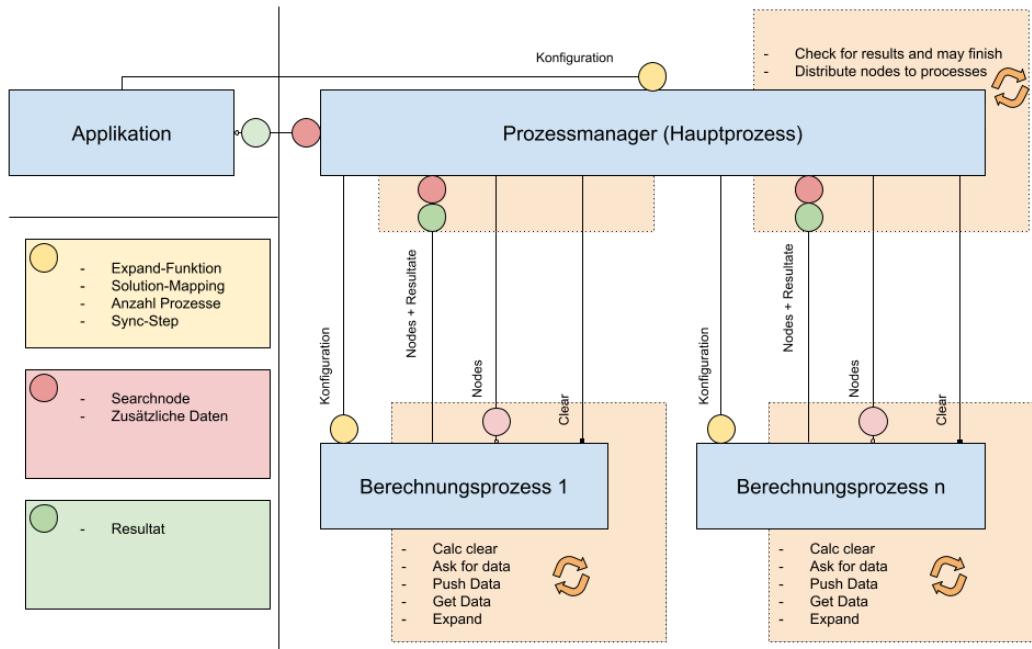


Abbildung 4.39: Berechnungsprozess

Abbildung 4.39 erläutert die Durchführung dieser Expansionsschritte. Die Applikation auf der linken Seite liefert die konkreten Aufgaben, welche von einem Pool an Berechnungsprozessen bearbeitet werden sollen. Sie beinhaltet die Logik, um eine solche Aufgabe zu lösen, da die Berechnungsprozesse selbst nicht über dieses Wissen verfügen. Der Pool von Berechnungsprozessen wird vom Prozessmanager verwaltet und dieser ist die zentrale Ansprechsperson der Applikation.

Datenaustausche sind über gefärbte Kreise repräsentiert. Handelt es sich um einen asynchronen Datenaustausch, dann ist der entsprechende Kreis heller eingefärbt und die Verbindungsleitung ist durch einen nicht ausgefüllten Punkt gekennzeichnet. Es werden die Farben Rot den zu bearbeitenden Daten, Gelb den Konfigurationsdaten und Grün den Resultaten zugeordnet. Signale sind durch ein ausgefülltes Quadrat an der Verbindungsleitung gekennzeichnet. Das Signal oder die Daten sind jeweils beim Empfänger angegeben. Repetitive Aufgaben sind durch ein hinterlegtes orangefaches Rechteck markiert.

Bei der Erstellung des Prozessmanagers werden alle Konfigurationen mitgeliefert. Der Prozessmanager erzeugt während seiner Instanziierung die Prozesse, welche ebenfalls direkt konfiguriert werden. Die Prozesse laufen nun im Hintergrund und warten auf Daten.

Erhält die Applikation eine Suchanfrage, ruft sie in einem ersten Schritt den Prozessmanager auf. Dieser nimmt noch zu berechnende Daten entgegen. Im Fall von Billard sind dies die Root-Nodes des Suchbaums. Jeder Root-Node repräsentiert ein Loch. Der Prozessmanager gibt in dem Fall eine Datenstruktur zurück, welche es erlaubt, in Zukunft auf die Anfrage zu antworten. In dieser Antwort werden die Resultate geliefert. Der Hauptprozess prüft regelmässig, ob die Bearbeitung der gestellten Aufgaben beendet werden soll. Dies ist der Fall, wenn entweder genügend Lösungen gefunden wurden, die maximal zur Verfügung stehende Zeit abgelaufen ist oder keine noch zu expandierenden Nodes mehr vorhanden sind. Sollte die Berechnung nicht abgebrochen werden, dann werden die offenen Anfragen der Berechnungsprozesse beantwortet. Der Prozessmanager verteilt alle ihm gemeldeten noch zu bearbeitenden Daten an die einzelnen Berechnungsprozesse.

In Abbildung 4.40 wird das Ziel verdeutlicht, welches der Prozessmanager bei einer Verteilung dieser Daten verfolgt. So sollen die jeweils  $n$  besten Kandidaten, wobei  $n$  für die Anzahl der Berechnungsprozesse steht, auf die verschiedenen Berechnungsprozesse verteilt werden. Danach erhalten die Berechnungsprozesse die nächstbesten  $n$ -Kandidaten. Dadurch wird sichergestellt, dass immer an den vielversprechendsten Arbeitspaketen weitergemacht wird.

Der Algorithmus 5 zeigt, wie die offenen Arbeitspakete auf die Berechnungsprozesse verteilt werden, damit das Ziel in Abbildung 4.40 erreicht werden kann. So werden zuerst alle Antworten auf die offenen Anfragen erstellt. Dies geschieht in einer „While-Loop“, welche solange läuft, wie es noch offene Arbeitspakete hat. In jedem Schritt der Schleife

In Progress (Arbeitspakete nach Kosten aufsteigend)

1	2	3	4	5	6	7
Berechnungsprozess 1		Berechnungsprozess 2		Berechnungsprozess 3		
	1			2		3
	4			5		6
	7					Drittbeste 3
						Beste 3
						Zweitbeste 3

Abbildung 4.40: Berechnungsprozessverteilung

fe wird ein Arbeitspaket einer Antwort für eine Anfrage hinzugefügt. Am Ende der Schleife wird eine Index-Variable hochgezählt. Sollte diese die Anzahl der offenen Anfragen erreichen, beginnt sie wieder bei 0. Dies ist z.B. in Abbildung 4.40 beim Wechsel von Arbeitspakte 3 zu 4 ersichtlich. Sobald keine offenen Arbeitspakete mehr existieren, werden sämtliche Anfragen beantwortet, zu welchen Antworten erstellt werden konnten. Wenn z.B. mehr Anfragen als Arbeitspakte existieren, bleiben diese erhalten und werden im nächsten Synchronisationszyklus beantwortet<sup>20</sup>.

**Function** distribute(*requests*: List[future], *inProgress*: Queue[Node]) —> List[future]

```

answers: List[List[Node]] ← list()
forRequest ← 0
while ! inProgress.empty() do
    if answers[forRequest] is null then
        | answers[forRequest] = list()
    end
    node ← pop(inProgress)
    answers[forRequest] ← append(node, answers[forRequest])
    forRequest ← (forRequest + 1) % size(requests)
end
while ! answers.empty() do
    request ← pop(requests)
    fulfill(request, pop(answers))
end
return requests

```

**Algorithm 5:** Algorithmus zur Durchführung eines Expansionsschritts im Berechnungsprozess

In einem Zyklus eines Berechnungsprozesses wird zuerst geprüft, ob eine Berechnung abgebrochen werden soll. Dies ist der Fall, wenn der Prozessmanager ein „Clear-Signal“ gesendet hat. Dieses Signal tritt auf, wenn eine neue Berechnung gestartet oder wenn eine aktuell Laufende erfolgreich beendet oder abgebrochen wird.

Der Berechnungsprozess stellt in einem nächsten Schritt eine Datenanfragen an den Prozessmanager, sollte keine offene oder noch nicht bearbeitete Anfrage existieren. Danach erfolgt die Prüfung, ob die Zeit zwischen einem Synchronisationsschritt abgelaufen ist. In dem Fall liefert der Berechnungsprozess seine besten weiterzuführenden Berechnungen wie auch Resultate dem Prozessmanager. Er wird demnach für eine Weile an weniger erfolgsversprechenden Resultaten weiterarbeiten. In einem weiteren Schritt wird geprüft, ob eine Datenanfrage bereits beantwortet wurde. Ist dies der Fall, so werden die Daten den zu bearbeitenden Daten des Berechnungsprozesses hinzugefügt. Zuletzt erfolgt der Expansionsschritt, wobei der erfolgsversprechendste Kandidat expandiert wird. Der Algorithmus 6 zeigt, wie dies in etwa funktionieren kann. Als Eingabe dient eine priorisierte Queue von offenen Arbeitspaketen sowie die Implementation der „Expand-Funktion“. Wenn es offene Arbeitspakte gibt, dann wird das Beste genommen und über die Implementation der „Expand-Funktion“ expandiert. Das Resultat dieses Schrittes wird danach entweder den Lösungen oder den offenen Arbeitspaketen hinzugefügt. Am Schluss werden die offenen

<sup>20</sup>Man stelle sich bei der Abbildung 4.40 nur zwei offene Arbeitspakete vor. Demnach würde Paket 1 auf den Berechnungsprozess 1 und Paket 2 auf den Berechnungsprozess 2 verteilt werden. Die Anfrage des Berechnungsprozesses 3 bliebe offen.

Arbeitspakete wie auch die Lösungen zurückgegeben.

```
Function expand(inProgress: Queue[Node], expandImpl: functional) —> Queue[Node], Queue[Node]
    solutions ← priority_queue()
    if !inProgress.empty() then
        node ← pop(inProgress)
        expandedNodes ← expandImpl(node)
        for expandedNode in expandedNodes do
            if expandedNode.isSolution() then
                | solutions ← append(expandedNode, solutions)
            end
            else
                | inProgress ← append(expandedNode, inProgress)
            end
        end
    end
    return inProgress, solutions
```

**Algorithm 6:** Algorithmus zur Durchführung eines Expansionsschritts im Berechnungsprozess

Die Synchronisation zwischen den Prozessen wird in Abbildung 4.41 erläutert. Sie findet nach einer konfigurierten Zeit  $k$  statt. Diese wird in Millisekunden [ms] angegeben. Da die Synchronisation ein exklusives Verwenden einer geteilten Ressource erfordert, um die erledigte Arbeit dem Prozessmanager mitzuteilen, erhalten alle laufenden Prozesse (dies umfasst den Hauptprozess des Prozessmanagers wie auch alle Berechnungsprozesse) ein Zeitfenster zugeteilt, welches nach  $k$  Millisekunden startet. Danach kann jeder Prozess nach  $k + i \cdot n$ , wobei  $i$  für die Prozessid beginnend bei 0 und  $n$  für das Zeitfenster einer Synchronisation steht, mit der Synchronisation beginnen. Es wird so verhindert, dass zu viele Prozesse auf die Ressource des Prozessmanagers warten müssen und untätig bleiben. Ein weiterer Vorteil dabei ist die Tatsache, dass in der Zeit der Synchronisation einige Prozesse Zeit erhalten, um eher schlechter bewertete Kandidaten weiterzuverfolgen. Dadurch ist es möglich, dass ein sehr gutes Ergebnis gefunden werden kann, obwohl dessen Kandidat anfänglich als schlecht beurteilt wurde.

Prozess\Zeit [ms]	$k$	$k + 0n$	$k + n$	$k + 2n$	$2k$	$2k + 0n$	$2k + n$	$2k + 2n$
Hauptprozess								
Berechnungspr. 1								
Berechnungspr. 2								

Abbildung 4.41: Berechnungsprozesssynchronisation

Die Abbildung 4.41 zeigt den Hauptprozess wie auch zwei Berechnungsprozesse, welche eine Synchronisation in einem Intervall von  $k$  Millisekunden durchführen. Grüne Spalten stehen hierbei für die Zeit, welche für die Berechnung einer Lösung verwendet wird, also als produktiv bezeichnet werden kann. Rote Spalten hingegen signalisieren den unproduktiven Overhead, der bei der Synchronisation entsteht. Der Prozessmanager macht nebst seinem Synchronisationsfenster nichts, weswegen seine Spalten grau markiert sind. Eine mögliche Verbesserung wäre die Auslastung des Prozessmanagers mit zusätzlicher Berechnungsarbeit, wie sie die Berechnungsprozesse durchführen.

## 4.8 Anwendung

Die Anwendung zur Steuerung der Applikation, welche mit Unity entwickelt wurde, bietet diverse Möglichkeiten zur Darstellung, das Umschalten von Spielmodi und das Erzeugen eines Spielstands zwecks Debugging/Testing. Die Funktionalitäten werden nachfolgend beschrieben.



**Halo:** Der Halo um die Kugel kann über die Taste **H** angezeigt oder versteckt werden.

**Löcher und Banden:** Die Löcher und Banden können über die Taste **T** angezeigt oder versteckt werden, siehe Abbildung 4.42.

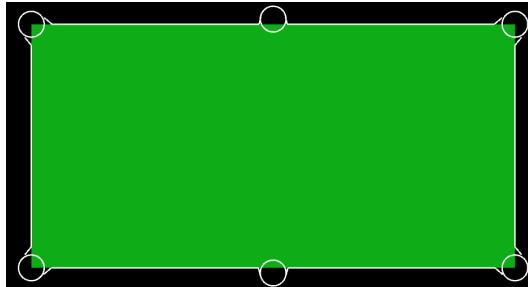


Abbildung 4.42: Anzeige des Tisches mit Banden und Löchern



**Punktpfad:** Über die Taste **P** kann ein Pfad bestehend aus Punkten eingeblendet werden. Diese Punkte werden jeweils an einer Kugelposition für zwei Sekunden angezeigt. Dies ermöglicht die visuelle Pfadverfolgung einer Kugel.

**Livemode:** Die Livedetektion kann über die Taste **L** ein- oder ausgeschaltet werden.

**Alle Suchergebnisse anzeigen:** Über die Taste **X** werden alle Suchergebnisse nacheinander visualisiert.

**Infinity-Mode:** Über die Taste **M** kann der Infinity-Modus ein- oder ausgeschaltet werden. Dieser ist standardmäßig deaktiviert. Im Infinity-Modus werden gewisse Regeln des Spiels ignoriert. Es wird erkannt, wann der Spielstand stabil bleibt, sich demnach nichts bewegt. In diesem Fall wird automatisch eine Suche ausgelöst und das Resultat angezeigt. Dieser Modus hat die Intention aufzuzeigen, wie man eine Kugel anspielen muss und hat auch als Ziel möglichst wenig direkte Interaktion mit dem Computersystem zu erfordern.

**Tiefensuche:** Die Suche über mehrere Stöße kann über die Taste **Y** ein- und ausgeschaltet werden.

**Animationsfunktionalitäten:** Die Animation eines Stosses wird automatisch nach der Suche ausgelöst. Eine pausierte Animation ist in Abbildung 4.43 ersichtlich. Mittels der Tasten **Pfeil links** und **Pfeil rechts** kann innerhalb des Stosses nach vorne oder zurück gespult werden. Mit den Tasten **Pfeil oben** und **Pfeil unten** kann zwischen verschiedenen Stößen gewechselt werden. Es kann gut sein, dass der erste Stoß jeweils derselbe ist, Unterschiede können erst in tieferen Stößen auftreten. Mit der **Leertaste** kann eine Animation gestoppt und mit der **Returntaste** auf Anfang zurückgesetzt werden.

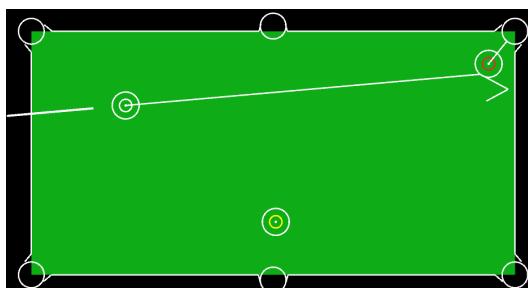


Abbildung 4.43: Unity Animationsdarstellung



**Anzeige der animierten Kugeln:** Die Kugeln können über die Taste **K** entweder als Umrandung, volle Kugel oder unsichtbar dargestellt werden.



**Informationsausgabe:** Zu Entwicklungszwecken kann die Position und die ID mit der Taste **D** angezeigt oder versteckt werden.

**Spielstandeingabe:** Ebenfalls zu Entwicklungszwecken kann das Erstellen eines Spielstands über die Taste **S** erfolgen. In diesem Modus kann über das Scroll-Rad der Maus der Typ der zu platzierenden Kugel ausgewählt werden. Mit einem linken Mausklick wird die Kugel platziert, mit einem rechten Mausklick auf eine Kugel wird diese entfernt. Ein Status kann auch textuell beschrieben eingegeben werden. Dazu kann im Spielstanderstellmodus die Taste **T** gedrückt werden. Es erscheint ein Eingabefeld. Das Format lautet: ID, TYP, X-Koordinate, Y-Koordinate  
ID, TYP, X-Koordinate, Y-Koordinate

Die Erfassung einer weissen und roten Kugel wird nachfolgend beschrieben:

WHITE0, WHITE, -300, 0

RED1, RED, -150, -300

Sobald die Beschreibung erfasst wurde, kann die Eingabe mit **Enter** bestätigt werden. Es wird der normale Modus angezeigt mit dem erstellten Spielstand.

## 4.9 Infinity-Modus

Der Infinity-Modus soll es dem Spieler ermöglichen, einen Vorschlag für einen Stoss zu erhalten, einen Stoss auszuführen und anschliessend auf Basis des dadurch entstandenen Spielstandes einen neuen Vorschlag zu erhalten.

Dies bedingt die Klassifikation des Spielstandes in stabil und instabil. Ein Spielstand ist instabil, sofern sich noch Kugeln bewegen oder der Spieler gerade dabei ist, einen Stoss durchzuführen. Der Spielstand gilt als stabil, wenn die Kugeln stehen bleiben. Sofern sich der Spielstand zu stabil ändern, wird eine Suche nach dem optimalen Stoss ausgeführt, und das Resultat wird dem Spieler auf dem Tisch angezeigt. Währenddessen, dass der Spielstand instabil ist, soll keine erneute Suche ausgelöst werden, damit der Spieler die Chance hat, den Vorschlag zu prüfen und gemäss angezeigter Animation auszuführen.

Die Detektion des aktuellen Spielstandes enthält über die Zeit betrachtet ein gewisses zufälliges Rauschen. Dadurch verändern sich die Positionen aller Kugeln auch bei komplettem Stillstand leicht. Ausserdem werden manchmal Kugeln erkannt wo keine Kugeln liegen, oder es werden Kugeln nicht erkannt. Dieser Ungewissheit muss Rechnung getragen werden. Dieser Modus baut hierzu auf der in Abschnitt 4.4 beschriebenen Tracking und Positionsstabilisierung auf.

Die Bedingungen, dass der Spielstand als stabil gilt, sind nachfolgend aufgeführt und werden in den nächsten Abschnitten erläutert.

1. Tracking des Spielballs: Der Spielball wurde gefunden.
2. Bewegungserkennung: Keine Kugel ist als *bewegt* eingestuft.
3. Verlust von Kugeln: Keine Kugel ging seit der letzten Detektion des Spielstandes verloren.
4. Neue Kugeln: Es ist keine Kugel zum Spielstand hinzugefügt worden.
5. Robustheit: Die aktuelle Anzahl detekтирter Kugeln entspricht der Anzahl über längere Zeit getrackter Kugeln.
6. Zeitliche Stabilität: Der Spielstand muss konsequent über einige Frames als stabil eingestuft werden.

### 4.9.1 Tracking des Spielballs

Der Spielball ist von allen Kugeln einzigartig, weil in einem regelkonformen Spiel höchstens ein Spielball auf dem Tisch liegt. Dieses Wissen kann daher ausgenutzt werden, um diesen wiederzufinden. Wenn der Spielball nicht auf dem Tisch liegt, weil er beispielsweise eingelocht wurde, dann soll der Spielstand instabil sein.

Die in Abschnitt 4.3 beschriebene Detektion führt manchmal zu fälschlicherweise erkannten Kugeln. So kann es passieren, dass mehrere weisse Kugeln auf dem Tisch erkannt werden. Daher muss der echte Spielball gefunden werden.

Im Fall dass eine einzige weisse Kugel erkannt wurde, wird diese als Spielball angenommen. Sofern es mehrere weisse Kugeln gibt wird diejenige gesucht, welche der zuletzt bekannten Position des Spielballs am nächsten ist. Falls sich diese nicht zu weit entfernt befindet, dann wird diese Kugel als Spielball angenommen und die anderen weissen Kugeln werden ignoriert. Falls keine naheliegende Kugel gefunden wurde, wird angenommen, dass der Spielball nicht auf dem Tisch liegt. Das Gleiche passiert, sofern keine weisse Kugel erkannt wurde.

## 4.9.2 Bewegungserkennung

Der Spielstand darf erst als stabil gelten, wenn keine Kugeln mehr in Bewegung sind. Da alle Kugeln bereits getrackt werden, kann für jede erfolgreich getrackte Kugel eine History analog derjenigen für die Kugelpositionen aus Abschnitt 4.4 geführt werden. So wird für jede getrackte Kugel die Bewegung der detektierten Kugelposition zwischen zwei aufeinanderfolgenden Frames über  $M$  Frames gespeichert. Mithilfe dieser Bewegungsvektoren wird anschliessend ein Durchschnitt gebildet, um den durchschnittlichen Bewegungsvektor über die letzten  $M$  Frames zu erhalten. Der Betrag dieses Vektors kann anschliessend genutzt werden, um einzuschätzen, ob die Kugel in Bewegung ist oder nicht. Wenn eine Kugel in Bewegung versetzt wird, steigt dieser Durchschnitt zunächst und sinkt wieder sobald die Kugel ausrollt. Eine Kugel gilt als bewegt, wenn der Betrag des durchschnittlichen Bewegungsvektors über dem Grenzwert von  $0.5mm$  liegt.

## 4.9.3 Verlust von Kugeln

Sofern eine Kugel eingelocht wird, verändert sich die Gesamtanzahl der Kugeln und der Spielstand ist für eine kurze Zeit instabil. Es reicht allerdings nicht aus, die Anzahl detekтирter Kugeln zweier aufeinanderfolgender Frames zu vergleichen, da Detektionsfehler diesen Vergleich verfälschen können. Wenn beispielsweise in Frame  $T_{t-1}$  eine Kugel zu viel detektiert und diese Kugel in Frame  $T_t$  nicht mehr detektiert wird, dann würde diese Kugel in Frame  $T_t$  als verloren eingestuft. Da diese Detektionsfehler oft nur auf einzelnen Frames auftauchen, gilt eine Kugel erst als verloren, wenn diese zuvor  $O$  Frames erfolgreich getrackt werden konnte.

## 4.9.4 Neue Kugeln

Sofern eine Kugel auf dem Tisch platziert wird, weil beispielsweise eine der farbigen Kugeln neu platziert wird, dann muss diese Änderung den Spielstand zu instabil ändern. Sobald die neue Kugel anschliessend eine gewisse Zeit lang auf dem Tisch liegt und nicht mehr als *neu* gilt, kann der Spielstand wieder zu stabil wechseln. Damit auch hier fälschlicherweise detektierte Kugeln nicht zu Fehlentscheidungen führen, werden hierzu nur Kugeln betrachtet, welche  $N$  Frames lang erfolgreich getrackt wurden. Anschliessend kann die Anzahl der in Frame  $F_t$  erfolgreich getrackten Kugeln mit derjenigen aus Frame  $F_{t-1}$  verglichen werden. Sofern diese Differenz positiv ist, wurden Kugeln dem Spielstand hinzugefügt und der Spielstand muss als instabil markiert werden.

## 4.9.5 Robustheit

Damit Detektionsfehler nicht in der Suche nach Stößen berücksichtigt werden, muss der Spielstand als instabil gelten, solange sich fälschlicherweise detektierte Kugeln auf dem Tisch befinden. Daher muss die Anzahl aktuell detekтирter Kugeln mit der Anzahl Kugeln übereinstimmen, welche über  $N$  Frames erfolgreich getrackt wurden. Dadurch ist sichergestellt, dass der Spielstand erst stabil wird, wenn alle tatsächlich vorhandenen Kugeln lange genug getrackt wurden und keine Fehldetections vorhanden sind.

## 4.9.6 Zeitliche Stabilität

Der Wechsel zum Status stabil sollte möglichst nur dann auftreten, wenn der Spielstand vollkommen ruhig ist. Außerdem darf es nicht passieren, dass der Status zu volatil ist. Es wäre problematisch wenn der Status innerhalb weniger Frames mehrfach zwischen stabil und instabil wechselt, denn bei jedem Wechsel zum Status stabil wird eine Suche ausgelöst und dem Spieler ein Vorschlag angezeigt. Würde der Status nach Anzeige des Vorschlags auf instabil und dann auf stabil ändern, würde das eine erneute Suche auslösen und der bisherige Vorschlag würde nicht mehr angezeigt, bis der neue Vorschlag erscheint. Daher wird der Status erst tatsächlich als stabil ausgewiesen, wenn er über mindestens  $P$  Frames als stabil erkannt wurde.

Die oben beschriebenen Bedingungen dienen dem Ziel des Infinity-Modus, dem Spieler zu geeigneter Zeit einen Vorschlag für einen Stoss zu machen und anzusehen, bis der Spieler einen Stoss ausgeführt hat und ein neuer Vorschlag erwartet wird.

## 4.10 Verwandte Arbeiten

Im Umfeld dieser Arbeit existieren bereits einige Lösungen. Es gibt diverse Ansätze, welche unter anderem auch Roboter miteinbeziehen[Mül21] oder sich ganz einer AI eines optimalen Stosses ohne direkten Realitätsbezug widmen[Smi06]. Diese Arbeiten werden nicht direkt als Vergleich hinzugezogen, es wird sich eher auf praktische Lösungen mit ähnlichem Funktionsumfang konzentriert.

### 4.10.1 Automatic pool stick vs. strangers

Auf dem Youtube-Kanal „Stuff Made Here“ werden regelmässig beeindruckende Projekte veröffentlicht, so befasst sich eine Arbeit ebenfalls mit der Thematik des Billardspiels, wo es um das Bauen eines Billardqueues geht, der einen über eine Kamera detektierten Stoss ausführen kann[Her]. Die Herangehensweise ist sehr ähnlich, so werden für die Detektion der Kugeln ebenfalls Aruco-Marker eingesetzt und die Suche nach einem optimalen Stoss geschieht über eine Diskretisierung des Suchraums, in der von jedem Loch aus alle Möglichkeiten berechnet und bewertet werden. Zusätzlich wurde, wie bereits erwähnt, ein automatischer Queue gebaut, der den Stoss mit dem korrekten vertikalen Versatz und der passenden Geschwindigkeit ausführen kann. Dieser Queue lässt sich auch über das Internet steuern, sodass ein Spieler teilnehmen kann, der gar nicht physisch vor Ort ist.

Die vorliegende Arbeit beinhaltet hingegen nicht nur die Anzeige des Suchresultats wie bei „Stuff Made Here“, sondern auch deren Simulation. Es werden die Wege aller Kugeln bis zum Stillstand oder Einlochen auf Basis der physikalischen Eigenschaften des Tisches sowie der Kugeln korrekt animiert. Dadurch ist die Konsequenz des Stosses bekannt, was wiederum die Möglichkeit der Berücksichtigung mehrerer aufeinanderfolgender Stösse eröffnet hat. Diese Ergänzung wird als mögliche weitere Arbeit bei „Stuff Made Here“ am Ende des Videos erwähnt.

### 4.10.2 Pool Live AR

Dieses Projekt befasst sich nicht mit der Suche und Visualisierung eines optimalen Stosses, sondern mit der Detektion des Queues und der Anzeige des resultierenden Weges der weissen Kugel[AR]. Zudem wurden einige Effekte/Animationen eingebaut, die das Spielerlebnis verbessern sollen, z.B. wenn die weisse Kugel angestossen wird. Das System kann den Weg der weissen Kugel mit einem konstanten Fehlerwinkel und einer bestimmten Anzahl an Bandenreflektionen anzeigen. Kollisionen mit anderen Kugeln wie auch deren Konsequenzen für den Spielstand werden nicht erkannt. Das System hat demnach vermutlich keine Simulation, sondern detektiert die 2D-Orientierung des Queues und reflektiert die daraus entstehende Linie des Spielballs an den Banden.

### 4.10.3 Innovationsgrad

Es gibt bereits viele Arbeiten die sich thematisch mit derselben Problematik befassen, sei dies theoretischer oder praktischer Natur. Als innovativ kann die animierte Darstellung der Konsequenz eines Stosses wie auch die Suche über mehrere Spielstände genannt werden. Ausserdem wurde mit dem Infinity-Modus eine interaktive Möglichkeit geschaffen, um den Spieler mit Stossvorschlägen zu unterstützen. Dem Spieler wird die benötigte Stärke eines Stosses angezeigt, wodurch dieser lernt, dass viele Stösse im Billard nicht viel Kraft benötigen, um das Ziel zu erreichen. Die Suche über mehrere Spielstände verdeutlicht dem Spieler, dass in Snooker die Platzierung des Spielballs essentiell ist und fördert dessen Weitsicht.

Des Weiteren bietet die Interpretation der Kugeln in Modellkoordinaten Spielraum, der weitere Innovationen wie z.B. eine Replay-Funktionalität oder andere Verarbeitungen des Spielstandes zulassen.

Physikalische Eigenschaften wie Topspin und dass die Billardkugeln zunächst gleiten und nach einer gewissen Zeit rollen wurde in der Simulation berücksichtigt. Durch die Berücksichtigung von Sidespin oder Verbesserungen in der Simulation des Rollverhalten könnte sich das Projekt noch weiter von der Konkurrenz abheben.

# 5 Resultate

Dieses Kapitel beinhaltet die Aufführung aller Ergebnisse der Arbeit. Dies betrifft insbesondere die Fortsetzung der Genauigkeitsanalyse der Kugeldetektion aus der Vorarbeit[Luk21d], die Genauigkeitsanalyse der Klassifikation wie auch eine Aufstellung einiger Spielstände, deren Suchresultate und verwendete Berechnungszeiten für die einfache direkte wie auch erweiterte Suche. Zudem wird die Berechnung des Rollreibungskoeffizienten und der Energieverluste bei Ereignissen behandelt.

## 5.1 Klassifikation

Für die Beurteilung der Klassifikation, die in Kapitel 4.5 beschrieben wurde, wird eine Wahrheitsmatrix[Wik21b] (engl. *confusion matrix*) verwendet. Daraus können Metriken berechnet werden, welche eine Aussage über die Qualität der Klassifikation erlauben.

In Tabelle 5.1 ist die Wahrheitsmatrix der Testdaten für die Klassifikation aufgeführt. Es wurden 32 Testbilder erstellt, wobei jeweils alle farbigen und einige rote Kugeln auf jedem Bild abgebildet sind. Dies führt dazu, dass es mehr Testdaten für die roten Kugeln gibt als für die anderen farbigen Kugeln. In den Testbildern sind keine Kugeln mit der Klasse *unknown* vorhanden.

Aus der Wahrheitsmatrix ist ersichtlich, dass braune und rote Kugeln teilweise verwechselt werden. Tatsächlich führt Die restlichen Kugelfarben werden sehr gut klassifiziert, lediglich ein Bild des Spielballs wird mit einer pinken Kugel verwechselt. Keiner der Kugeln wurde die Klasse *unknown* zugewiesen.

Klassifizierte Farbe	Wahre Farbe									
	BROWN	PINK	RED	BLACK	YELLOW	WHITE	BLUE	GREEN	UNKNOWN	
BROWN	29	0	11	0	0	0	0	0	0	0
PINK	0	32	0	0	0	1	0	0	0	0
RED	3	0	213	0	0	0	0	0	0	0
BLACK	0	0	0	32	0	0	0	0	0	0
YELLOW	0	0	0	0	32	0	0	0	0	0
WHITE	0	0	0	0	0	31	0	0	0	0
BLUE	0	0	0	0	0	0	32	0	0	0
GREEN	0	0	0	0	0	0	0	32	0	0
UNKNOWN	0	0	0	0	0	0	0	0	0	0

Tabelle 5.1: Ergebnisse Klassifikation der Kugelfarben

## 5.2 Detektion

Zur Überprüfung der Genauigkeit der Kugeldetektion wurden 50 Testbilder von verschiedenen Spielständen aufgenommen und von Hand die Positionen der Kugeln in Pixelkoordinaten angegeben. Die Pixelkoordinaten wurden anschliessend in Modellkoordinaten umgewandelt [Luk21c]. Diese Referenzdaten können mit dem Resultat des Detektionsalgorithmus verglichen werden, indem dieser die Kugeln auf denselben Bildern detektiert, auf dieselbe Weise in Modellkoordinaten umwandelt und die Positionen mit den Referenzdaten verglichen werden. Die Distanzen zwischen den Kugelpositionen in den Referenzdaten und in der Detektion können anschliessend statistisch ausgewertet werden.

In den 50 Referenzbildern sind insgesamt 835 Kugelpositionen hinterlegt, in Tabelle 5.2 sind einige statistische Kennzahlen aus diesem Vergleich aufgeführt. Der Median liegt bei ca. 2.33mm und 50% der Detektionen haben einen Positionsfehler zwischen ca. 0mm - 3.7mm. Die maximale Abweichung zwischen Referenzdaten und Detektion liegt bei 8.35mm.

Diese Vergleichsmethode enthält an sich bereits einen Fehlerbereich, welcher der Auflösung des Bildes geschuldet ist. Eine genaue Beschreibung ist in [Luk21e] enthalten, hier werden die wichtigsten Erkenntnisse erneut aufgeführt. Die Ausmasse

Bezeichnung	Wert
Anzahl Kugeln	835
Minimum	0.000004mm
Unteres Quartil	0.000601mm
Median	2.329217mm
Oberes Quartil	3.680264mm
Maximum	8.3566mm

Tabelle 5.2: Statistische Zahlen zu den Distanzen zwischen den Kugelpositionen der Referenzdaten und den detektierten Kugelpositionen.

des Spielbereichs des Billardtisches betragen 1881mm mal 943mm, die Auflösung des Bildes beträgt 1280x720 Pixel. Der Spielbereich füllt nicht die gesamte Bildauflösung, sondern lediglich 1118x565 Pixel. Ein einzelnes Pixel des Bildes entspricht damit einer Fläche von 1.681mm mal 1.668mm. Der Detektionsalgorithmus detektiert die Kugelpositionen subpixelgenau. Die maximale Abweichung unter der Annahme, dass das korrekte Pixel in den Referenzdaten angegeben wurde, beträgt 1.18406mm.

Die Annahme, dass in den Referenzdaten das korrekte Pixel angegeben wurde, ist selbstverständlich höchst unwahrscheinlich. In Abbildung 5.1 sind verschiedene Fälle aufgeführt, um eine Aussage darüber zu machen, wie sich ein Fehler in den Referenzdaten auf deren Genauigkeit auswirkt.

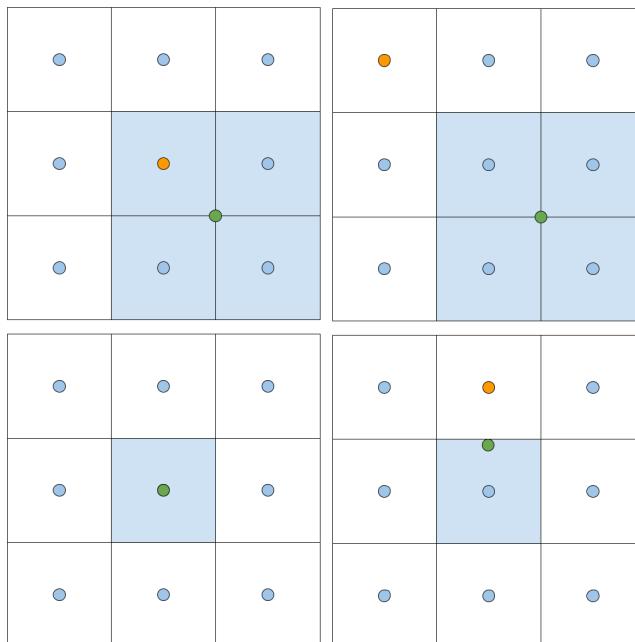


Abbildung 5.1: Extremsituationen zum Vergleich der Referenzposition und der wahren Position des Kugelmittelpunktes. Abgebildet ist ein Bildausschnitt mit 9 Pixeln. Die blauen Punkte sind die Pixelzentren, grün ist die wahre Position (subpixelgenau), orange ist die Referenzposition aus den Referenzdaten. Die Pixel rund um die wahre Position sind blau hinterlegt. Oben-Links: Der maximale Fehler ist eine halbe Pixeldiagonale, wenn die wahre Position zwischen vier Pixeln liegt und die Referenzposition einem der Pixel rund um die tatsächliche Position entspricht. Unten-Links: Der minimale Fehler ist 0, wenn die wahre Position genau dem Pixelzentrum entspricht und dieses Pixel in den Referenzdaten ausgewählt wurde. Oben-Rechts: Der maximale Fehler ist 1.5 Pixeldiagonalen, wenn die wahre Position zwischen vier Pixeln liegt und die Referenzposition um ein Pixel daneben ist. Unten-Rechts: Der minimale Fehler ist eine halbe Pixelbreite/-höhe wenn die wahre Position innerhalb des zentralen Pixels liegt und die Referenzposition um ein Pixel falsch daneben ist.

In Abschnitt 4.4 wurde beschrieben, dass in der Live-Detektion eine Stabilisierung der detektierten Positionen über die Zeit durchgeführt wird. In diesem Kapitel wurden nur Bilder für den Vergleich zwischen Detektion und Realität verwendet, wodurch es sich um Momentaufnahmen handelt. Es kann sein, dass die Stabilisierung in gewissen Fällen die detektierten Positionen verbessert, weil Positionsfehler in einzelnen Bildern geglättet werden.

Die implementierte Live-Detektion gilt es nachfolgend ebenfalls zu überprüfen.

Ein Problem der Live-Detektion ist, dass wenn ein Spieler mit der Hand in das Bild greift, um u.a. einen Stoss durchzuführen, dann werden bei der Hand ebenfalls Kugeln detektiert, siehe Abbildung 5.2. Dies ist darauf zurückzuführen, dass die Detektion eine Segmentierung des HSV-Farbraumes durchführt[Luk21b] und anschliessend darauf den Circle Hough transform[Wik21a] ausführt. Dadurch werden Kleider oder Haut ebenfalls als Bereiche erkannt, wo eine Kugel sein könnte. Der Circle Hough transform findet anschliessend aufgrund von Kanten in diesen Bereichen Kreise, welche dann als detektierte Kugeln übernommen werden.

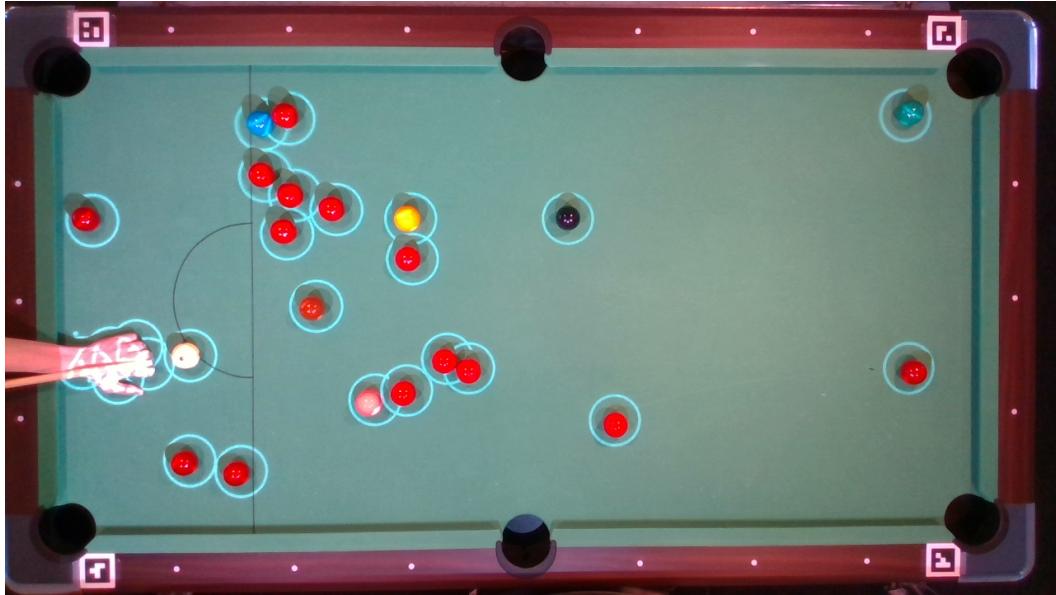


Abbildung 5.2: Fälschlicherweise detektierte Kugeln auf der Hand

Sofern die Arme und Hände des Spielers segmentiert werden könnten, wäre es möglich diese Fehler in der Detektion zu verhindern. Dazu würden die segmentierten Regionen, wo eine Hand oder ein Arm abgebildet ist, in der Detektion ignoriert. Diese Segmentation ist allerdings schwierig durchzuführen, da diese stark von Hautfarbe und Bekleidung abhängt. Es wäre hier denkbar, ein vortrainiertes neuronales Netzwerk zu verwenden, welches eine Handsegmentation oder -detektion machen kann.

Das in Abschnitt 4.4 beschriebene Tracking zur Stabilisierung der Kugelpositionen hat zu einer deutlichen visuellen Verbesserung geführt. Sofern der Spielstand ruhig ist, also keine Kugeln in Bewegung sind, bewegen sich die projizierten Kreise rund um die detektierte Position kaum noch. Ohne Tracking ist das Rauschen in der detektierten Position anhand der bewegenden Kreise sichtbar.

Diese visuelle Verbesserung kann auch statistisch quantifiziert werden, indem die Detektion auf einem Video eines ruhigen Spielstandes durchgeführt wird. Die Detektion kann auf jedem einzelnen Frame des Videos durchgeführt werden, und die detektierte Position mit derjenigen im vorherigen Frame verglichen werden. Die Distanz zwischen der detektierten Position in Frame  $F_{t-1}$  und derjenigen in Frame  $F_t$  bildet anschliessend eine Messung, die gesammelt wird.

Auf den gesammelten Distanzen können anschliessend statistische Kennzahlen berechnet werden. Diese Messungen wurden mit und ohne Stabilisierung der Kugelpositionen durchgeführt, die Resultate sind in Tabelle 5.3 aufgeführt. Die Messungen basieren auf einem Videoausschnitt von 178 Frames eines Videos mit einem Spielstand wo 21 Kugeln stillstehen. Die Messung wurde erst nach 30 Frames gestartet, damit die Messung mit eingeschalteter Stabilisierung bereits einige Daten sammeln konnte. Während dieser Aufwärmphase wäre die Detektion mit eingeschalteter Stabilisierung anfällig auf Rauschen in der Detektion, wie es die Detektion ohne Stabilisierung ist. Durch diesen Vorlauf von 30 Frames beträgt der gemessene Videoausschnitt 148 Frames. Dies resultiert in  $148 \cdot 21 = 3108$  Messwerten. Aus Tabelle 5.3 wird klar, dass die Stabilisierung die Bewegung der detektierten Kugelpositionen deutlich verringern konnte.

In Abbildung 5.3 sind die Daten aus Tabelle 5.3 in zwei Box-Plot-Diagrammen visualisiert.

### 5.3 Bestimmen des Rollreibungskoeffizienten

Rollt eine Kugel einen bestimmten Weg über den Billardtisch, so entsteht eine Reibungskraft, welche der Geschwindigkeit entgegengesetzt wirkt. Die Kraft kann als negative Beschleunigung angegeben werden, wobei diese von einem Rollreibungs-

Bezeichnung	Stabilisierung ausgeschaltet	Stabilisierung eingeschaltet
Anzahl Messwerte	3108	3108
Minimum	0mm	0mm
Unteres Quartil $Q_{0.25}$	1.638916mm	0.081972mm
Median $Q_{0.5}$	2.359856mm	0.129632mm
Oberes Quartil $Q_{0.75}$	3.714067mm	0.222898mm
Quartilabstand $Q_{0.75} - Q_{0.25}$	2.075151mm	0.140926mm
Maximum	13.590104mm	4.043161mm
Summe aller Messungen	7654.91mm	588.849mm

Tabelle 5.3: Statistische Zahlen zu der Verteilung der Veränderung der detektierten Kugelpositionen über eine Dauer von 148 Video-Frames mit und ohne Stabilisierung. Bei einem Messwert handelt es sich um die Distanz zwischen der detektierten Kugelposition in Frame  $F_{t-1}$  und Frame  $F_t$  für eine Kugel.

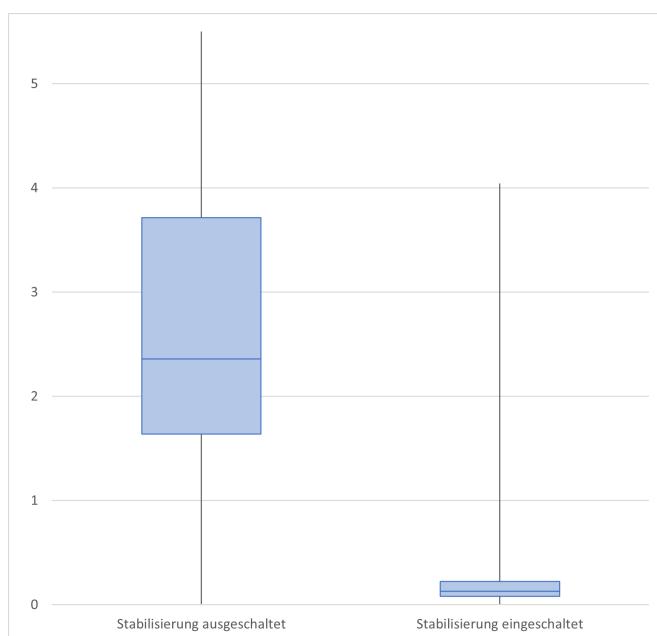


Abbildung 5.3: Box-Plot-Diagramme zu der Bewegung der Kugelpositionen mit und ohne Stabilisierung gemäss Tabelle 5.3. Der Wertebereich der Y-Achse wurde auf 5 beschränkt, damit das rechte Diagramm ersichtlich ist. Die Antenne des linken Diagrammes geht bis zum Maximum von 13.59mm.

koeffizienten  $c_R$  abhängig ist. Das Finden dieses Koeffizienten wird in diesem Kapitel beschrieben. Die Herleitung und theoretischen Überlegungen finden sich in Kapitel 8.11.

Abbildung 5.4 veranschaulicht den Versuchsaufbau zur Bestimmung des Rollreibungskoeffizienten. Wie in Kapitel 8.11 erläutert, muss für die Rampe ein Material mit möglichst geringer Reibung verwendet werden, weswegen die Wahl auf Glas fiel. Weiterhin wird der Weg der Kugel geführt, da sie eine möglichst gerade Bahn rollen muss. Die so entstehende Reibung an der Bahn wird ebenfalls vernachlässigt.

Es wurden zwei Versuche durchgeführt mit unterschiedlicher Höhe der Rampe. Die Ergebnisse dieser Messungen wie deren Median sind in der Tabelle 5.4 aufgeführt.

Anhand des Ablaufs aus Kapitel 8.11 können die Reibungskoeffizienten mithilfe der Daten aus Tabelle 5.4 bestimmt werden. Die Resultate sind in Tabelle 5.5. Daraus wird der Mittelwert berechnet, welcher das Resultat bildet.

## 5.4 Energieverlust

In den Formeln für die Berechnung der Initialgeschwindigkeit wie auch die Auswirkung bei einer auftretenden Kugelkollision wurde ein Faktor  $E_V$ , angegeben in Prozent, berücksichtigt, welcher den Energieverlust modelliert. Es gilt durch ein experimentelles Verfahren diesen Wert für die jeweilige Art zu bestimmen.



Abbildung 5.4: Versuchsaufbau zur Bestimmung des Reibungskoeffizienten

Höhe [mm]	13	19
Distanz [mm]		
957	1228	
949	1253	
926	1263	
914	1280	
931	1281	
932	1296	
941	1256	
926	1295	
931	1295	
941	1304	
943	1314	
937	1316	
947	1284	
949	1303	
947	1319	
-	1320	
<b>Median</b>	<b>941</b>	<b>1295</b>

Tabelle 5.4: Ergebnisse Distanzmessung einer rollenden Kugel

#### 5.4.1 Energieverlust bei Kugelkollision

Der Energieverlust wird sich ca. im Bereich von 5%[And07] bewegen, demnach wird  $E_v = 0.05$  gewählt.

#### 5.4.2 Energieverlust bei Bandenkollision

Es wird angenommen, dass der Energieverlust bei einer Kollision einer Kugel mit der Bande ca. 2% beträgt[MJP10]. An anderer Stelle beträgt er ca. 38%[Jac95].

### 5.5 Suche

Um die Resultate der Suche zu bewerten werden nachfolgend Situationen aufgestellt und die von der Suche vorgeschlagenen Stöße untersucht.

Die erste Situation ist in Abbildung 5.5 ersichtlich. Der Spielball ist auf der linken Seite des Tisches und es liegen einige rote Kugeln auf dem Tisch verstreut. Die Kugeln 1, 2 und 3 sind nahe bei den Löchern unten-links, oben-rechts resp. oben-zentral. Kugel 4 ist in der Nähe des Spielballs, die Kugel 5 ist nahe dem Tischzenturm aufgestellt.

Strecke [mm]	Reibungskoeffizient
941	0.0138151
1295	0.0146718
<b>Mittelwert</b>	<b>0.0142435</b>

Tabelle 5.5: Reibungskoeffizienten über Strecke

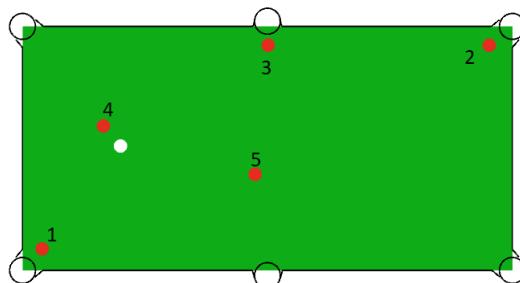


Abbildung 5.5: Situation 1: Einige verstreute Kugeln

Nach Betrachten dieser Situation sind einige mögliche Stöße denkbar:

1. Kugel 1 ins Loch unten-links.
2. Kugel 2 ins Loch oben-rechts.
3. Kugel 3 ins Loch oben-zentral.
4. Kugel 4 ins Loch oben-links.
5. Kugel 5 ins Loch unten-rechts.
6. Kugel 5 ins Loch unten-zentral.

In Abbildung 5.6 sind die von der Suche gefundenen Stöße in aufsteigender Schwierigkeit abgebildet. Mit den ersichtlichen weissen Linien auf jeder Abbildung ist der Pfad jeder Kugel, nicht nur des Spielballs, aufgezeichnet. Diese voraussichtlichen Pfade werden nach der Suche mithilfe der Simulation des Stosses eingezeichnet.

Die meisten der zuvor beschriebenen denkbaren Stöße wurden tatsächlich gefunden. Der Stoß 1 wurde als der einfachste Stoß bewertet, weil die gesamte zurückzulegende Distanz klein ist, der Pfad der beteiligten Kugeln insgesamt sehr geradlinig ist und weil die Kugel 1 sehr nahe am Loch unten-links liegt. Der zweiteinfachste Stoß 2 zeigt eine grössere Distanz, welche der Spielball zurücklegen muss und daher erfordert der Stoß auch eine höhere Startgeschwindigkeit des Spielballs wodurch dieser als schwieriger eingestuft wird als Stoß 1. Dafür ist die Kugel 2 sehr nahe am Loch oben-rechts, wodurch die Kugel nicht mit absoluter Genauigkeit getroffen werden muss, damit diese den erwünschten Pfad nicht so weit verlässt, dass sie das Ziel verfehlt.

Im Stoß 3 ist die Gesamtdistanz klein, allerdings ist die Distanz der Kugel 4 zum Spielball kleiner als die Distanz zum Loch. Dadurch führt ein kleiner Fehler beim Treffen der Kugel zu einer grösseren Abweichung des Pfades. In diesem Beispiel könnte dieser Fehler allerdings relativ klein sein, da der Pfad geradlinig ist und der Spielball die Kugel nicht in einem Winkel treffen muss.

Stoß 2 wurde besser als Stoß 3 bewertet, weil bei Stoß 2 die Kugel näher am Loch liegt als bei Stoß 3. Die anderen Bewertungskriterien Distanz und Winkel sind bei Stoß 3 besser als bei Stoß 2. Ob diese Rangfolge objektiv korrekt ist, ist schwierig zu beurteilen.

Der Stoß 5 wird als schwieriger bewertet, weil der Winkel, in dem der Spielball die Kugel 3 treffen muss ca  $45^\circ$  ist. Im Gegensatz dazu ist die Distanz der Kugel zum Loch sehr klein und die Gesamtdistanz ist relativ klein, was diesen Stoß wiederum einfacher macht. Trotzdem hat in diesem Fall der Winkel die Bewertung so stark reduziert, dass Stoß 4 trotzdem als einfacher eingestuft wird.

Stoß 6 zeigt die Idee, Kugel 5 ins Loch unten-zentral zu spielen. Dazu muss diese in einem sehr grossen Winkel angespielt werden, was die Treffgenauigkeit reduziert und eine erhöhte Startgeschwindigkeit des Spielballs erfordert. Die weiteren weissen Linien zeigen den weiteren Verlauf der weissen Kugel, welche noch weitere Kugeln anstösst und eine weitere rote Kugel ins Loch spielt, wodurch 2 Punkte statt einem Punkt gemacht werden. Es wirkt unwahrscheinlich, dass die zweite rote Kugel tatsächlich ins Loch gespielt werden kann, da der Pfad der weissen Kugel sehr lange ist und zwei Bandenkollisionen

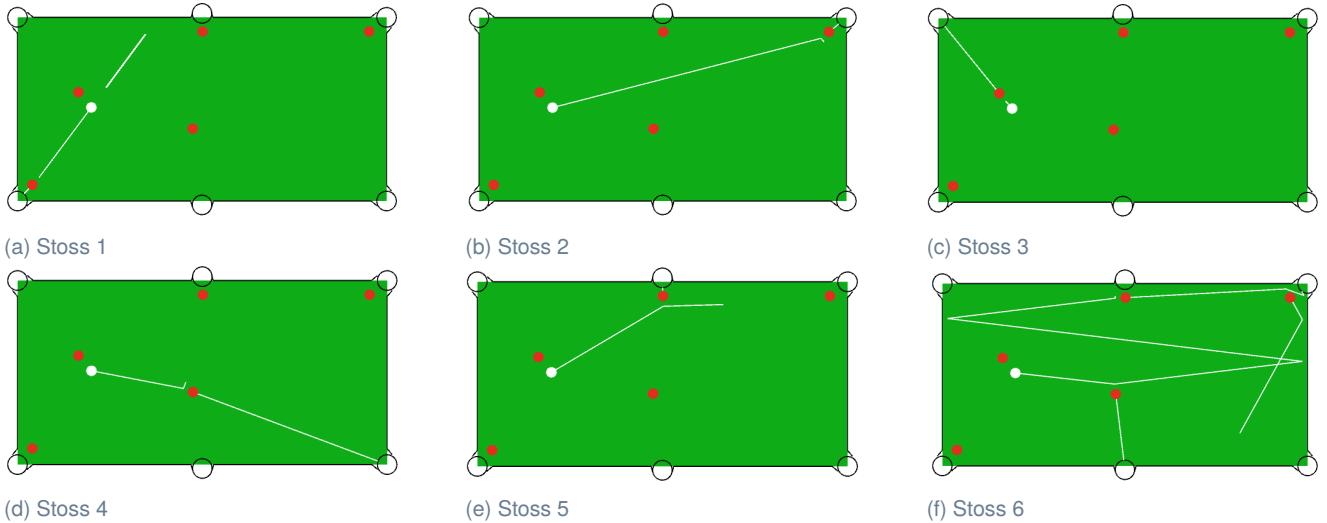


Abbildung 5.6: Gefundene Stöße zu Situation 1 nach bewerteter Schwierigkeit aufsteigend

und zwei Kugelkollisionen enthält, bevor diese ins Loch rollt. Allerdings sorgt der grosse Winkel, in dem die Kugel 5 angespielt werden muss dafür, dass dieser Stoß als sehr schwierig bewertet wird, was den Nutzen von einem zusätzlichen Punkt überwiegt.

Aus den obigen Resultaten ist ersichtlich, dass die Suche diejenigen Stöße vorschlägt, welche nach Studium des Spielstandes möglich erscheinen. Zudem berücksichtigt die Bewertung der Schwierigkeit eines Stosses die gesamte Distanz, die Winkel und die Nähe der einzulochenden Kugel zum Loch.

## 5.6 Vergleich Simulation und Realität

Um die Realitätsnähe der Simulation zu quantifizieren, wird der Ansatz über die Analyse von Aufnahmen diverser real durchgeföhrter Stöße verfolgt. Dabei ist die Geschwindigkeit, welche die Kugel zu Beginn durch den Queue erfährt, unbekannt. Diese wird in einem ersten Schritt eruiert. Bekannt dabei ist die Zeit sowie die Distanz, welche über die Selektion einzelner Frames berechnet werden. Die Zeit ergibt sich durch den Abstand zweier Frames. Bei einem Video mit 30 FPS liegt der Fehler bei  $\frac{1}{30} t [s] = 0.03 [s]$ . Die Distanz wird über zwei selektierte Pixelpositionen auf denselben Frames zur Berechnung der Zeit bestimmt. Diese Positionen werden zu Modellkoordinaten  $P_0, P_1$  übersetzt[Luk21c], um die Distanz in der Einheit [mm] anzugeben. Anschliessend kann über die Formel 5.1 die Anfangsgeschwindigkeit berechnet werden, wobei auch die Gleitreibung  $\mu_g$  sowie die Rollreibung  $\mu_r$  bekannt sein müssen<sup>1</sup>.

$$v_0^2 \cdot \frac{2 \cdot (\mu_g - \mu_r)}{49 \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - s = 0 \quad (5.1)$$

Es wird eine quadratische Formel gelöst, welche zwei Resultate  $v_0^1, v_0^2$  liefert. Von diesen wird nur die kleinste positive Lösung  $v_0$  in Betracht gezogen.

Sobald  $v_0$  bestimmt ist, kann der Geschwindigkeitsvektor  $\vec{v}_0$  berechnet werden. Dies geschieht über den Einheitsvektor, welcher über die beiden Punkte zur Bestimmung der Distanz gegeben ist:

$$\vec{d} = P_1 - P_0 \quad (5.2)$$

$$\hat{d} = \frac{\vec{d}}{|\vec{d}|} \quad (5.3)$$

$$\vec{v}_0 = v_0 \cdot \hat{d} \quad (5.4)$$

Im nächsten Schritt kann die Simulation mit der errechneten Startgeschwindigkeit des Spielballs ausgeführt werden. Daraus resultiert eine Abfolge von Ereignissen, die in der Simulation gefunden wurden. Diese werden mit manuell erstellten Ereignissabfolgen abgeglichen, bei welchem alle Positionen der Kugeln und Ereigniszeitpunkte anhand der Videoaufnahmen bestimmt wurden. Die Summe der Differenzen zwischen den erwarteten Zeiten und Positionen bilden den Gesamtfehler.

<sup>1</sup>Die Herleitung erfolgt in Kapitel 8.2.



# 6 Weitere Arbeiten

Auf den in dieser Arbeit erzielten Resultaten sind potentielle weitere Arbeiten für Erweiterungen und Verbesserungen möglich. Nachfolgend werden einige Ideen beschrieben.

Die Suche nach Stößen wird wie in Kapitel 4.6.1 beschrieben ausgehend von den Löchern gestartet, wodurch diese als Rückwärtssuche bezeichnet werden kann. Der Raum der möglichen Stößen wird diskretisiert, weil nicht allen Möglichkeiten untersucht werden. Eine Vorwärtssuche würde viele Simulationen mit unterschiedlichen Anfangsbedingungen (Stossrichtung & -stärke) durchführen und so mögliche Stöße finden. Dabei müsste der unendliche Raum der Anfangsbedingungen sinnvoll diskretisiert werden, um eine solche Suche zu ermöglichen. Wegen der Diskretisierung der Rückwärtssuche kann es in gewissen Situationen Stöße geben, welche nicht gefunden werden. Durch eine Kombination von Rückwärts- und Vorwärtssuche könnten mehr potentielle Stöße untersucht werden.

Die in Kapitel 4.6.4 beschriebene Simulation ist zweidimensional. Zur korrekten Behandlung von aller Arten von Spin, beispielsweise Sidespin, müsste diese auf eine 3D-Simulation erweitert werden. Die Suche könnte weiterhin zweidimensional bleiben, würde dadurch aber auch keine Stöße mit Spin vorschlagen. Dabei könnte evtl. die bereits erwähnte Vereinigung mit einer Vorwärtssuche helfen, welche dem Stoss als Anfangsbedingung zusätzlich auch Spin geben könnte.

Die Genauigkeit der Klassifikation der Kugelfarben könnte weiter optimiert werden, um bessere Resultate zu erzielen. Dies ist bei den über den gesamten Tisch unterschiedlichen Lichtverhältnissen und projizierten Augmentationen herausfordernd.

Des Weiteren könnte es für den Spieler von Nutzen sein, wenn das System die Position und Orientierung des Billardqueue erkennen könnte. Dank dieser Informationen könnte dem Spieler angezeigt werden, wie der Spielball bei einem Stoss in diese Richtung an den Banden reflektieren würde, um so Erfahrungen mit Stößen über die Banden zu sammeln. Es gibt bereits verwandte Arbeiten, wo dies umgesetzt ist, siehe Kapitel 4.10. Außerdem wäre es möglich, verschiedene starke Stöße aufgrund der aktuellen Position und Orientierung des Queue zu simulieren und dem Spieler den besten Stoss animiert anzuzeigen.

Für ein erweitertes Spielerlebnis sind weitere Spielmodi denkbar, beispielsweise die Erkennung und Behandlung von Re gelverstößen, dem Wechsel zwischen zwei Spielern und dem Zählen der Punktzahl jedes Spielers. In einem Trainingsmodus könnte dem Spieler ein Stoss angezeigt, dessen Ausführung beobachtet und dem Spieler Rückmeldung gegeben werden. Über die Darstellung würde es sich anbieten, verschiedene Ereignisse mit speziellen Animationen oder über Ton hervorzuheben, beispielsweise wenn ein Spieler eine Kugel erfolgreich einlocht, oder wenn der Spielball ebenfalls versenkt wird.



## 7 Fazit

TODO: Fazit



# Abbildungsverzeichnis

4.1 Billard-AI-Cycle . . . . .	9
4.2 Risikoanalyse . . . . .	11
4.3 Feedback-Loop in der Detektion . . . . .	11
4.4 Visualisierung des Rauschens in den detektierten Positionen einer roten Kugel über 5 Frames. Die schwarzen Kreise visualisieren die im entsprechenden Frame detektierte Position, die grauen Kreise die zuvor detektierten Positionen. . . . .	12
4.5 Detektierte Positionen derselben Kugel zum Zeitpunkt $t - 1$ und $t$ . Der Durchschnitt dieser beiden Positionen liegt hinter der aktuellen Position. . . . .	13
4.6 Auswahl aus den Trainingsbilder . . . . .	14
4.7 Kandidatensuche 1 . . . . .	16
4.8 Kandidatensuche 2 . . . . .	17
4.9 Kandidatensuche 3 . . . . .	17
4.10 Kugelexpansion . . . . .	18
4.11 Tiefe über Bande erreichen mittels Reflektion . . . . .	18
4.12 Berücksichtigung des Kugelradius bei Bandenreflektion . . . . .	19
4.13 Zweifaches Bandenspiel - A . . . . .	19
4.14 Zweifaches Bandenspiel - B . . . . .	20
4.15 Zweifaches Bandenspiel - C . . . . .	20
4.16 Zweifaches Bandenspiel - D . . . . .	21
4.17 Dreifache Reflektion an Banden . . . . .	21
4.18 Definitionsbereich einer Bandenspiegelung . . . . .	22
4.19 Kugelexpansion - Platz für Queue . . . . .	24
4.20 Bewertung eines Suchbaums . . . . .	25
4.21 Expansionskosten eines Knotens . . . . .	26
4.22 Gewichtung der Winkelkosten . . . . .	26
4.23 Kollisionspunkt zweier Kugeln . . . . .	28
4.24 Typen von Objekten . . . . .	29
4.25 Der Energy-Transfer-Node . . . . .	30
4.26 Beispiel für ein Resultat des Algorithmus 4 . . . . .	31
4.27 Elastischer Stoss zweier Kugeln . . . . .	33
4.28 (a) Eine einkommende Geschwindigkeit $\vec{v}$ wird auf die Komponenten $\vec{z}$ und $\vec{t}$ aufgeteilt. Ohne Berücksichtigung des Topspins wird die Position der weissen Kugel bei $P$ berechnet, die effektive Position ist bei $E$ . (b) Bei grösserer Geschwindigkeit ist die Abweichung zwischen $P$ und $E$ kleiner. . . . .	35
4.29 Vorbedingung einer Prüfung auf Kollision zwischen Kugeln bei statischer Beteiligung . . . . .	37
4.30 Vorbedingung einer Prüfung auf Kollision zwischen dynamischen Kugeln . . . . .	37
4.31 Vorbedingung einer Prüfung auf Kollision zwischen dynamischen parallel velaufenden Kugeln . . . . .	38
4.32 Vorbedingung einer Prüfung auf Kollision zwischen dynamischen hintereinander velaufenden Kugeln . . . . .	38
4.33 Keyframe Animation . . . . .	42
4.34 Simulationsmodell mit Keyframes . . . . .	42
4.35 Simulationsmodell mit paarweisen Keyframefenstern . . . . .	43
4.36 Suchbaum der Tiefensuche . . . . .	44
4.37 Ausgangslage bei Snooker[Unk] . . . . .	45
4.38 Replatzierung einer Kugel nahe am Spot $s$ . . . . .	45
4.39 Berechnungsprozess . . . . .	46
4.40 Berechnungsprozessverteilung . . . . .	47
4.41 Berechnungsprozessynchronisation . . . . .	48
4.42 Anzeige des Tisches mit Banden und Löchern . . . . .	49
4.43 Unity Animationsdarstellung . . . . .	49

5.1	Extremsituationen zum Vergleich der Referenzposition und der wahren Position des Kugelmittelpunktes. Abgebildet ist ein Bildausschnitt mit 9 Pixeln. Die blauen Punkte sind die Pixelzentren, grün ist die wahre Position (subpixelgenau), orange ist die Referenzposition aus den Referenzdaten. Die Pixel rund um die wahre Position sind blau hinterlegt. Oben-Links: Der maximale Fehler ist eine halbe Pixeldiagonale, wenn die wahre Position zwischen vier Pixeln liegt und die Referenzposition einem der Pixel rund um die tatsächliche Position entspricht. Unten-Links: Der minimale Fehler ist 0, wenn die wahre Position genau dem Pixelzentrum entspricht und dieses Pixel in den Referenzdaten ausgewählt wurde. Oben-Rechts: Der maximale Fehler ist 1.5 Pixeldiagonalen, wenn die wahre Position zwischen vier Pixeln liegt und die Referenzposition um ein Pixel daneben ist. Unten-Rechts: Der minimale Fehler ist eine halbe Pixelbreite/-höhe wenn die wahre Position innerhalb des zentralen Pixels liegt und die Referenzposition um ein Pixel falsch daneben ist.	54
5.2	Fälschlicherweise detektierte Kugeln auf der Hand	55
5.3	Box-Plot-Diagramme zu der Bewegung der Kugelpositionen mit und ohne Stabilisierung gemäss Tabelle 5.3. Der Wertebereich der Y-Achse wurde auf 5 beschränkt, damit das rechte Diagramm ersichtlich ist. Die Antenne des linken Diagrammes geht bis zum Maximum von 13.59mm.	56
5.4	Versuchsaufbau zur Bestimmung des Reibungskoeffizienten	57
5.5	Situation 1: Einige verstreute Kugeln	58
5.6	Gefundene Stöße zu Situation 1 nach bewerteter Schwierigkeit aufsteigend	59
8.1	Kollisionspunkt zweier Kugeln	78
8.2	Elastischer Stoß zweier Kugeln	79
8.3	Modell zur Berechnung der Startgeschwindigkeit auf dem Tisch	90
8.4	Modell zur Berechnung des Reibungskoeffizienten	91
8.5	Spiegelung an den verschiedenen Banden	94

# Tabellenverzeichnis

3.1 Ziele . . . . .	6
4.1 Risiken . . . . .	10
5.1 Ergebnisse Klassifikation der Kugelfarben . . . . .	53
5.2 Statistische Zahlen zu den Distanzen zwischen den Kugelpositionen der Referenzdaten und den detektierten Kugelpositionen. . . . .	54
5.3 Statistische Zahlen zu der Verteilung der Veränderung der detektierten Kugelpositionen über eine Dauer von 148 Video-Frames mit und ohne Stabilisierung. Bei einem Messwert handelt es sich um die Distanz zwischen der detektierten Kugelposition in Frame $F_{t-1}$ und Frame $F_t$ für eine Kugel. . . . .	56
5.4 Ergebnisse Distanzmessung einer rollenden Kugel . . . . .	57
5.5 Reibungskoeffizienten über Strecke . . . . .	58



# Literatur

- [And07] Andreas Herschbach. *Entwicklung eines virtuellen Billardspiels*. [Online; accessed 19.11.2021]. 2007. URL: <https://kola.opus.hbz-nrw.de/opus45-kola/frontdoor/deliver/index/docId/128/file/Ausarbeitung.pdf>.
- [AR] Pool Live AR. *Pool Live AR - AR for Billiards or Pool Games*. [Online; accessed 18.11.2021]. Pool Live AR. URL: <https://youtu.be/ozyt6hajcVM>.
- [Bou00] Paul Bourke. *Calculating reflected ray*. [Online; accessed 05.10.2021]. 2000. URL: <http://paulbourke.net/geometry/reflected/>.
- [Dan13] Daniel Arnold. *Billardkugel ohne Schlupf angespielt*. [Online; accessed 19.11.2021], laut Tipler. 2013. URL: <https://de.sci.physik.narkive.com/DLs2FRpR/billardkugel-ohne-schlupf-angespielt>.
- [DJo13] DJohnM. *The puzzle of billiard geometry*. [Online; accessed 24.11.2021]. 2013. URL: <https://math.stackexchange.com/questions/465963/the-puzzle-of-billiard-geometry>.
- [Her] Stuff made Here. *Automatic pool stick vs. strangers*. [Online; accessed 18.11.2021]. Stuff made Here. URL: <https://www.youtube.com/watch?v=vsTTXYxyd0E&list=WL&index=89>.
- [Jac95] Jack Koehler. *The scienece of pocket billards*. [Online; accessed 19.11.2021]. 1995. URL: <https://billard-aktuell.de/forum/viewtopic.php?t=15509>.
- [Luk21a] Luca Ritz Lukas Seglias. "Billiard-AI". In: (2021), S. 5–6.
- [Luk21b] Luca Ritz Lukas Seglias. "Billiard-AI". In: (2021), S. 17–19.
- [Luk21c] Luca Ritz Lukas Seglias. "Billiard-AI". In: (2021), S. 10–16.
- [Luk21d] Luca Ritz Lukas Seglias. "Billiard-AI". In: (2021), S. 21–23.
- [Luk21e] Luca Ritz Lukas Seglias. "Billiard-AI". In: (2021), S. 38.
- [MJP10] Senthon Mathavan, M Jackson und Robert Parkin. "A theoretical analysis of billiard ball dynamics under cushion impacts". In: *Proceedings of The Institution of Mechanical Engineers Part C-journal of Mechanical Engineering Science - PROC INST MECH ENG C-J MECH E 1* (Sep. 2010), S. 1–10. doi: 10.1243/09544062JMES1964.
- [Mül21] Arnd Müller. *Ein Billardroboter - Praktische Realisierung von ausgewählten Konzepten der Robotik*. [Online; accessed 28.10.2021]. 2021. URL: <https://nbn-resolving.org/urn:nbn:de:bsz:15-qucosa2-164739>.
- [Smi06] Michael Smith. "Running the Table: An AI for Computer Billiards." In: Bd. 1. Jan. 2006, S. 996.
- [Unk] Unknown. *Spielregel - Snooker*. [Online; accessed 12.11.2021]. URL: <https://www.stoppball.de/Spielregel-Snooker>.
- [Unk13] Unknown. *Stoß einer Billiardkugel*. [Online; accessed 05.11.2021]. 2013. URL: <https://www.physikerboard.de/topic,35448,-sto%C3%9F-einer-billiardkugel.html>.
- [Unk21a] Unknown. *Arbeit (Physik)*. [Online; accessed 04.10.2021]. 2021. URL: [https://de.wikipedia.org/wiki/Arbeit\\_\(Physik\)](https://de.wikipedia.org/wiki/Arbeit_(Physik)).
- [Unk21b] Unknown. *Bézier curve*. [Online; accessed 28.10.2021]. 2021. URL: [https://en.wikipedia.org/wiki/B%C3%A9zier\\_curve](https://en.wikipedia.org/wiki/B%C3%A9zier_curve).
- [Unk21c] Unknown. *Dijkstra's algorithm*. [Online; accessed 28.10.2021]. 2021. URL: [https://en.wikipedia.org/wiki/Dijkstra%27s\\_algorithm](https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm).
- [Unk21d] Unknown. *Drehmoment*. [Online; accessed 05.11.2021]. 2021. URL: <https://de.wikipedia.org/wiki/Drehmoment>.
- [Unk21e] Unknown. *Effet*. [Online; accessed 05.10.2021]. 2021. URL: <https://de.wikipedia.org/wiki/Effet>.
- [Unk21f] Unknown. *Elastischer Stoß (Physik)*. [Online; accessed 05.10.2021]. 2021. URL: [https://de.wikipedia.org/wiki/Sto%C3%9F\\_\(Physik\)#Elastischer\\_Sto%C3%9F](https://de.wikipedia.org/wiki/Sto%C3%9F_(Physik)#Elastischer_Sto%C3%9F).
- [Unk21g] Unknown. *LearnOpenCV - Module 3.3 Stabilizing Landmark Points in Videos*. [Online; accessed 24.11.2021]. 2021. URL: <https://courses.learnopencv.com/courses/245607/lectures/3819798>.
- [Unk21h] Unknown. *Polynom vierten Grades*. [Online; accessed 04.10.2021]. 2021. URL: [https://de.wikipedia.org/wiki/Polynom\\_vierten\\_Grades](https://de.wikipedia.org/wiki/Polynom_vierten_Grades).

- [Unk21i] Unknown. *Quadratische Gleichung - Lösungsformel*. [Online; accessed 04.10.2021]. 2021. URL: [https://de.wikipedia.org/wiki/Quadratische\\_Gleichung#Allgemeine\\_L%C3%B6sungsformeln](https://de.wikipedia.org/wiki/Quadratische_Gleichung#Allgemeine_L%C3%B6sungsformeln).
- [Unk21j] Unknown. *Rollen*. [Online; accessed 05.11.2021]. 2021. URL: <https://de.wikipedia.org/wiki/Rollen>.
- [Unk21k] Unknown. *Rollwiderstand*. [Online; accessed 04.10.2021]. 2021. URL: <https://de.wikipedia.org/wiki/Rollwiderstand>.
- [Unk21l] Unknown. *Winkelbeschleunigung*. [Online; accessed 05.11.2021]. 2021. URL: <https://de.wikipedia.org/wiki/Winkelbeschleunigung>.
- [Wik21a] Wikipedia. *Wikipedia - Circle Hough Transform*. [Online; accessed 18.11.2021]. 2021. URL: [https://en.wikipedia.org/wiki/Circle\\_Hough\\_Transform](https://en.wikipedia.org/wiki/Circle_Hough_Transform).
- [Wik21b] Wikipedia. *Wikipedia - Confusion Matrix*. [Online; accessed 01.12.2021]. 2021. URL: [https://en.wikipedia.org/wiki/Confusion\\_matrix](https://en.wikipedia.org/wiki/Confusion_matrix).
- [Wik21c] Wikipedia. *Wikipedia - HSV-Farbraum*. [Online; accessed 18.11.2021]. 2021. URL: <https://de.wikipedia.org/wiki/HSV-Farbraum>.
- [Wik21d] Wikipedia. *Wikipedia - k-nearest neighbors algorithm*. [Online; accessed 18.11.2021]. 2021. URL: [https://en.wikipedia.org/wiki/K-nearest\\_neighbors\\_algorithm](https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm).
- [Wik21e] Wikipedia. *Wikipedia - Moving average*. [Online; accessed 24.11.2021]. 2021. URL: [https://en.wikipedia.org/wiki/Moving\\_average](https://en.wikipedia.org/wiki/Moving_average).



## Erklärung der Diplandinnen und Diplomanden

### *Déclaration des diplômant-e-s*

#### **Selbständige Arbeit / Travail autonome**

Ich bestätige mit meiner Unterschrift, dass ich meine vorliegende Bachelor-Thesis selbstständig durchgeführt habe. Alle Informationsquellen (Fachliteratur, Besprechungen mit Fachleuten, usw.) und anderen Hilfsmittel, die wesentlich zu meiner Arbeit beigetragen haben, sind in meinem Arbeitsbericht im Anhang vollständig aufgeführt. Sämtliche Inhalte, die nicht von mir stammen, sind mit dem genauen Hinweis auf ihre Quelle gekennzeichnet.

*Par ma signature, je confirme avoir effectué ma présente thèse de bachelor de manière autonome. Toutes les sources d'information (littérature spécialisée, discussions avec spécialistes etc.) et autres ressources qui m'ont fortement aidé-e dans mon travail sont intégralement mentionnées dans l'annexe de ma thèse. Tous les contenus non rédigés par mes soins sont dûment référencés avec indication précise de leur provenance.*

Name/Nom, Vorname/Prénom .....  
.....

Datum/Date .....  
.....

Unterschrift/Signature .....  
.....

Dieses Formular ist dem Bericht zur Bachelor-Thesis beizulegen.  
*Ce formulaire doit être joint au rapport de la thèse de bachelor.*



## Erklärung der Diplandinnen und Diplomanden

### *Déclaration des diplômant-e-s*

#### **Selbständige Arbeit / Travail autonome**

Ich bestätige mit meiner Unterschrift, dass ich meine vorliegende Bachelor-Thesis selbstständig durchgeführt habe. Alle Informationsquellen (Fachliteratur, Besprechungen mit Fachleuten, usw.) und anderen Hilfsmittel, die wesentlich zu meiner Arbeit beigetragen haben, sind in meinem Arbeitsbericht im Anhang vollständig aufgeführt. Sämtliche Inhalte, die nicht von mir stammen, sind mit dem genauen Hinweis auf ihre Quelle gekennzeichnet.

*Par ma signature, je confirme avoir effectué ma présente thèse de bachelor de manière autonome. Toutes les sources d'information (littérature spécialisée, discussions avec spécialistes etc.) et autres ressources qui m'ont fortement aidé-e dans mon travail sont intégralement mentionnées dans l'annexe de ma thèse. Tous les contenus non rédigés par mes soins sont dûment référencés avec indication précise de leur provenance.*

Name/Nom, Vorname/Prénom .....  
.....

Datum/Date .....  
.....

Unterschrift/Signature .....  
.....

Dieses Formular ist dem Bericht zur Bachelor-Thesis beizulegen.  
*Ce formulaire doit être joint au rapport de la thèse de bachelor.*

# 8 Anhang

## 8.1 Herleitung Startgeschwindigkeit auf Basis bekannter Endgeschwindigkeit unter Einbezug von Reibung

Dies erfordert den Energieerhaltungssatz. Dazu wird die kinetische Energie vor sowie nachher betrachtet.

$$E_{kin} = \frac{1}{2} \cdot m \cdot |\vec{v}|^2 \quad (8.1)$$

$$E_{vorher} = \frac{1}{2} \cdot m \cdot |\vec{v}_1|^2 \quad (8.2)$$

$$E_{nachher} = \frac{1}{2} \cdot m \cdot |\vec{v}_2|^2 \quad (8.3)$$

Die Reibung  $F_R$  wird mithilfe der Normalkraft  $F_N$  und dem Widerstandskoeffizienten  $\mu$  definiert[Unk21k]. Mithilfe der Masse  $m$  der Kugel und der Schwerkraft  $g$  kann die Normalkraft  $F_N$  berechnet werden.

$$F_R = \mu \cdot F_N \quad (8.4)$$

$$F_N = m \cdot g \quad (8.5)$$

Da die Kugel nicht von Beginn an rollt, sondern zuerst gleitet, besteht die Reibung nicht nur aus Roll-  $F_R^r$  sondern auch aus Gleitreibung  $F_R^g$ .

$$F_R = F_R^r + F_R^g \quad (8.6)$$

Es gibt daher auch zwei Widerstandskoeffizienten für die jeweiligen Reibungen. Für die Rollreibung ist dies  $\mu_r$  und für die Gleitreibung  $\mu_g$ . Die Reibung  $F_R$  wird über eine bestimmte Strecke  $\Delta s$ , welche die Kugel zurücklegt, angewendet. Dadurch entsteht eine Arbeit  $E_{Reibung}$ [Unk21a]:

$$E_{Reibung} = F_R \cdot \Delta s \quad (8.7)$$

Da die beiden Reibungen nur auf bestimmten Teilstrecken relevant sind, werden diese entsprechend aufgeteilt.

$$\Delta s = \Delta s_r + \Delta s_g \quad (8.8)$$

$$E_{Reibung}^r = F_R^r \cdot \Delta s_r \quad (8.9)$$

$$E_{Reibung}^g = F_R^g \cdot \Delta s_g \quad (8.10)$$

$$(8.11)$$

Ein relevanter Zusatz vor der Herleitung ist die Tatsache, dass die Strecke  $\Delta s_g$  des Gleitens von der initialen Startgeschwindigkeit abhängig ist, welche in diesem Schritt berechnet werden soll. Diese Abhängigkeit wird in Kapitel 8.9 beschrieben und hergeleitet. Die wichtigsten Erkenntnisse lauten:

$$\Delta s_g = \frac{12}{49} \cdot \frac{v_0^2}{g \cdot \mu_g} \quad (8.12)$$

$$\Delta s_r = \Delta s - \Delta s_g \quad (8.13)$$

Nun kann der Energieerhaltungssatz angewendet werden.

$$E_{vorher} = E_{nachher} + E_{Reibung} \quad (8.14)$$

$$E_{vorher} = E_{nachher} + E_{Reibung}^r + E_{Reibung}^g \quad (8.15)$$

$$\frac{1}{2} \cdot m \cdot |\vec{v}_1|^2 = \frac{1}{2} \cdot m \cdot |\vec{v}_2|^2 + F_R^r \cdot \Delta s_r + F_R^g \cdot \Delta s_g \quad (8.16)$$

$$m \cdot |\vec{v}_1|^2 = m \cdot |\vec{v}_2|^2 + 2 \cdot F_R^r \cdot \Delta s_r + 2 \cdot F_R^g \cdot \Delta s_g \quad (8.17)$$

$$|\vec{v}_1|^2 = \frac{m \cdot |\vec{v}_2|^2 + 2 \cdot F_R^r \cdot \Delta s_r + 2 \cdot F_R^g \cdot \Delta s_g}{m} \quad (8.18)$$

$$|\vec{v}_1|^2 = \frac{m \cdot |\vec{v}_2|^2 + 2 \cdot m \cdot g \cdot \mu_r \cdot \Delta s_r + 2 \cdot m \cdot g \cdot \mu_g \cdot \Delta s_g}{m} \quad (8.19)$$

$$|\vec{v}_1|^2 = |\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot \Delta s_r + 2 \cdot g \cdot \mu_g \cdot \Delta s_g \quad (8.20)$$

$$|\vec{v}_1|^2 = |\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot (\Delta s - \Delta s_g) + 2 \cdot g \cdot \mu_g \cdot \Delta s_g \quad (8.21)$$

$$|\vec{v}_1|^2 = |\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot (\Delta s - \frac{12}{49} \cdot \frac{|\vec{v}_1|^2}{g \cdot \mu_g}) + 2 \cdot g \cdot \mu_g \cdot (\frac{12}{49} \cdot \frac{|\vec{v}_1|^2}{g \cdot \mu_g}) \quad (8.22)$$

$$|\vec{v}_1|^2 = |\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot \Delta s - \frac{24 \cdot g \cdot \mu_r \cdot |\vec{v}_1|^2}{49 \cdot g \cdot \mu_g} + \frac{24 \cdot g \cdot \mu_g \cdot |\vec{v}_1|^2}{49 \cdot g \cdot \mu_g} \quad (8.23)$$

$$|\vec{v}_1|^2 = |\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot \Delta s - \frac{24 \cdot \mu_r \cdot |\vec{v}_1|^2}{49 \cdot \mu_g} + \frac{24 \cdot |\vec{v}_1|^2}{49} \quad (8.24)$$

$$|\vec{v}_1|^2 + \frac{24 \cdot \mu_r \cdot |\vec{v}_1|^2}{49 \cdot \mu_g} - \frac{24 \cdot |\vec{v}_1|^2}{49} = |\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot \Delta s \quad (8.25)$$

$$|\vec{v}_1|^2 \cdot (1 + \frac{24 \cdot \mu_r}{49 \cdot \mu_g} - \frac{24}{49}) = |\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot \Delta s \quad (8.26)$$

$$|\vec{v}_1|^2 = \frac{|\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot \Delta s}{1 + \frac{24 \cdot \mu_r}{49 \cdot \mu_g} - \frac{24}{49}} \quad (8.27)$$

$$|\vec{v}_1|^2 = \frac{|\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot \Delta s}{\frac{49 \cdot \mu_g}{49 \cdot \mu_g} + \frac{24 \cdot \mu_r}{49 \cdot \mu_g} - \frac{24 \cdot \mu_g}{49 \cdot \mu_g}} \quad (8.28)$$

$$|\vec{v}_1|^2 = \frac{|\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot \Delta s}{\frac{49 \cdot \mu_g + 24 \cdot \mu_r - 24 \cdot \mu_g}{49 \cdot \mu_g}} \quad (8.29)$$

$$|\vec{v}_1|^2 = \frac{|\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot \Delta s}{\frac{49 \cdot \mu_g + 24 \cdot (\mu_r - \mu_g)}{49 \cdot \mu_g}} \quad (8.30)$$

$$|\vec{v}_1|^2 = \frac{49 \cdot \mu_g \cdot (|\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot \Delta s)}{49 \cdot \mu_g + 24 \cdot (\mu_r - \mu_g)} \quad (8.31)$$

$$|\vec{v}_1| = \sqrt{\frac{49 \cdot \mu_g \cdot (|\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot \Delta s)}{49 \cdot \mu_g + 24 \cdot (\mu_r - \mu_g)}} \quad (8.32)$$

Damit ist die Startgeschwindigkeit bestimmt. Nun stellt sich noch die Frage nach der Richtung. Diese zeigt in dieselbe, wie die Endgeschwindigkeit. Daher kann nun die bekannte Länge mit dem normalisierten Vektor  $\vec{v}_2$  multipliziert werden:

$$\vec{v}_1 = |\vec{v}_1| \cdot \frac{\vec{v}_2}{|\vec{v}_2|} \quad (8.33)$$

Daraus folgt die Formel:

$$\vec{v}_1 = \sqrt{\frac{49 \cdot \mu_g \cdot (|\vec{v}_2|^2 + 2 \cdot g \cdot \mu_r \cdot \Delta s)}{49 \cdot \mu_g + 24 \cdot (\mu_r - \mu_g)}} \cdot \frac{\vec{v}_2}{|\vec{v}_2|} \quad (8.34)$$

## 8.2 Herleitung Startgeschwindigkeit auf Basis der Zeit und Distanz unter Einbezug von Reibung

Über eine Strecke  $s$  wirkt eine Gleit-  $F_R^g$  sowie eine Rollreibungskraft  $F_R^r$  auf die Kugel. Die verschiedenen Reibungskräfte wirken nur auf sukzessiven Teilstrecken. So wirkt die Gleitreibungskraft nur auf der Teilstrecke  $s_g$  und die Rollreibungskraft auf der Teilstrecke  $s_r$ <sup>1</sup>.

<sup>1</sup>Eine Herleitung der Initialgeschwindigkeit auf Basis der bekannten Endgeschwindigkeit wie auch der Strecke wird im Kapitel 8.1 vorgestellt.

Die nachfolgenden Gleichungen sind demnach gegeben:

$$s = \frac{1}{2} \cdot a \cdot t^2 + v_0 \cdot t \quad (8.35)$$

$$s = s_g + s_r \quad (8.36)$$

Aus Kapitel 8.9 ist die Teilstrecke, auf welcher die Kugel nur gleitet, zu entnehmen. Hierbei steht  $\mu_g$  für den Gleitreibungskoeffizienten:

$$s_g = \frac{12}{49} \cdot \frac{v_0^2}{g \cdot \mu_g} \quad (8.37)$$

Die Zeit  $t$  bezieht sich auf die Gesamtzeit, welche eine Kugel braucht, um die Strecke  $s$  zu überqueren. Diese setzt sich wiederum aus der Zeit  $t_g$  sowie  $t_r$  zusammen, die die Kugel für das Zurücklegen der einzelnen Teilstrecken benötigt. Es wird bereits eine Umformung vollzogen, da diese in der nachfolgenden Herleitung erforderlich ist.

$$t = t_g + t_r \quad (8.38)$$

$$t_r = t - t_g \quad (8.39)$$

Die Zeit  $t_g$  ist aus Kapitel 8.8 bekannt. In der Herleitung wird auch die quadrierte Form verwendet, weswegen diese ebenso angegeben wird.

$$t_g = \frac{v_0}{\frac{7}{2} \cdot g \cdot \mu_g} \quad (8.40)$$

$$t_g^2 = \frac{v_0^2}{\frac{49}{4} \cdot g^2 \cdot \mu_g^2} \quad (8.41)$$

Für die abschliessende Herleitung fehlt nur noch die Teilstrecke  $s_r$ . Da die Beschleunigung  $a$  mit  $g$  definiert ist, und  $g$  als positive Zahl angenommen wird, kommt das negative Vorzeichen dazu, damit  $a$  negativ wird.

$$s_r = \frac{1}{2} \cdot a_r \cdot t_r^2 + v_0^r \cdot t_r \quad (8.42)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t_r^2 + v_0^r \cdot t_r \quad (8.43)$$

Die Anfangsgeschwindigkeit  $v_0^r$  entspricht der Endgeschwindigkeit der Kugel nach dem Gleiten:

$$v_0^r = a_g \cdot t_g + v_0 \quad (8.44)$$

$$v_0^r = -g \cdot \mu_g \cdot t_g + v_0 \quad (8.45)$$

Die Rollzeit der Gleitung der Teilstrecke  $s_r$  kann über die Beziehung zur Gesamtzeit und der Gleitzeit ausgedrückt werden.

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot (t - t_g)^2 + v_0^r \cdot (t - t_g) \quad (8.46)$$

Die Anfangsgeschwindigkeit  $v_0^r$  beim Rollen kann über die Endgeschwindigkeit nach dem Gleiten ausgedrückt werden.

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot (t - t_g)^2 + (-g \cdot \mu_g \cdot t_g + v_0) \cdot (t - t_g) \quad (8.47)$$

Es werden die Variablen  $t_g$  nacheinander ersetzt und umgeformt.

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot (t^2 - 2 \cdot t \cdot t_g + t_g^2) + (-g \cdot \mu_g \cdot t_g + v_0) \cdot (t - t_g) \quad (8.48)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot (t^2 + t_g^2) + \frac{1}{2} \cdot g \cdot \mu_r \cdot 2 \cdot t \cdot t_g + (-g \cdot \mu_g \cdot t_g + v_0) \cdot (t - t_g) \quad (8.49)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot (t^2 + t_g^2) + g \cdot \mu_r \cdot t \cdot t_g + (-g \cdot \mu_g \cdot t_g + v_0) \cdot (t - t_g) \quad (8.50)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot (t^2 + t_g^2) + g \cdot \mu_r \cdot t \cdot \frac{v_0}{\frac{7}{2} \cdot g \cdot \mu_g} + (-g \cdot \mu_g \cdot t_g + v_0) \cdot (t - t_g) \quad (8.51)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot (t^2 + t_g^2) + v_0 \cdot \frac{\mu_r \cdot t}{\frac{7}{2} \cdot \mu_g} + (-g \cdot \mu_g \cdot t_g + v_0) \cdot (t - t_g) \quad (8.52)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - \frac{1}{2} \cdot g \cdot \mu_r \cdot t_g^2 + v_0 \cdot \frac{\mu_r \cdot t}{\frac{7}{2} \cdot \mu_g} + (-g \cdot \mu_g \cdot t_g + v_0) \cdot (t - t_g) \quad (8.53)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - \frac{1}{2} \cdot g \cdot \mu_r \cdot \frac{v_0^2}{\frac{49}{4} \cdot g^2 \cdot \mu_g^2} + v_0 \cdot \frac{\mu_r \cdot t}{\frac{7}{2} \cdot \mu_g} + (-g \cdot \mu_g \cdot t_g + v_0) \cdot (t - t_g) \quad (8.54)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - v_0^2 \cdot \frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{\mu_r \cdot t}{\frac{7}{2} \cdot \mu_g} + (-g \cdot \mu_g \cdot t_g + v_0) \cdot (t - t_g) \quad (8.55)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - v_0^2 \cdot \frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{\mu_r \cdot t}{\frac{7}{2} \cdot \mu_g} + (-g \cdot \mu_g \cdot \frac{v_0}{\frac{7}{2} \cdot g \cdot \mu_g} + v_0) \cdot (t - t_g) \quad (8.56)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - v_0^2 \cdot \frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{\mu_r \cdot t}{\frac{7}{2} \cdot \mu_g} + (-v_0 \cdot \frac{1}{\frac{7}{2}} + v_0) \cdot (t - t_g) \quad (8.57)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - v_0^2 \cdot \frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{\mu_r \cdot t}{\frac{7}{2} \cdot \mu_g} + v_0 \cdot (-\frac{1}{\frac{7}{2}} + 1) \cdot (t - t_g) \quad (8.58)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - v_0^2 \cdot \frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{\mu_r \cdot t}{\frac{7}{2} \cdot \mu_g} + v_0 \cdot (-\frac{2}{7} + \frac{7}{7}) \cdot (t - t_g) \quad (8.59)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - v_0^2 \cdot \frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{\mu_r \cdot t}{\frac{7}{2} \cdot \mu_g} + v_0 \cdot \frac{5}{7} \cdot (t - t_g) \quad (8.60)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - v_0^2 \cdot \frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{\mu_r \cdot t}{\frac{7}{2} \cdot \mu_g} + v_0 \cdot \frac{5 \cdot t}{7} - v_0 \cdot \frac{5}{7} \cdot t_g \quad (8.61)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - v_0^2 \cdot \frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} + v_0 \cdot (\frac{\mu_r \cdot t}{\frac{7}{2} \cdot \mu_g} + \frac{5 \cdot t}{7}) - v_0 \cdot \frac{5}{7} \cdot t_g \quad (8.62)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - v_0^2 \cdot \frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} + v_0 \cdot (\frac{2 \cdot \mu_r \cdot t}{7 \cdot \mu_g} + \frac{5 \cdot t \cdot \mu_g}{7 \cdot \mu_g}) - v_0 \cdot \frac{5}{7} \cdot t_g \quad (8.63)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - v_0^2 \cdot \frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{2 \cdot \mu_r \cdot t + 5 \cdot t \cdot \mu_g}{7 \cdot \mu_g} - v_0 \cdot \frac{5}{7} \cdot t_g \quad (8.64)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - v_0^2 \cdot \frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - v_0 \cdot \frac{5}{7} \cdot t_g \quad (8.65)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - v_0^2 \cdot \frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - v_0 \cdot \frac{v_0}{\frac{7}{2} \cdot g \cdot \mu_g} \quad (8.66)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - v_0^2 \cdot \frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - v_0^2 \cdot \frac{5}{\frac{49}{2} \cdot g \cdot \mu_g} \quad (8.67)$$

$$s_r = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - v_0^2 \cdot (\frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} - \frac{5}{\frac{49}{2} \cdot g \cdot \mu_g}) + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} \quad (8.68)$$

$$s_r = -v_0^2 \cdot (\frac{\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} + \frac{5}{\frac{49}{2} \cdot g \cdot \mu_g}) + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 \quad (8.69)$$

$$s_r = v_0^2 \cdot (\frac{-\mu_r}{\frac{98}{4} \cdot g \cdot \mu_g^2} - \frac{5}{\frac{49}{2} \cdot g \cdot \mu_g}) + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 \quad (8.70)$$

$$s_r = v_0^2 \cdot (\frac{-\mu_r}{\frac{49}{2} \cdot g \cdot \mu_g^2} - \frac{5 \cdot \mu_g}{\frac{49}{2} \cdot g \cdot \mu_g^2}) + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 \quad (8.71)$$

$$s_r = v_0^2 \cdot \frac{-\mu_r - 5 \cdot \mu_g}{\frac{49}{2} \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 \quad (8.72)$$

Es wird nun die Anfangsgeschwindigkeit  $v_0$  berechnet, welche die Kugel braucht, um die Strecke  $s$  in der Zeit  $t$  zurückzulegen.

$$s = s_g + s_r \quad (8.73)$$

$$s = \frac{12}{49} \cdot \frac{v_0^2}{g \cdot \mu_g} + v_0^2 \cdot \frac{-\mu_r - 5 \cdot \mu_g}{\frac{49}{2} \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 \quad (8.74)$$

$$s = v_0^2 \cdot \frac{12}{49 \cdot g \cdot \mu_g} + v_0^2 \cdot \frac{-\mu_r - 5 \cdot \mu_g}{\frac{49}{2} \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 \quad (8.75)$$

$$s = v_0^2 \cdot \left( \frac{-\mu_r - 5 \cdot \mu_g}{\frac{49}{2} \cdot g \cdot \mu_g^2} + \frac{12}{49 \cdot g \cdot \mu_g} \right) + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 \quad (8.76)$$

$$s = v_0^2 \cdot \left( \frac{-2 \cdot \mu_r - 10 \cdot \mu_g}{49 \cdot g \cdot \mu_g^2} + \frac{12 \cdot \mu_g}{49 \cdot g \cdot \mu_g^2} \right) + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 \quad (8.77)$$

$$s = v_0^2 \cdot \frac{-2 \cdot \mu_r - 10 \cdot \mu_g + 12 \cdot \mu_g}{49 \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 \quad (8.78)$$

$$s = v_0^2 \cdot \frac{-2 \cdot \mu_r + 2 \cdot \mu_g}{49 \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 \quad (8.79)$$

$$s = v_0^2 \cdot \frac{2 \cdot (\mu_g - \mu_r)}{49 \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 \quad (8.80)$$

$$v_0^2 \cdot \frac{2 \cdot (\mu_g - \mu_r)}{49 \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - s = 0 \quad (8.81)$$

Die Formel zur Berechnung der Initialgeschwindigkeit auf Basis der bekannten Strecke  $s$ , Zeit  $t$ , Gleitreibung  $\mu_g$  und Rollreibung  $\mu_r$  lautet:

$$v_0^2 \cdot \frac{2 \cdot (\mu_g - \mu_r)}{49 \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - s = 0 \quad (8.82)$$

Die Koeffizienten zur Lösung der quadratischen Gleichung sind:

$$a = \frac{2 \cdot (\mu_g - \mu_r)}{49 \cdot g \cdot \mu_g^2} \quad (8.83)$$

$$b = \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} \quad (8.84)$$

$$c = -\frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - s \quad (8.85)$$

Wenn die Reibungskoeffizienten  $\mu_g$  und  $\mu_r$  äquivalent sind, reduziert sich der Ausdruck auf die gleichmässig beschleunigte Geschwindigkeit. Es werden  $\mu_g$  und  $\mu_r$  durch  $\mu$  ersetzt.

$$v_0^2 \cdot \frac{2 \cdot (\mu_g - \mu_r)}{49 \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 - s = 0 \quad (8.86)$$

$$s = v_0^2 \cdot \frac{2 \cdot (\mu_g - \mu_r)}{49 \cdot g \cdot \mu_g^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu_r + 5 \cdot \mu_g)}{7 \cdot \mu_g} - \frac{1}{2} \cdot g \cdot \mu_r \cdot t^2 \quad (8.87)$$

$$s = v_0^2 \cdot \frac{2 \cdot (\mu - \mu)}{49 \cdot g \cdot \mu^2} + v_0 \cdot \frac{t \cdot (2 \cdot \mu + 5 \cdot \mu)}{7 \cdot \mu} - \frac{1}{2} \cdot g \cdot \mu \cdot t^2 \quad (8.88)$$

$$s = v_0 \cdot \frac{t \cdot 7 \cdot \mu}{7 \cdot \mu} - \frac{1}{2} \cdot g \cdot \mu \cdot t^2 \quad (8.89)$$

$$s = v_0 \cdot t - \frac{1}{2} \cdot g \cdot \mu \cdot t^2 \quad (8.90)$$

$$s = -\frac{1}{2} \cdot g \cdot \mu \cdot t^2 + v_0 \cdot t \quad (8.92)$$

$$s = \frac{1}{2} \cdot a \cdot t^2 + v_0 \cdot t \quad (8.93)$$

### 8.3 Herleitung Initialgeschwindigkeit bei Kugel-Kollision

Eine Kugel T soll in eine gewünschte Richtung rollen. Dazu soll sie von einer anderen Kugel A angestossen werden. Gesucht ist der Punkt, wohin Kugel A rollen muss, um mit Kugel T zusammenzustossen. Die Kugel T soll mit einer bestimmten Geschwindigkeit von der Kollision ausgehen. Dazu ist die Geschwindigkeit zu bestimmen, die Kugel A zum Zeitpunkt der Kollision haben muss.

Die Situation mit der Kugel T, an der Position C und der Kugel A, an der Position A ist in Abbildung 8.1 dargestellt. Die Kugel A muss zum Punkt B rollen, wo sie auf Kugel T prallt und ein elastischer Stoß[Unk21f] stattfindet. Während die Kugel die Distanz  $|\vec{d}|$  zurücklegt, verliert sie an Geschwindigkeit aufgrund von Reibung, diese wird in Abschnitt 4.6.2.1 behandelt. Hier ist lediglich relevant, welche Geschwindigkeit die Kugel A am Punkt B haben muss.

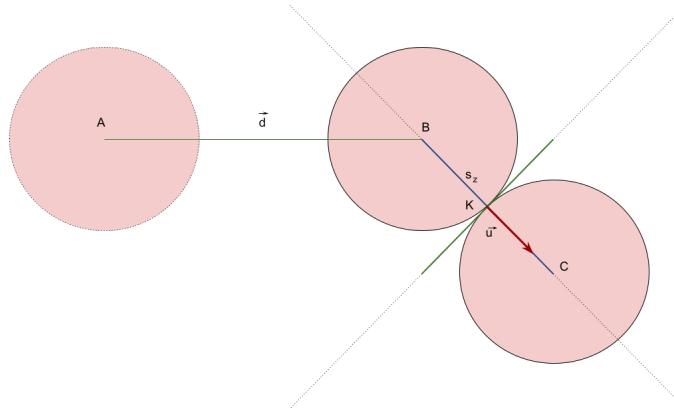


Abbildung 8.1: Kollisionspunkt zweier Kugeln

Sei die gewünschte Richtung, in die Kugel T nach der Kollision rollen soll,  $\hat{u}$  und der Kugelradius  $r$ , dann kann die Position B bestimmt werden:

$$s_z = 2 \cdot r \quad (8.94)$$

$$B = C - s_z \cdot \hat{u} \quad (8.95)$$

$$B = C - 2 \cdot r \cdot \hat{u} \quad (8.96)$$

Nun gilt es die Geschwindigkeit  $v_{1,z}$  der Kugel A zum Zeitpunkt der Kollision zu bestimmen. Nachfolgend wird angenommen, dass die zu treffende Kugel T stillsteht und somit die Geschwindigkeit  $\vec{v}_2 = \vec{0}$  hat. Der Geschwindigkeitsvektor  $\vec{u}$  ist durch die gewünschte Richtung und Geschwindigkeit, welche die Kugel T nach dem Zusammenstoß haben soll, gegeben.

$$\vec{u} = \vec{v}_{1,z} + \vec{v}_{2,t} \quad (8.97)$$

$$\vec{u} = \vec{v}_{1,z} + \vec{0} \quad (8.98)$$

$$\vec{u} = \vec{v}_{1,z} \quad (8.99)$$

Die Situation ist in Abbildung 8.2 dargestellt.

Gegeben sind die nachfolgenden Informationen:

- A: Startposition der Kugel A.
- C: Position der Kugel T.
- $\vec{u}$ : Gewünschte Geschwindigkeit und Richtung der Kugel T nach dem Zusammenstoß.

Weiterhin ist die Richtung von  $\vec{v}_1$  gegeben, da dieser parallel zum Vektor  $\vec{d}$  ist. Allerdings muss die Länge des Vektors noch bestimmt werden:

$$\vec{d} = B - A \quad (8.100)$$

$$\hat{d} = \frac{\vec{d}}{|\vec{d}|} \quad (8.101)$$

$$\vec{v}_1 = \alpha \cdot \hat{d} \quad (8.102)$$

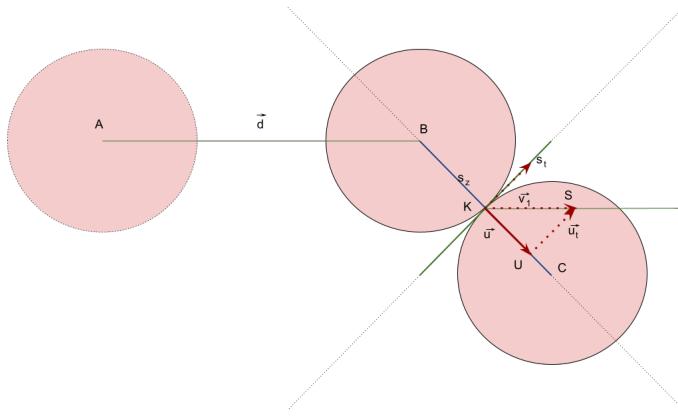


Abbildung 8.2: Elastischer Stoss zweier Kugeln

Die Länge des Vektors  $\vec{v}_1$  muss nach Projektion auf den Einheitsvektor  $\hat{u}$  in Richtung von  $\vec{u}$  der Länge des Vektors  $\vec{u}$  entsprechen. Dazu muss ein Skalierungsfaktor  $\alpha$  bestimmt werden, mit dem der Einheitsvektor  $\hat{d}$  skaliert werden kann:

$$\hat{u} = \frac{\vec{u}}{|\vec{u}|} \quad (8.103)$$

$$\vec{v}_1 \cdot \hat{u} = |\vec{u}| \quad (8.104)$$

$$(\alpha \cdot \hat{d}) \cdot \hat{u} = |\vec{u}| \quad (8.105)$$

$$\alpha \cdot (\hat{d} \cdot \hat{u}) = |\vec{u}| \quad (8.106)$$

$$\alpha = \frac{|\vec{u}|}{\hat{d} \cdot \hat{u}} \quad (8.107)$$

$$\alpha = \frac{|\vec{u}|}{\hat{d} \cdot \frac{\vec{u}}{|\vec{u}|}} \quad (8.108)$$

$$\alpha = \frac{|\vec{u}|^2}{\hat{d} \cdot \vec{u}} \quad (8.109)$$

$$\alpha = \frac{\vec{u} \cdot \vec{u}}{\hat{d} \cdot \vec{u}} \quad (8.110)$$

$$\vec{v}_1 = \alpha \cdot \hat{d} \quad (8.111)$$

$$\vec{v}_1 = \frac{\vec{u} \cdot \vec{u}}{\hat{d} \cdot \vec{u}} \cdot \hat{d} \quad (8.112)$$

Somit ist die Geschwindigkeit, welche die Kugel A an Position B haben muss bestimmt. Der Energieverlust wird über eine Konstante  $E_v$  definiert, welche in Prozent angegeben wird. Es resultiert eine Verlängerung des berechneten Vektors.

$$\vec{v}_1 = \frac{\vec{u} \cdot \vec{u}}{\hat{d} \cdot \vec{u}} \cdot \frac{1}{1 - E_v} \cdot \hat{d} \quad (8.113)$$

## 8.4 Herleitung Ereignis - Totaler Energieverlust

Dieses Ereignis tritt ein, sobald eine Kugel ausrollt. Der Zeitpunkt wird über die Geschwindigkeitsgleichung der gleichmässig beschleunigten Bewegung bestimmt.

$$\vec{v}(t) = \vec{a} \cdot t + \vec{v}_0 \quad (8.114)$$

Bekannt ist, dass der Betrag des Geschwindigkeitsvektors 0 sein muss.

$$|\vec{v}(t)| = 0 \quad (8.115)$$

$$\vec{0} = \vec{a} \cdot t + \vec{v}_0 \quad (8.116)$$

Um  $t$  anhand von bekannten Werten  $\vec{a}$  und  $\vec{v}_0$  zu ermitteln, gibt es zwei Möglichkeiten. Bei der Ersten kann die Gleichung 8.116 Komponentenweise aufgestellt werden, wobei auch  $t$  in zwei verschiedene Richtungen betrachtet werden muss.

$$0 = a_x \cdot t_x + v_{0,x} \quad (8.117)$$

$$0 = a_y \cdot t_y + v_{0,y} \quad (8.118)$$

Damit ergeben sich zwei unabhängige Gleichungen mit zwei unbekannten  $t_x$  und  $t_y$ , die Zeitpunkte, an denen die Kugel in der entsprechenden Richtung stillsteht. Angenommen, eine Kugel rollt nur in X-Richtung, dann ist der Zeitpunkt  $t_x$ , zu dem die Kugel in X-Richtung stillsteht und der Zeitpunkt  $t_y$ , zu dem die Kugel in Y-Richtung stillsteht, unterschiedlich. Angenommen, eine Kugel rollt mit einer Geschwindigkeit in X- und Y-Richtung. Dann entspricht die Beschleunigung  $\vec{a}$  einem konstanten Faktor angewendet auf den Geschwindigkeitsvektor.

$$\vec{a} = -\mu \cdot \vec{v}_0 \quad (8.119)$$

Die beiden Variablen  $t_x$  und  $t_y$  sind in diesem Fall äquivalent. Eine Unterscheidung auf die beiden Komponenten wird also nur getroffen, da sich das Problem auf nur eine Dimension beschränken kann. Es ergeben sich die beiden nachfolgenden Formeln.

$$t_x = \frac{-v_{0,x}}{a_x} \quad (8.120)$$

$$t_y = \frac{-v_{0,y}}{a_y} \quad (8.121)$$

Der Zeitpunkt  $t$  zu dem die Kugel vollständig stillsteht, ist so wie folgt gegeben:

$$t = \max(t_x, t_y) \quad (8.122)$$

$$t = \max\left(\frac{-v_{0,x}}{a_x}, \frac{-v_{0,y}}{a_y}\right) \quad (8.123)$$

Die andere Möglichkeit ist die Betrachtung der Gleichung 8.116 über die Beträge der Vektoren  $\vec{a}$  sowie  $\vec{v}_0$ .

Werden die Beträge genommen, dann kann die Gleichung 8.116 als eindimensionales Problem aufgefasst und dementsprechend gelöst werden. Auch hier ist der Vektor  $\vec{a}$  abhängig von der Geschwindigkeit  $\vec{v}_0$  über einen konstanten Faktor.

$$v(t) = |\vec{a}| \cdot t + |\vec{v}_0| \quad (8.124)$$

Es folgen einige Umformungen, um die Variable  $t$  auf eine Seite der Gleichung zu bringen.

$$0 = |\vec{a}| \cdot t + |\vec{v}_0| \quad (8.125)$$

$$-|\vec{v}_0| = |\vec{a}| \cdot t \quad (8.126)$$

$$-\sqrt{v_{0,x}^2 + v_{0,y}^2} = \sqrt{a_x^2 + a_y^2} \cdot t \quad (8.127)$$

$$v_{0,x}^2 + v_{0,y}^2 = a_x^2 + a_y^2 \cdot t^2 \quad (8.128)$$

$$\frac{v_{0,x}^2 + v_{0,y}^2}{a_x^2 + a_y^2} = t^2 \quad (8.129)$$

$$\sqrt{\frac{v_{0,x}^2 + v_{0,y}^2}{a_x^2 + a_y^2}} = t \quad (8.130)$$

Damit stehen zwei Möglichkeiten zur Verfügung.

$$t = \max\left(\frac{-v_{0,x}}{a_x}, \frac{-v_{0,y}}{a_y}\right) \quad (8.131)$$

$$t = \sqrt{\frac{v_{0,x}^2 + v_{0,y}^2}{a_x^2 + a_y^2}} \quad (8.132)$$

Verwendet wird die Gleichung 8.131, da angenommen wird, dass das Nehmen eines Maximalwerts zweier Divisionen effizienter ist, als die Berechnung einer Wurzel einer Division zweier Additionen.

## 8.5 Herleitung Ereignis - Kollision dynamischer Objekte

In diesem Kapitel erfolgt die Herleitung zur Bestimmung des Kollisionszeitpunkts zweier dynamischer Objekte. Beim Billard sind dies zwei Kugeln, welche in Bewegung sind.

Um die Formel herzuleiten, wird die Ortsgleichung der gleichförmig beschleunigten Bewegung betrachtet:

$$\vec{s}(t) = \frac{1}{2} \cdot \vec{a} \cdot t^2 + \vec{v}_0 \cdot t + \vec{s}_0 \quad (8.133)$$

$$\begin{pmatrix} s(t)_x \\ s(t)_y \end{pmatrix} = \frac{1}{2} \cdot \begin{pmatrix} a_x \\ a_y \end{pmatrix} \cdot t^2 + \begin{pmatrix} v_{0,x} \\ v_{0,y} \end{pmatrix} \cdot t + \begin{pmatrix} s_{0,x} \\ s_{0,y} \end{pmatrix} \quad (8.134)$$

Diese ist für beide Kugeln unterschiedlich:

$$\vec{s}_1(t) = \frac{1}{2} \cdot \vec{a}_1 \cdot t^2 + \vec{v}_1 \cdot t + \vec{s}_1 \quad (8.135)$$

$$\vec{s}_2(t) = \frac{1}{2} \cdot \vec{a}_2 \cdot t^2 + \vec{v}_2 \cdot t + \vec{s}_2 \quad (8.136)$$

Da die Objekte zweidimensional sind, also eine bestimmte Fläche aufweisen, können die Gleichungen nicht einfach gleichgesetzt werden. Der Radius der Kugel muss mitbeachtet werden, weswegen die Gleichungen einen Punkt suchen, wo die Distanz zwischen ihnen dem Kugeldurchmesser entspricht. Dies ist der Ort, wo die Kugeln im zweidimensionalen (auch im dreidimensionalen) Raum kollidieren werden. Dazu wird die Parameterform nach  $\vec{d}$  bestimmt.

$$\vec{d} = \vec{s}_1(t) - \vec{s}_2(t) \quad (8.137)$$

$$\vec{d} = \frac{1}{2} \cdot \vec{a}_1 \cdot t^2 + \vec{v}_1 \cdot t + \vec{s}_1 - \left( \frac{1}{2} \cdot \vec{a}_2 \cdot t^2 + \vec{v}_2 \cdot t + \vec{s}_2 \right) \quad (8.138)$$

$$\vec{d} = \frac{1}{2} \cdot \vec{a}_1 \cdot t^2 + \vec{v}_1 \cdot t + \vec{s}_1 - \frac{1}{2} \cdot \vec{a}_2 \cdot t^2 - \vec{v}_2 \cdot t - \vec{s}_2 \quad (8.139)$$

$$\vec{d} = \frac{1}{2} \cdot (\vec{a}_1 - \vec{a}_2) \cdot t^2 + \vec{v}_1 \cdot t + \vec{s}_1 - \vec{v}_2 \cdot t - \vec{s}_2 \quad (8.140)$$

$$\vec{d} = \frac{1}{2} \cdot (\vec{a}_1 - \vec{a}_2) \cdot t^2 + \vec{v}_1 \cdot t - \vec{v}_2 \cdot t + \vec{s}_1 - \vec{s}_2 \quad (8.141)$$

$$\vec{d} = \frac{1}{2} \cdot (\vec{a}_1 - \vec{a}_2) \cdot t^2 + t \cdot (\vec{v}_1 - \vec{v}_2) + \vec{s}_1 - \vec{s}_2 \quad (8.142)$$

$$\vec{d} = t^2 \cdot \frac{1}{2} \cdot \begin{pmatrix} a_{1,x} - a_{2,x} \\ a_{1,y} - a_{2,y} \end{pmatrix} + t \cdot \begin{pmatrix} v_{1,x} - v_{2,x} \\ v_{1,y} - v_{2,y} \end{pmatrix} + \begin{pmatrix} s_{1,x} - s_{2,x} \\ s_{1,y} - s_{2,y} \end{pmatrix} \quad (8.143)$$

$$(8.144)$$

Die Parameterform wird nach Komponenten aufgeteilt:

$$\begin{pmatrix} d_x \\ d_y \end{pmatrix} = t^2 \cdot \frac{1}{2} \cdot \begin{pmatrix} a_{1,x} - a_{2,x} \\ a_{1,y} - a_{2,y} \end{pmatrix} + t \cdot \begin{pmatrix} v_{1,x} - v_{2,x} \\ v_{1,y} - v_{2,y} \end{pmatrix} + \begin{pmatrix} s_{1,x} - s_{2,x} \\ s_{1,y} - s_{2,y} \end{pmatrix} \quad (8.145)$$

$$d_x = t^2 \cdot \frac{1}{2} \cdot (a_{1,x} - a_{2,x}) + t \cdot (v_{1,x} - v_{2,x}) + s_{1,x} - s_{2,x} \quad (8.146)$$

$$d_y = t^2 \cdot \frac{1}{2} \cdot (a_{1,y} - a_{2,y}) + t \cdot (v_{1,y} - v_{2,y}) + s_{1,y} - s_{2,y} \quad (8.147)$$

Wobei bekannt ist, dass  $\vec{d}$  die Länge  $D$  hat (entspricht dem Kugeldurchmesser):

$$|\vec{d}| = \sqrt{(d_x)^2 + (d_y)^2} = D \quad (8.148)$$

$$(d_x)^2 + (d_y)^2 = D^2 \quad (8.149)$$

Die einzelnen Komponenten werden wie folgt substituiert:

$$\Delta a_x = a_{1,x} - a_{2,x} \quad (8.150)$$

$$\Delta a_y = a_{1,y} - a_{2,y} \quad (8.151)$$

$$\Delta v_x = v_{1,x} - v_{2,x} \quad (8.152)$$

$$\Delta v_y = v_{1,y} - v_{2,y} \quad (8.153)$$

$$\Delta s_x = s_{1,x} - s_{2,x} \quad (8.154)$$

$$\Delta s_y = s_{1,y} - s_{2,y} \quad (8.155)$$

Daraus folgt:

$$d_x = t^2 \cdot \frac{1}{2} \cdot \Delta a_x + t \cdot \Delta v_x + \Delta s_x \quad (8.156)$$

$$d_y = t^2 \cdot \frac{1}{2} \cdot \Delta a_y + t \cdot \Delta v_y + \Delta s_y \quad (8.157)$$

$$d_x^2 = (t^2 \cdot \frac{1}{2} \cdot \Delta a_x + t \cdot \Delta v_x + \Delta s_x)^2 \quad (8.158)$$

$$d_x^2 = t^4 \cdot (\frac{1}{2})^2 \cdot \Delta a_x^2 + 2 \cdot t^3 \cdot \frac{1}{2} \cdot \Delta a_x \cdot \Delta v_x + 2 \cdot t^2 \cdot \frac{1}{2} \cdot \Delta a_x \cdot \Delta s_x + t^2 \cdot \Delta v_x^2 + 2 \cdot t \cdot \Delta v_x \cdot \Delta s_x + \Delta s_x^2 \quad (8.159)$$

$$d_x^2 = (\frac{1}{2})^2 \cdot \Delta a_x^2 \cdot t^4 + \Delta a_x \cdot \Delta v_x t^3 + (\Delta a_x \cdot \Delta s_x + \Delta v_x^2) t^2 + 2 \cdot \Delta v_x \cdot \Delta s_x \cdot t + \Delta s_x^2 \quad (8.160)$$

Aus Gleichung 8.160 folgt für  $d_y^2$ :

$$d_y^2 = (\frac{1}{2})^2 \cdot \Delta a_y^2 \cdot t^4 + \Delta a_y \cdot \Delta v_y t^3 + (\Delta a_y \cdot \Delta s_y + \Delta v_y^2) t^2 + 2 \cdot \Delta v_y \cdot \Delta s_y \cdot t + \Delta s_y^2 \quad (8.161)$$

Aus Gleichung 8.149, 8.160 und 8.161 folgt:

$$\begin{aligned} D^2 = & (\frac{1}{2})^2 \cdot \Delta a_x^2 \cdot t^4 + \\ & \Delta a_x \cdot \Delta v_x t^3 + \\ & (\Delta a_x \cdot \Delta s_x + \Delta v_x^2) t^2 + \\ & 2 \cdot \Delta v_x \cdot \Delta s_x \cdot t + \\ & \Delta s_x^2 + \\ & (\frac{1}{2})^2 \cdot \Delta a_y^2 \cdot t^4 + \\ & \Delta a_y \cdot \Delta v_y t^3 + \\ & (\Delta a_y \cdot \Delta s_y + \Delta v_y^2) t^2 + \\ & 2 \cdot \Delta v_y \cdot \Delta s_y \cdot t + \\ & \Delta s_y^2 \end{aligned} \quad (8.162)$$

Daraus ergibt sich das folgende Polynom vierten Grades:

$$\begin{aligned} D^2 = & (\frac{1}{2})^2 \cdot (\Delta a_x^2 + \Delta a_y^2) \cdot t^4 + \\ & (\Delta a_x \cdot \Delta v_x + \Delta a_y \cdot \Delta v_y) \cdot t^3 + \\ & (\Delta a_x \cdot \Delta s_x + \Delta v_x^2 + \Delta a_y \cdot \Delta s_y + \Delta v_y^2) t^2 + \\ & 2 \cdot (\Delta v_x \cdot \Delta s_x + \Delta v_y \cdot \Delta s_y) \cdot t + \\ & \Delta s_x^2 + \Delta s_y^2 \end{aligned} \quad (8.163)$$

Die Koeffizienten können als Skalarprodukte von Vektoren repräsentiert werden. Dies geschieht durch Ausnutzung der Eigenschaft, dass die quadrierte Norm dem Skalarprodukt des Vektors mit sich selbst entspricht.

$$\vec{a} = \begin{pmatrix} a_x \\ a_y \end{pmatrix} \quad (8.164)$$

$$|\vec{a}| = \sqrt{a_x^2 + a_y^2} \quad (8.165)$$

$$|\vec{a}|^2 = a_x^2 + a_y^2 \quad (8.166)$$

$$|\vec{a}|^2 = \vec{a} \cdot \vec{a} \quad (8.167)$$

$$a_x^2 + a_y^2 = \vec{a} \cdot \vec{a} \quad (8.168)$$

Wird 8.168 auf die Gleichung 8.163 angewendet, resultieren die nachfolgenden Eigenschaften.

$$\Delta a_x^2 + \Delta a_y^2 = \vec{\Delta a} \cdot \vec{\Delta a} \quad (8.169)$$

$$\Delta a_x \cdot \Delta v_x + \Delta a_y \cdot \Delta v_y = \vec{\Delta a} \cdot \vec{\Delta v} \quad (8.170)$$

$$\Delta a_x \cdot \Delta s_x + \Delta v_x^2 + \Delta a_y \cdot \Delta s_y + \Delta v_y^2 = \vec{\Delta a} \cdot \vec{\Delta s} + \vec{\Delta v} \cdot \vec{\Delta v} \quad (8.171)$$

$$2 \cdot (\Delta v_x \cdot \Delta s_x + \Delta v_y \cdot \Delta s_y) = 2 \cdot (\vec{\Delta v} \cdot \vec{\Delta s}) \quad (8.172)$$

$$\Delta s_x^2 + \Delta s_y^2 = \vec{\Delta s} \cdot \vec{\Delta s} \quad (8.173)$$

Eingesetzt in 8.163:

$$D^2 = \left(\frac{1}{2}\right)^2 \cdot (\vec{\Delta a} \cdot \vec{\Delta a}) \cdot t^4 + (\vec{\Delta a} \cdot \vec{\Delta v}) \cdot t^3 + (\vec{\Delta a} \cdot \vec{\Delta s} + \vec{\Delta v} \cdot \vec{\Delta v}) t^2 + 2 \cdot (\vec{\Delta v} \cdot \vec{\Delta s}) \cdot t + \vec{\Delta s} \cdot \vec{\Delta s} \quad (8.174)$$

## 8.6 Herleitung Ereignis - Kollision mit Bande

Das Ziel ist es, die Position einer Kugel zum Zeitpunkt einer Kollision mit einer Bande zu bestimmen.

Die Bande wird über zwei Punkte beschrieben  $R_1$  und  $R_2$  (für Rail).

Auf Basis dieser Punkte kann eine Geradengleichung formuliert werden:

$$\vec{\Delta R} = \vec{R}_2 - \vec{R}_1 \quad (8.175)$$

$$s(\lambda_1) = \vec{R}_1 + \lambda_1 \cdot \vec{\Delta R} \quad (8.176)$$

Weiterhin kann aufgrund der Position  $C$  und der bekannten Geschwindigkeit  $\vec{v}$  der Kugel eine Linie definiert werden:

$$s(\lambda_2) = \vec{C} + \lambda_2 \cdot \vec{v} \quad (8.177)$$

Der Kollisionspunkt mit der Bande ist damit der Schnittpunkt dieser Linie und dem Banden-Liniensegment. Nun kann der Schnittpunkt der beiden Geraden anhand der Formel in Anhang 8.12 bestimmt werden. Dabei sind einige Variablensubstitutionen zu tun:

$$P = \vec{R}_1 \quad (8.178)$$

$$D = \vec{\Delta R} \quad (8.179)$$

$$Q = \vec{C} \quad (8.180)$$

$$V = \vec{v} \quad (8.181)$$

$$(8.182)$$

Damit gilt dann:

$$\lambda_1 = \frac{V_x \cdot P_y - V_y \cdot P_x + Q_x \cdot V_y - Q_y \cdot V_x}{D_x \cdot V_y - D_y \cdot V_x} \quad (8.183)$$

$$\lambda_2 = \frac{P_x \cdot D_y - P_y \cdot D_x - Q_x \cdot D_y + Q_y \cdot D_x}{V_x \cdot D_y - V_y \cdot D_x} \quad (8.184)$$

Es findet nur dann eine Kollision statt, wenn der Nenner der  $\lambda$  ungleich 0 ist.

Da die Bande nur ein Liniensegment darstellt, muss  $\lambda_1$  zwischen 0 und 1 liegen. Ausserdem muss  $\lambda_2$  grösser oder gleich 0 sein, da der Weg der Kugel eine Half-Line darstellt und ansonsten Schnittpunkte hinter der Kugel möglich wären.

Sofern die beiden  $\lambda$  diese Bedingungen einhalten, dann findet eine Kollision statt.

Nun soll der Ort  $s(t)$  und die Zeit  $t$  der Kollision festgestellt werden. Da der Kugelradius berücksichtigt werden muss, werden die beiden Punkte  $R_1$  und  $R_2$  um den Kugelradius zur Tischmitte verschoben. Diese verschobenen Punkte seien  $R'_1$  und  $R'_2$ .

Mithilfe dieser Punkte kann eine Linie definiert werden:

$$\Delta \vec{R}' = \vec{R}'_2 - \vec{R}'_1 \quad (8.185)$$

$$s(\lambda_1) = \vec{R}'_1 + \lambda_1 \cdot \Delta \vec{R}' \quad (8.186)$$

Der Schnittpunkt dieser Linie mit der in 8.177 definierten Linie ist der Kollisionspunkt  $s(t)$  der Kugel. Angewandt auf die in Anhang 8.12 definierte Formel gelten folgende Variablensubstitutionen:

$$P = \vec{R}'_1 \quad (8.187)$$

$$D = \Delta \vec{R}' \quad (8.188)$$

$$Q = \vec{C} \quad (8.189)$$

$$V = \vec{v} \quad (8.190)$$

$$(8.191)$$

Damit gilt dann:

$$\lambda_1 = \frac{V_x \cdot P_y - V_y \cdot P_x + Q_x \cdot V_y - Q_y \cdot V_x}{D_x \cdot V_y - D_y \cdot V_x} \quad (8.192)$$

$$\lambda_2 = \frac{P_x \cdot D_y - P_y \cdot D_x - Q_x \cdot D_y + Q_y \cdot D_x}{V_x \cdot D_y - V_y \cdot D_x} \quad (8.193)$$

Der Ort der Kollision  $s(t)$  kann nun mit  $\lambda_1$  berechnet werden.

Der Zeitpunkt  $t$  der Kollision wird mit der Formel der gleichmäigig beschleunigten Bewegung berechnet.

$$s(\vec{t}) = \frac{1}{2} \cdot \vec{a} \cdot t^2 + \vec{v} \cdot t + \vec{s}_0 \quad (8.194)$$

Die Beschleunigung  $\vec{a}$ , Geschwindigkeit  $\vec{v}$ , Startposition  $\vec{s}_0$  zum Zeitpunkt  $t_0$  und die Position  $\vec{s}_1$  zum Zeitpunkt  $t_1$  sind bekannt.

Da die Startposition der Kugel zum Zeitpunkt  $t_0$  bekannt ist, kann die Formel auch mit der zurückzulegenden Strecke  $\Delta \vec{s}$  definiert werden:

$$t = t_1 - t_0 \quad (8.195)$$

$$\Delta \vec{s} = \vec{s}_1 - \vec{s}_0 \quad (8.196)$$

$$\Delta \vec{s} = \frac{1}{2} \cdot \vec{a} \cdot t^2 + \vec{v} \cdot t + \vec{s}_0 - \vec{s}_0 \quad (8.197)$$

$$\Delta \vec{s} = \frac{1}{2} \cdot \vec{a} \cdot t^2 + \vec{v} \cdot t \quad (8.198)$$

Das zweidimensionale Problem kann auf eine Dimension reduziert werden, indem die Beträge der Vektoren verwendet werden, weil die Vektoren  $\Delta \vec{s}$ ,  $\vec{a}$  und  $\vec{v}$  kollinear sind:

$$\Delta s = |\Delta \vec{s}| \quad (8.199)$$

$$a = -|\vec{a}| \quad (8.200)$$

$$v = |\vec{v}| \quad (8.201)$$

$$|\Delta \vec{s}| = \frac{1}{2} \cdot (-|\vec{a}|) \cdot t^2 + |\vec{v}| \cdot t \quad (8.202)$$

$$\Delta s = \frac{1}{2} \cdot a \cdot t^2 + v \cdot t \quad (8.203)$$

Umgeformt zu einer quadratischen Gleichung folgt:

$$0 = \frac{1}{2} \cdot a \cdot t^2 + v \cdot t - \Delta s \quad (8.204)$$

Diese Gleichung kann nun mit der allgemeinen Lösungsformel für quadratische Gleichungen nach  $t$  gelöst werden [Unk21i]:

$$t_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \quad (8.205)$$

Diese beiden  $t_1$  und  $t_2$  sind nicht zu verwechseln mit den gleichnamigen Variablen zuvor, es handelt sich um Lösungen für  $t$ .

Wobei für die Koeffizienten gilt:

$$a = \frac{1}{2} \cdot a \quad (8.206)$$

$$b = v \quad (8.207)$$

$$c = -\Delta s \quad (8.208)$$

$$(8.209)$$

Die Lösungen lauten daher:

$$\Delta t_1 = \frac{-v + \sqrt{v^2 + 2 \cdot a \cdot \Delta s}}{a} \quad (8.210)$$

$$\Delta t_2 = \frac{-v - \sqrt{v^2 + 2 \cdot a \cdot \Delta s}}{a} \quad (8.211)$$

Die minimale Zeit repräsentiert den Ereigniszeitpunkt, weswegen nur diese verwendet wird.

$$t = \min(t_1, t_2) \quad (8.212)$$

Da es sich bei  $t$  nur um die vergangene Zeit von dem Startzeitpunkt  $t_0$  bis zum Kollisionszeitpunkt  $t_1$  handelt, muss der Startzeitpunkt wieder aufaddiert werden, sofern der Absolute Zeitpunkt relevant ist:

$$t_1 = t_0 + t \quad (8.213)$$

## 8.7 Herleitung Ereignis - Kollision mit Ziel

Um den Kollisionszeitpunkt zu bestimmen, wird in einem ersten Schritt der Schnittpunkt des Weges der Kugel mit dem Zielkreis berechnet. Dazu wird die Kreisgleichung wie in 8.215 verwendet.

$$|X - C|^2 = r^2 \quad (8.214)$$

$$(x - C_x)^2 + (y - C_y)^2 = r^2 \quad (8.215)$$

Hierbei gelten die Definitionen aus 8.216.

$$\hat{d} = \frac{\vec{d}}{|\vec{d}|} \quad (8.216)$$

Die Parameterform der Geradengleichung ist gegeben in 8.218.

$$\vec{d} = \vec{p} + l \cdot \hat{v} \quad (8.217)$$

$$\begin{pmatrix} d_x \\ d_y \end{pmatrix} = \begin{pmatrix} p_x \\ p_y \end{pmatrix} + l \cdot \begin{pmatrix} v_x \\ v_y \end{pmatrix} \quad (8.218)$$

Durch Einsetzen der Gleichung 8.218 in 8.215 folgt 8.219.

$$(p_x + l \cdot v_x - C_x)^2 + (p_y + l \cdot v_y - C_y)^2 = r^2 \quad (8.219)$$

Im nächsten Schritt wird die Gleichung 8.219 nach  $t$  umgeformt.

$$(p_x + l \cdot v_x - C_x)^2 + (p_y + l \cdot v_y - C_y)^2 = r^2 \quad (8.220)$$

$$|\vec{p} + I \cdot \hat{v} - \vec{C}|^2 = r^2 \quad (8.221)$$

$$|I \cdot \hat{v} + \vec{p} - \vec{C}|^2 = r^2 \quad (8.222)$$

$$|I \cdot \hat{v} + (\vec{p} - \vec{C})|^2 = r^2 \quad (8.223)$$

$$(I \cdot \hat{v} + (\vec{p} - \vec{C})) \cdot (I \cdot \hat{v} + (\vec{p} - \vec{C})) = r^2 \quad (8.224)$$

$$I^2 \cdot (\hat{v} \cdot \hat{v}) + 2I \cdot (\hat{v} \cdot (\vec{p} - \vec{C})) + (\vec{p} - \vec{C}) \cdot (\vec{p} - \vec{C}) = r^2 \quad (8.225)$$

$$I^2 \cdot (\hat{v} \cdot \hat{v}) + 2I \cdot (\hat{v} \cdot (\vec{p} - \vec{C})) + (\vec{p} - \vec{C}) \cdot (\vec{p} - \vec{C}) - r^2 = 0 \quad (8.226)$$

Die Gleichung 8.219 ist von einer quadratischen Form und kann mithilfe der Mitternachtsformel gelöst werden.

$$a \cdot I^2 + b \cdot I + c = 0 \quad (8.227)$$

$$a = \hat{v} \cdot \hat{v} = |\hat{v}|^2 = 1 \quad (8.228)$$

$$b = 2 \cdot \hat{v} \cdot (\vec{p} - \vec{C}) \quad (8.229)$$

$$c = (\vec{p} - \vec{C}) \cdot (\vec{p} - \vec{C}) - r^2 \quad (8.230)$$

Die Lösungen lauten demnach:

$$I_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \quad (8.231)$$

$$I_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a} \quad (8.232)$$

Wobei  $b^2 - 4ac$  die Diskrimante ist. Hat diese einen negativen Wert, existiert kein Schnittpunkt, beim Wert 0 gibt es einen Schnittpunkt und bei einem Wert grösser als 0 gibt es zwei Schnittpunkte.

Die Lösung lautet demnach:

$$I_1 = \frac{-(2 \cdot \hat{v} \cdot (\vec{p} - \vec{C})) + \sqrt{4 \cdot ((\vec{p} - \vec{C}) \cdot (\vec{p} - \vec{C}) - r^2)}}{2} \quad (8.233)$$

$$I_2 = \frac{-(2 \cdot \hat{v} \cdot (\vec{p} - \vec{C})) - \sqrt{4 \cdot ((\vec{p} - \vec{C}) \cdot (\vec{p} - \vec{C}) - r^2)}}{2} \quad (8.234)$$

$$I = \max(\min(I_1, I_2), 0) \quad (8.235)$$

Ein  $I$  ist valid, wenn es positiv ist und nicht innerhalb des Ziellochs  $(-I_1, +I_2)$  liegt. Dieser Fall wird nicht vorkommen, da die Kugel bei der ersten Kollision aus dem System entfernt wird. Sollte ein valides  $I$  existieren, kann damit die Distanz  $d$  zwischen der Kugel und dem Schnittpunkt bestimmt werden.

$$d(\vec{I}) = I \cdot \hat{v} \quad (8.236)$$

Anhand der Distanz aus 8.236 kann der Zeitpunkt durch die Gleichung 8.238 bestimmt werden, wobei gilt  $s(\vec{t}) = d(\vec{I})$

$$s(\vec{t}) = \frac{1}{2} \cdot \vec{a} \cdot t^2 + \vec{v} \cdot t \quad (8.237)$$

$$\frac{1}{2} \cdot \vec{a} \cdot t^2 + \vec{v} \cdot t - d(\vec{I}) = 0 \quad (8.238)$$

Da es sich um die Berechnung der benötigten Zeit von der Kugel zum Ziel handelt und keine weiteren Faktoren relevant sind, kann die Berechnung der Zeit entweder auf eine Dimension reduziert werden oder die X- wie auch Y-Dimensionen werden separat berechnet. Dieselbe Sachlage wurde bereits in Kapitel 8.4 erläutert. Da es sich hier um eine quadratische Gleichung handelt, wäre hingegen der Mehraufwand der separaten Betrachtung der X- und Y-Dimensionen höher als die Reduktion auf eine Dimension durch die Betragnahme der Vektoren, weswegen der zweitgenannte Weg verfolgt wird. Demnach ergibt sich die Situation in 8.240.

$$a \cdot t^2 + b \cdot t + c = 0 \quad (8.239)$$

$$a = \frac{1}{2} \cdot -|\vec{a}| \quad (8.240)$$

$$b = |\vec{v}| \quad (8.241)$$

$$c = -|d(\vec{l})| \quad (8.242)$$

Es ergeben sich wiederum zwei Lösungen, wobei nur die der erste Schnittpunkte (die Minimalzeit) relevant ist.

$$t_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \quad (8.243)$$

$$t_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a} \quad (8.244)$$

$$t = \min(t_1, t_2) \quad (8.245)$$

Sollte die Diskriminante kleiner denn 0 sein, bedeutet dies, dass die Geschwindigkeit der Kugel nicht ausreichend ist, um den Zielkreis zu erreichen. Es existiert demnach keine Lösung. Die Lösung lautet demnach:

$$t_1 = \frac{-|\vec{v}| + \sqrt{|\vec{v}|^2 + 2 \cdot -|\vec{a}| |d(\vec{l})|}}{-|\vec{a}|} \quad (8.246)$$

$$t_2 = \frac{-|\vec{v}| - \sqrt{|\vec{v}|^2 + 2 \cdot -|\vec{a}| |d(\vec{l})|}}{-|\vec{a}|} \quad (8.247)$$

$$t = \min(t_1, t_2) \quad (8.248)$$

## 8.8 Herleitung Ereignis - Rollen

Wird die Kugel zentral getroffen, so gleitet sie eine bestimmte Distanz und beginnt erst später zu rollen[Unk13]. Es gelten die nachfolgenden Gleichungen, wobei  $m$  für die Masse und  $r$  den Radius der Kugel,  $J$  für das Trägheitsmoment,  $F$  für die Gleitreibungskraft und  $\mu$  für den Gleitreibungskoeffizienten steht.

$$F_G = F_N = m \cdot g \quad (8.249)$$

$$F = F_N \cdot \mu \quad (8.250)$$

$$F = m \cdot g \cdot \mu \quad (8.251)$$

$$F = m \cdot a \quad (8.252)$$

$$J = \frac{2}{5} \cdot m \cdot r^2 \quad (8.253)$$

Bei gegebenem Trägheitsmoment  $J$ , Radius  $r$ , Gleitreibungskraft  $F$ , und Winkelbeschleunigung  $\alpha$  gilt für den Drehmoment  $M$  [Unk21d]:

$$M = F \cdot r \quad (8.254)$$

$$M = J \cdot \alpha \quad (8.255)$$

Die Rollbedingung[Unk21] bei Radius  $r$ , zurückgelegtem Weg  $X$  und Winkel  $\phi$  ist gegeben durch:

$$X = \phi \cdot r \quad (8.256)$$

Die Winkelbeschleunigung  $\alpha$  ist wie folgt definiert[Unk21], wobei  $\omega$  die Winkelgeschwindigkeit ist:

$$\omega = \frac{\delta \phi}{\delta t} \quad (8.257)$$

$$\alpha = \frac{\delta \omega}{\delta t} = \frac{\delta^2 \phi}{\delta t^2} \quad (8.258)$$

Die Winkelbeschleunigung ist gegeben durch:

$$M = F \cdot r \quad (8.259)$$

$$M = J \cdot \alpha \quad (8.260)$$

$$F \cdot r = J \cdot \alpha \quad (8.261)$$

$$F \cdot r = J \cdot \frac{\delta^2 \phi}{\delta t^2} \quad (8.262)$$

Aus der Winkelbeschleunigung folgt für die erste Ableitung nach der Zeit die Gleichung 8.265

$$F \cdot r \cdot t = J \cdot \frac{\delta^2 \phi}{\delta t^2} \cdot t \quad (8.263)$$

$$F \cdot r \cdot t = J \cdot \frac{\delta \phi}{\delta t} \quad (8.264)$$

$$\frac{\delta \phi}{\delta t} = \omega = \frac{F \cdot r \cdot t}{J} \quad (8.265)$$

Die Rollbedingung gilt nur, wenn die Kugel von Beginn an rollt, ansonsten ist eine konstante Strecke beinhaltet, auf welcher die Kugel nur gleitet[Unk13]. Um den Zeitpunkt zu bestimmen, an dem die Kugel zu rollen beginnt, wird die erste Ableitung nach der Zeit verwendet (siehe Gleichungen 8.256 und 8.265).

$$X = \phi \cdot r \quad (8.266)$$

$$v = \frac{\delta X}{\delta t} = \frac{\delta \phi}{\delta t} \dot{r} \quad (8.267)$$

$$v = \frac{\delta X}{\delta t} = \frac{\delta \phi}{\delta t} \dot{r} = \frac{F \cdot r^2 \cdot t}{J} \quad (8.268)$$

Um  $t$  zu bestimmen, können die Gesetze der gleichmäig beschleunigten Bewegung verwendet werden. Diese ergibt dieselbe Geschwindigkeit.

$$v = \frac{\delta X}{\delta t} = a \cdot t + v_0 \quad (8.269)$$

$$F = m \cdot a \quad (8.270)$$

$$a = \frac{F}{m} \quad (8.271)$$

$$v = \frac{\delta X}{\delta t} = \frac{F}{m} \cdot t + v_0 \quad (8.272)$$

Da die Beschleunigung  $a$  entgegen der Rollrichtung der Kugel wirkt, wird diese noch negativ gemacht:

$$a = \frac{-F}{m} \quad (8.273)$$

$$v = \frac{\delta X}{\delta t} = \frac{-F}{m} \cdot t + v_0 \quad (8.274)$$

Die Gleichungen 8.268 und 8.274 werden gleichgesetzt und nach  $t$  gelöst.

$$\frac{F \cdot r^2 \cdot t}{J} = \frac{-F}{m} \cdot t + v_0 \quad (8.275)$$

$$\frac{F \cdot r^2}{J} = -\frac{F}{m} + \frac{v_0}{t} \quad (8.276)$$

$$\frac{F \cdot r^2}{J} + \frac{F}{m} = \frac{v_0}{t} \quad (8.277)$$

$$t \cdot \left( \frac{F \cdot r^2}{J} + \frac{F}{m} \right) = v_0 \quad (8.278)$$

$$t = \frac{v_0}{\frac{F \cdot r^2}{J} + \frac{F}{m}} \quad (8.279)$$

$$t = \frac{v_0}{\frac{m \cdot g \cdot \mu \cdot r^2}{\frac{5}{3} \cdot m \cdot r^2} + \frac{m \cdot g \cdot \mu}{m}} \quad (8.280)$$

$$t = \frac{v_0}{5 \cdot \frac{g \cdot \mu}{2} + g \cdot \mu} \quad (8.281)$$

$$t = \frac{v_0}{\frac{5}{2} g \cdot \mu + g \cdot \mu} \quad (8.282)$$

$$t = \frac{v_0}{g \cdot \mu \cdot (\frac{5}{2} + 1)} \quad (8.283)$$

$$t = \frac{v_0}{\frac{7}{2} \cdot g \cdot \mu} \quad (8.284)$$

Demnach kann der Zeitpunkt über die Formel 8.285 bestimmt werden.

$$t = \frac{v_0}{\frac{7}{2} \cdot g \cdot \mu} \quad (8.285)$$

## 8.9 Gleitdistanz

Die zurückgelegte Gleitdistanz ohne Rollen einer Kugel, lässt sich mithilfe der Formel zur Bestimmung des Rollzeitpunkts des Kapitels 8.8 herleiten. Dazu wird die gleichmäig beschleunigte Bewegungsgleichung verwendet.

$$F = m \cdot a \quad (8.286)$$

$$s = \frac{1}{2} \cdot a \cdot t^2 + v \cdot t + s_0 \quad (8.287)$$

Da es sich bei  $a$  um eine entgegengesetzte Kraft handelt, wird diese mit  $-1$  multipliziert. Weterhin gilt nun für die Kraft  $F = m \cdot g \cdot \mu$ , wobei es sich bei  $\mu$  um den Gleitreibungskoeffizienten handelt. Wird die Formel des Rollzeitpunkts für  $t$  eingesetzt, resultiert die folgende Gleichung.

$$s = -\frac{1}{2} \cdot \frac{F}{m} \cdot \left(\frac{v_0}{\frac{7}{2} \cdot g \cdot \mu}\right)^2 + v_0 \cdot \frac{v_0}{\frac{7}{2} \cdot g \cdot \mu} \quad (8.288)$$

$$s = -\frac{1}{2} \cdot \frac{F}{m} \cdot \frac{v_0^2}{\frac{49}{4} \cdot g^2 \cdot \mu^2} + \frac{v_0^2}{\frac{7}{2} \cdot g \cdot \mu} \quad (8.289)$$

$$s = -\frac{1}{2} \cdot \frac{F}{m} \cdot \frac{4 \cdot v_0^2}{49 \cdot g^2 \cdot \mu^2} + \frac{2 \cdot v_0^2}{7 \cdot g \cdot \mu} \quad (8.290)$$

$$s = -\frac{F \cdot 4 \cdot v_0^2}{m \cdot 98 \cdot g^2 \cdot \mu^2} + \frac{2 \cdot v_0^2}{7 \cdot g \cdot \mu} \quad (8.291)$$

$$s = -\frac{m \cdot g \cdot \mu \cdot 4 \cdot v_0^2}{m \cdot 98 \cdot g^2 \cdot \mu^2} + \frac{2 \cdot v_0^2}{7 \cdot g \cdot \mu} \quad (8.292)$$

$$s = -\frac{4 \cdot v_0^2}{98 \cdot g \cdot \mu} + \frac{2 \cdot v_0^2}{7 \cdot g \cdot \mu} \quad (8.293)$$

$$s = -\frac{4 \cdot v_0^2}{98 \cdot g \cdot \mu} + \frac{28 \cdot v_0^2}{98 \cdot g \cdot \mu} \quad (8.294)$$

$$s = \frac{-4 \cdot v_0^2 + 28 \cdot v_0^2}{98 \cdot g \cdot \mu} \quad (8.295)$$

$$s = \frac{24 \cdot v_0^2}{98 \cdot g \cdot \mu} \quad (8.296)$$

$$s = \frac{24}{98} \cdot \frac{v_0^2}{g \cdot \mu} \quad (8.297)$$

$$s = \frac{12}{49} \cdot \frac{v_0^2}{g \cdot \mu} \quad (8.298)$$

## 8.10 Herleitung Beschleunigung

Die Beschleunigung  $a$  einer Kugel, welche durch die Rollreibung auf dem Tisch entsteht, kann über die wirkenden Kräfte ausgedrückt werden. Die Länge des Vektors  $\vec{a}$  kann folgendermassen ausgedrückt werden [Unk21k], wenn die Reibungskraft  $F_R$  sowie die Normalkraft  $F_N$  bekannt ist:

$$F_R = m \cdot a \quad (8.299)$$

$$a = \frac{F_R}{m} \quad (8.300)$$

$$F_N = m \cdot g \quad (8.301)$$

$$F_R = F_N \cdot c_R \quad (8.302)$$

$$a = g \cdot c_R \quad (8.303)$$

Die Richtung des Vektors  $\vec{a}$  zeigt in die entgegengesetzte Richtung von  $\vec{v}$ . Dazu wird  $\vec{v}$  normiert. Damit kann  $\vec{a}$  bestimmt werden:

$$\vec{a} = g \cdot c_R \cdot \frac{\vec{v}}{|\vec{v}|} \quad (8.304)$$

## 8.11 Bestimmung Reibungskoeffizient

Um den Reibungskoeffizienten  $c_R$  zu bestimmen, wird ein Versuchsaufbau in Form einer Rampe benötigt, wobei eine theoretische Reibung auf einer perfekt glatten Oberfläche der Rampe von 0 angenommen wird.

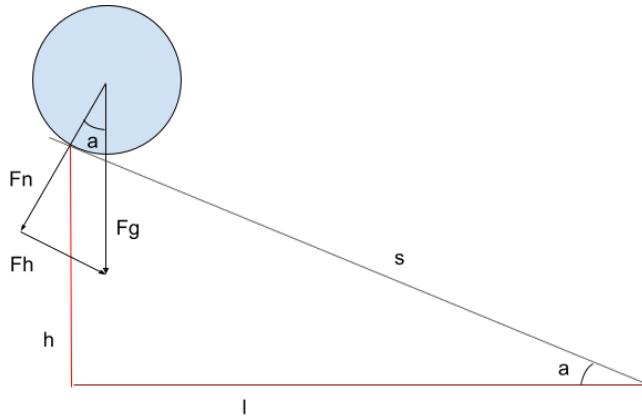


Abbildung 8.3: Modell zur Berechnung der Startgeschwindigkeit auf dem Tisch

In Abbildung 8.3 werden die wirkenden Kräfte und ihr Einfluss veranschaulicht. Es gelten die nachfolgenden Beziehungen.

$$g = 9.81 \frac{m}{s^2} \quad (8.305)$$

$$\alpha = \arctan\left(\frac{h}{l}\right) \quad (8.306)$$

$$F_G = m \cdot g \quad (8.307)$$

$$F_N = \cos(\alpha) * F_G \quad (8.308)$$

$$F_H = \sin(\alpha) * F_G \quad (8.309)$$

$$s = \sqrt{l^2 + h^2} \quad (8.310)$$

$$(8.311)$$

Während die Kugel die Rampe herunterrollt, wird die Formel der gleichmässig beschleunigten Bewegung verwendet. Die Kugel wird auf der Rampe platziert und losgelassen. Das bedeutet, dass sie keine Initialgeschwindigkeit hat. Relevant ist am Ende die zurückgelegte Strecke wie auch die benötigte Zeit.

$$s(t) = s = \frac{1}{2} \cdot a \cdot t^2 + v_0 \cdot t + s_0 \quad (8.312)$$

$$v_0 = 0 \quad (8.313)$$

$$s_0 = 0 \quad (8.314)$$

$$s(t) = s = \frac{1}{2} \cdot a \cdot t^2 \quad (8.315)$$

$$F_H = m \cdot a \quad (8.316)$$

$$a = \frac{F_H}{m} \quad (8.317)$$

$$\frac{1}{2} \cdot \frac{F_H}{m} \cdot t^2 - s = 0 \quad (8.318)$$

Der Zeitpunkt, wo die Kugel das Ende der Rampe erreicht, kann mithilfe der allgemeinen Lösungsformel für quadratische Gleichungen nach  $t$  gelöst werden [Unk21i]:

$$t_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \quad (8.319)$$

$$a = \frac{1}{2} \cdot \frac{F_H}{m} \quad (8.320)$$

$$b = 0 \quad (8.321)$$

$$c = -s \quad (8.322)$$

$$t_{1,2} = \frac{\pm \sqrt{-4ac}}{2a} \quad (8.323)$$

$$(8.324)$$

Es wird nur die positive Lösung in Betracht gezogen.

Die Geschwindigkeit der Kugel am Ende der Rampe ist somit:

$$v(t) = a \cdot t + v_0 \quad (8.325)$$

$$v_0 = 0 \quad (8.326)$$

$$v(t) = \frac{F_H}{m} \cdot t \quad (8.327)$$

Ab diesem Zeitpunkt wird angenommen, dass die Kugel auf dem Tisch rollt und nur noch von der Rollreibung zwischen Kugel und Tisch abgebremst wird. Es muss die von der Kugel bis zum Stillstand zurückgelegte Strecke  $s$  gemessen werden. Weiterhin gelten die Beziehungen in Abbildung 8.4.



Abbildung 8.4: Modell zur Berechnung des Reibungskoeffizienten

Anschliessend kann die Beschleunigung berechnet werden:

$$s(t) = s = \frac{1}{2} \cdot a \cdot t^2 + v_0 \cdot t + s_0 \quad (8.328)$$

$$s_0 = 0 \quad (8.329)$$

$$v_0 = v(t_R) \quad (8.330)$$

$$\frac{1}{2} \cdot a \cdot t^2 + v_0 \cdot t - s = 0 \quad (8.331)$$

$$v(t) = 0 = a \cdot t + v_0 \quad (8.332)$$

$$a \cdot t = -v_0 \quad (8.333)$$

$$t = \frac{-v_0}{a} \quad (8.334)$$

$$\frac{1}{2} \cdot a \cdot \left(\frac{-v_0}{a}\right)^2 + v_0 \cdot \frac{-v_0}{a} - s = 0 \quad (8.335)$$

$$\frac{1}{2} \cdot a \cdot \frac{(-v_0)^2}{a^2} + v_0 \cdot \frac{-v_0}{a} - s = 0 \quad (8.336)$$

$$\frac{1}{2} \cdot \frac{(-v_0)^2}{a} + v_0 \cdot \frac{-v_0}{a} - s = 0 \quad (8.337)$$

$$\frac{1}{2} \cdot (-v_0)^2 \cdot \frac{1}{a} + v_0 \cdot (-v_0) \cdot \frac{1}{a} - s = 0 \quad (8.338)$$

$$\frac{1}{a} \cdot \left( \frac{1}{2} \cdot (-v_0)^2 + v_0 \cdot (-v_0) \right) - s = 0 \quad (8.339)$$

$$\frac{1}{2} \cdot (-v_0)^2 + v_0 \cdot (-v_0) = s \cdot a \quad (8.340)$$

$$a = \frac{\frac{1}{2} \cdot (-v_0)^2 + v_0 \cdot (-v_0)}{s} \quad (8.341)$$

Mit der Beschleunigung kann nun der Rollwiderstandskoeffizient berechnet werden. Da die Kugel auf dem Tisch liegt, ist die Normalenkraft  $F_N$  gleich der Gewichtskraft  $F_G$ .

$$F_R = c_R \cdot F_N \quad (8.342)$$

$$F_R = m \cdot a \quad (8.343)$$

$$F_N = F_G \quad (8.344)$$

$$F_G = m \cdot g \quad (8.345)$$

$$m \cdot a = c_R \cdot m \cdot g \quad (8.346)$$

$$c_R = \frac{m \cdot a}{m \cdot g} \quad (8.347)$$

$$c_R = \frac{a}{g} \quad (8.348)$$

## 8.12 Herleitung Linie-Linie-Schnittpunkt

Der Schnittpunkt zweier Linien ist zu finden. Die Linien  $L_1$  und  $L_2$  seien anhand parametrischer Gleichungen wie folgt definiert:

$$L_1(\lambda_1) : P + \lambda_1 \cdot D \quad (8.349)$$

$$L_2(\lambda_2) : Q + \lambda_2 \cdot V \quad (8.350)$$

Wobei  $P$  und  $Q$  Punkte auf den Linien und  $D$  und  $V$  Richtungsvektoren der Linien sind.

Der Schnittpunkt kann bestimmt werden, indem beide Gleichungen gleichgesetzt werden:

$$L_1(\lambda_1) = L_2(\lambda_2) \quad (8.351)$$

$$P + \lambda_1 \cdot D = Q + \lambda_2 \cdot V \quad (8.352)$$

Komponentenweise formuliert folgt:

$$P_x + \lambda_1 \cdot D_x = Q_x + \lambda_2 \cdot V_x \quad (8.353)$$

$$P_y + \lambda_1 \cdot D_y = Q_y + \lambda_2 \cdot V_y \quad (8.354)$$

Nach Isolation einer Unbekannten auf eine Seite:

$$\lambda_2 = \frac{P_x + \lambda_1 \cdot D_x - Q_x}{V_x} \quad (8.355)$$

$$\lambda_2 = \frac{P_y + \lambda_1 \cdot D_y - Q_y}{V_y} \quad (8.356)$$

Nun können beide Gleichungen gleichgesetzt werden, um die Variable  $\lambda_2$  zu entfernen:

$$\frac{P_x + \lambda_1 \cdot D_x - Q_x}{V_x} = \frac{P_y + \lambda_1 \cdot D_y - Q_y}{V_y} \quad (8.357)$$

Nun kann nach  $\lambda_1$  umformuliert werden:

$$\frac{P_x + \lambda_1 \cdot D_x - Q_x}{V_x} = \frac{P_y + \lambda_1 \cdot D_y - Q_y}{V_y} \quad (8.358)$$

$$(P_x + \lambda_1 \cdot D_x - Q_x) \cdot V_y = (P_y + \lambda_1 \cdot D_y - Q_y) \cdot V_x \quad (8.359)$$

$$P_x \cdot V_y + \lambda_1 \cdot D_x \cdot V_y - Q_x \cdot V_y = P_y \cdot V_x + \lambda_1 \cdot D_y \cdot V_x - Q_y \cdot V_x \quad (8.360)$$

$$\lambda_1 \cdot D_x \cdot V_y - \lambda_1 \cdot D_y \cdot V_x = P_y \cdot V_x - P_x \cdot V_y - Q_y \cdot V_x + Q_x \cdot V_y \quad (8.361)$$

$$\lambda_1 \cdot (D_x \cdot V_y - D_y \cdot V_x) = P_y \cdot V_x - P_x \cdot V_y - Q_y \cdot V_x + Q_x \cdot V_y \quad (8.362)$$

$$\lambda_1 = \frac{P_y \cdot V_x - P_x \cdot V_y - Q_y \cdot V_x + Q_x \cdot V_y}{D_x \cdot V_y - D_y \cdot V_x} \quad (8.363)$$

$$\lambda_1 = \frac{V_x \cdot P_y - V_y \cdot P_x + Q_x \cdot V_y - Q_y \cdot V_x}{D_x \cdot V_y - D_y \cdot V_x} \quad (8.364)$$

Wenn der Nenner der Gleichung 8.364 null ist, dann gibt es keinen Schnittpunkt der beiden Linien und weitere Berechnungen können abgebrochen werden.

Die Variable  $\lambda_2$  kann durch Einsetzen von  $\lambda_1$  in Gleichung 8.355 oder 8.356 gefunden werden.

Die Lösung nach  $\lambda_2$  kann auch analog  $\lambda_1$  umgestellt werden:

$$\lambda_2 = \frac{P_x \cdot D_y - P_y \cdot D_x - Q_x \cdot D_y + Q_y \cdot D_x}{V_x \cdot D_y - V_y \cdot D_x} \quad (8.365)$$

## 8.13 Herleitung Kugeltransformation über Bandenreflektion

Ein Zielpunkt  $C$  wird an einer Bande gespiegelt, was im Punkt  $\bar{C}$  resultiert. Gesucht wird der Punkt  $A$  an der Bande, welcher angespielt werden muss, damit der Zielpunkt  $C$  getroffen wird. Dieser Punkt  $A$  lässt sich über den Schnittpunkt der Geraden, definiert durch die Bande, sowie der Geraden zwischen dem Punkt  $\bar{C}$  und dem Punkt  $B$  bestimmen. Anzumerken sei hier, dass im Falle einer zweidimensionalen Betrachtung die Bande um den Radius in Richtung Zentrum verschoben werden muss, um den korrekten Anspielpunkt  $A$  zu finden.

Es ergeben sich dadurch die beiden Linien:

$$\vec{l} = \vec{\bar{C}} - \vec{B} \quad (8.366)$$

$$L_1 = \vec{B} + \lambda_1 \cdot \vec{l} \quad (8.367)$$

$$\vec{r} = \vec{R}_2 - \vec{R}_1 \quad (8.368)$$

$$L_2 = \vec{R}_1 + \lambda_2 \cdot \vec{r} \quad (8.369)$$

Der Schnittpunkt kann über die in Kapitel 8.12 vorgestellte Formel berechnet werden.

$$\lambda_1 = \frac{r_x \cdot B_y - r_y \cdot B_x + R_{1,x} \cdot r_y - R_{1,y} \cdot r_x}{l_x \cdot r_y - l_y \cdot r_x} \quad (8.370)$$

### 8.13.1 Den gespiegelten Zielpunkt bestimmen

Um den gespiegelten Zielpunkt  $\bar{C}$  zu bestimmen, wird in einem ersten Schritt das Koordinatensystem transformiert, damit der Ursprung bei einem der Eckpunkte an der zu spiegelnden Bande liegt. Sind die Start- und Endpunkte  $R^1$  sowie  $R^2$  gegeben und der Ursprung soll zum Startpunkt  $R^1$  verschoben werden, kann die folgende Transformationsmatrix in homogenen Koordinaten angewendet werden:

$$T^0 = \begin{pmatrix} 1 & 0 & -R_x^1 \\ 0 & 1 & -R_y^1 \\ 0 & 0 & 1 \end{pmatrix} \quad (8.371)$$

Die Transformation für einen Punkt  $C$  in homogenen Koordinaten lautet demnach:

$$C_0 = T^0 \cdot C \quad (8.372)$$

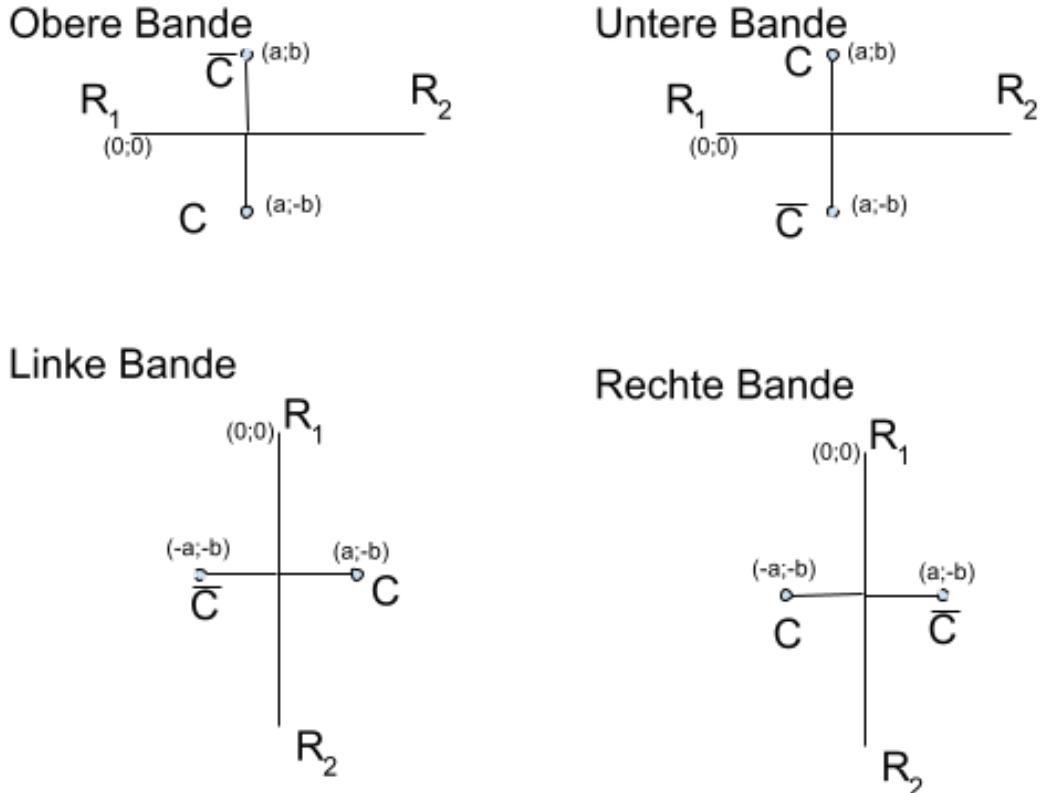


Abbildung 8.5: Spiegelung an den verschiedenen Banden

In einem weiteren Schritt kann der transformierte Zielpunkt  $C$  gespiegelt werden. Diese Spiegelung geschieht durch Multiplikation mit einem Vektor  $\vec{s}$ . Für die Herleitung kann Abbildung 8.5 hinzugezogen werden.

Aus dieser Abbildung ergeben sich die nachfolgenden Beziehungen:

$$\begin{pmatrix} a \\ -b \end{pmatrix} \rightarrow \begin{pmatrix} a \\ b \end{pmatrix} \quad (8.373)$$

$$\begin{pmatrix} a \\ b \end{pmatrix} \rightarrow \begin{pmatrix} a \\ -b \end{pmatrix} \quad (8.374)$$

$$\begin{pmatrix} a \\ -b \end{pmatrix} \rightarrow \begin{pmatrix} -a \\ -b \end{pmatrix} \quad (8.375)$$

$$\begin{pmatrix} -a \\ -b \end{pmatrix} \rightarrow \begin{pmatrix} a \\ -b \end{pmatrix} \quad (8.376)$$

$$(8.377)$$

Anhand der Beziehungen kann für die obere sowie untere Bande folgende Eigenschaft gefunden werden:

$$\vec{\tilde{C}} = \vec{C} \cdot \begin{pmatrix} 1 \\ -1 \end{pmatrix} \quad (8.378)$$

Für die linke sowie rechte Bande lautet die Eigenschaft:

$$\vec{\tilde{C}} = \vec{C} \cdot \begin{pmatrix} -1 \\ 1 \end{pmatrix} \quad (8.379)$$

Um den korrekten Vektor  $\vec{s}$  für die Spiegelung zu wählen, kann eine Funktion über die Normale auf die jeweilige Bande angewendet werden (es wird angenommen, dass die Banden horizontal oder vertikal verlaufen, ansonsten müsste zuerst eine entsprechende Rotation ergänzt werden). Es ergeben sich folgende Beziehungen für die obere und untere Bande. Zu beachten

ist, dass  $R^1$  und  $R^2$  bei den jeweiligen Bandensegmenten auch vertauscht werden können, weswegen die Normalen auch in die entgegengesetzte Richtung zeigen können. Aus diesem Grund werden beide Fälle betrachtet.

$$\begin{pmatrix} 0 \\ -1 \end{pmatrix} \xrightarrow{f} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \quad (8.380)$$

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix} \xrightarrow{f} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \quad (8.381)$$

(8.382)

Dasselbe gilt für die linke und rechte Bande:

$$\begin{pmatrix} 1 \\ 0 \end{pmatrix} \xrightarrow{f} \begin{pmatrix} -1 \\ 1 \end{pmatrix} \quad (8.383)$$

$$\begin{pmatrix} -1 \\ 0 \end{pmatrix} \xrightarrow{f} \begin{pmatrix} -1 \\ 1 \end{pmatrix} \quad (8.384)$$

(8.385)

Da die Abbildungen nicht injektiv, aber surjektiv sind, kann jeweils nur der Absolutwert der Normalen betrachtet werden. Die Beziehungen reduzieren sich auf zwei:

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix} \xrightarrow{f} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \quad (8.386)$$

$$\begin{pmatrix} 1 \\ 0 \end{pmatrix} \xrightarrow{f} \begin{pmatrix} -1 \\ 1 \end{pmatrix} \quad (8.387)$$

(8.388)

Es wird eine lineare Funktion gesucht, welche die Abbildung darstellt:

$$f(\vec{x}) = a \cdot \vec{x} + \vec{b} \quad (8.389)$$

$$\begin{pmatrix} 1 \\ -1 \end{pmatrix} = a \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} + \begin{pmatrix} b_x \\ b_y \end{pmatrix} \quad (8.390)$$

$$\begin{pmatrix} -1 \\ 1 \end{pmatrix} = a \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} b_x \\ b_y \end{pmatrix} \quad (8.391)$$

Daraus ergeben sich vier Gleichungen:

$$1 = b_x \quad (8.392)$$

$$-1 = a + b_y \quad (8.393)$$

$$-1 = a + b_x \quad (8.394)$$

$$1 = b_y \quad (8.395)$$

Demnach lauten die Parameter:

$$b_x = 1 \quad (8.396)$$

$$b_y = 1 \quad (8.397)$$

$$a = -2 \quad (8.398)$$

Anhand der nachfolgenden Funktion kann der Spiegelvektor  $\vec{s}$  bestimmt werden:

$$\vec{s} = \hat{n} \cdot -2 + \vec{1} \quad (8.399)$$

Die Spiegelung kann ebenfalls als Transformation  $S$  in homogenen Koordinaten ausgedrückt werden:

$$S = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (8.400)$$

Abschliessend wird die erhaltene Position in das ursprüngliche Koordinatensystem überführt:

$$T = \begin{pmatrix} 1 & 0 & R_x^1 \\ 0 & 1 & R_y^1 \\ 0 & 0 & 1 \end{pmatrix} \quad (8.401)$$

Die gesamte Transformation kann über die Matrix  $M$  ausgedrückt werden:

$$M = T \cdot S \cdot T^0 \quad (8.402)$$

$$M = \begin{pmatrix} s_x & 0 & R_x^1 - s_x \cdot R_x^1 \\ 0 & s_y & R_y^1 - s_y \cdot R_y^1 \\ 0 & 0 & 1 \end{pmatrix} \quad (8.403)$$

Demnach wird der gespiegelte Punkt  $\bar{C}$  wie folgt bestimmt, wobei die zusätzliche Z-Komponente weggelassen werden kann:

$$\bar{C} = M \cdot C \quad (8.404)$$

## 8.14 Simulation Algorithmusverbesserung

Zurzeit hat der Simulationsalgorithmus zwei Probleme. Er kann einerseits nur ein Ereignis zu einem Zeitpunkt bestimmen und andererseits werden alle Ereignisse ab einem Zeitpunkt neu berechnet. Neu soll demnach die Möglichkeit bestehen, mehrere Ereignisse zum selben Zeitpunkt zuzulassen und alle Ereignisse vorgängig einmal zu berechnen und entsprechend ihrem Eintreten abzuarbeiten. Sobald eine Kugel an einem Ereignis beteiligt ist, werden sämtliche gespeicherten Ereignisse für diese Kugel und ihre Partnerkugel gelöscht. Zudem werden alle Ereignisse bei anderen Kugeln entfernt, die mit an dem Ereignis beteiligten Kugeln zusammenhängen. Danach wird ein neuer Layer zu diesem Zeitpunkt berechnet. Wenn das System nicht konstant ist, werden für sämtliche Kugeln die Ereignisse neu berechnet, die keine mehr haben. Dabei werden auch die gelöschten Ereignisse mit am vorherigen Ereignis unbeteiligten Kugeln ergänzt, sollte es solche geben.

```
struct Event {
    timestamp: datetime
    me: string
    partner: string
    event: [OutOfEnergy, Collision, OutOfSystem, Rolling]
}

Function simulate(start: Layer, constantObjects: list) —> System
    system ← System()
    system ← appendLayer(system, start)
    events: Map[string, Event[]] = Map()
    while / system.isStatic() do
        events ← events(system.lastLayer(), constantObjects, events)
        nextEvents ← nextEvents(events)
        events ← delete(events, nextEvents.second)
        events ← subtractTime(events, nextEvents.first)
        layer ← atMoment(nextEvents)
        system ← appendLayer(system, layer)
    end
    return system

Function events(layer: Layer, constantObjects: list, events: Map[string, Event[]]) —> Map[string, Event[]]
    for object in layer.dynamicObjects() do
        if object.id not in events.keys then
            events = append(outOfEnergy(object), events)
            events = append(collision(object, layer.dynamicObjects()), events)
            events = append(collision(object, layer.staticObjects()), events)
            events = append(collision(object, constantObjects), events)
            events = append(outOfSystem(object, constantObjects), events)
            events = append(rolling(object), events)
        end
    end
    return events

Function append(event: Event, events: Map[string, Event[]]) —> Map[string, Event[]]
    events ← append(event.me, event, events)
    if event.me != event.partner then
        | events ← append(event.partner, event, events)
    end
    return events
```

**Algorithm 7:** Algorithmus zum Aufbau eines physikalischen Systems - Part 1

```

Function delete(events: Map[string, Event[]], nextEvents: Event[]) —> Map[string, Event[]]
  for event in nextEvents do
    events ← drop(event.me, events)
    events ← drop(event.partner, events)
    for pair in events do
      for anEvent in pair.second do
        if anEvent.partner == event.me or anEvent.partner == event.partner then
          cleaned ← drop(anEvent.partner, pair.second)
          events ← replace(pair.first, cleaned, events)
        end
      end
    end
  end
  return events

```

```

Function nextEvents(events: Map[string, Event[]]) —> [float, Event[]]
  nextEvents: [float, Event[]] ← [ $\infty$ , []]
  for pair in events do
    for event in pair.second do
      | nextEvents ← min(nextEvents.first, event)
    end
  end
  return nextEvent.second

```

```

Function subtractTime(events: Map[string, Event[]], time: float) —> Map[string, Event[]]
  for pair in events do
    for event in pair.second do
      | event.timestamp ← event.timestamp - time
    end
  end
  return events

```

**Algorithm 8:** Algorithmus zum Aufbau eines physikalischen Systems - Part 2