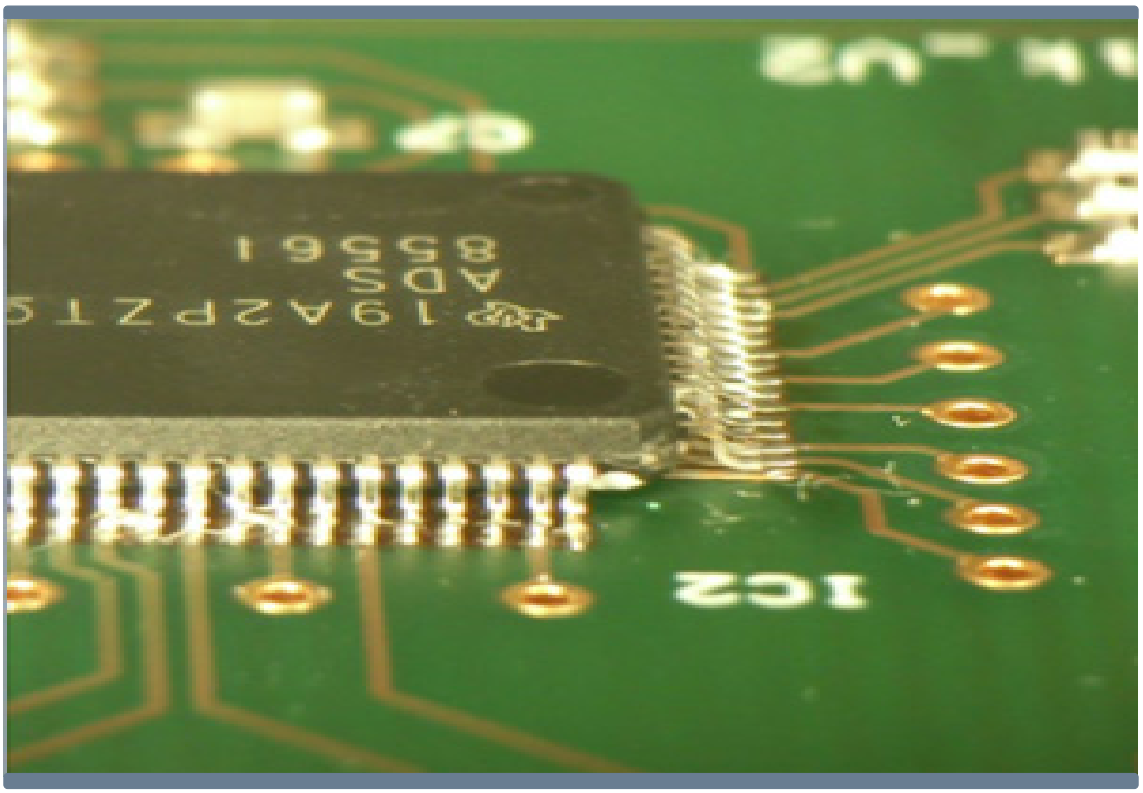
Berner
Fachhochschule



Billiard-AI

Ein intelligenter Billardtisch

Projekt 2

Studienrichtung:	Informatik - Computer Perception and Virtual Reality
Autor:	Lukas Seglias, Luca Ritz
Dozent:	Markus Hudritsch
Experte:	
Datum:	8. Mai 2021

Inhaltsverzeichnis

1 Zusammenfassung

2 Einführung

3 Ziele

Ins Billard-Spiel einzusteigen ist nicht ganz einfach. Zu Beginn kann man schlecht abschätzen, welchen Weg eine Kugel wählen wird, wenn man sie anschlägt und ebenfalls hat man Mühe, den optimalen Stoss zufinden, erst recht, wenn man weiter in die Tiefe sehen will. Wenn es also wichtig wird, die Weisse nach einem Stoss für den Nächsten optimal zu platzieren.

Am Ende wird also ein System entstehen, das einem den optimalen Stoss vorschlägt, basierend auf Kriterien und einer festgelegten Tiefe.

Die Projekt-2 Arbeit hat das grundlegende Ziel der Vorbereitung auf die Bachelor-Thesis. Daher sei zu Beginn erwähnt, dass möglichst viel vorgearbeitet werden kann, auch wenn dies an dieser Stelle keine explizite Erwähnung findet.

Im Wesentlichen geht es aber vor allem um die zugrunde liegenden Basisarbeiten. Diese setzen sich aus den folgenden Teilstücken zusammen:

Aufsetzen des Projekts Dazu gehören nebst Überlegungen zur Architektur und der Implementation davon auch das Aufsetzen der Dokumentation, welche in Latex geschrieben wird.

Aufbau des Systems Dies beinhaltet die Montage der Kamera wie auch des Projektors.

Kalibrierung der Kamera Bestimme die intrinsische Transformationsmatrix, um genaue Bildanalyse betreiben zu können.

Erkennung der Kugelpositionen Die Kugeln sollen einer Position im Pixelkoordinatensystem zugewiesen werden können.

Klassifikation der Kugeln Wurden die Kugeln erkannt, sollen sie entsprechend der Farbe klassifiziert werden.

Übersetzung in internes Koordinatensystem Mittels Marker, welche am Tisch angebracht werden, sollen die auflösungsabhängigen Pixelkoordinaten in ein standardisiertes internes Koordinatensystem überführt werden.

Theoriearbeiten zur Suche eines Stosses Um einen optimalen Stoss zu finden, bedarf es zunächst einiger theoretischen Grundüberlegungen. Dies beinhaltet z.B. einen Algorithmus, um einen Stoss zu finden sowie auch physikalisch korrekt zu beschreiben.

Suche eines einfachen Stosses Sobald der theoretische Ansatz erarbeitet wurde, soll eine erste einfache Suche implementiert werden. Diese soll nur direkte Stösse berücksichtigen. Indirekte über die Bande wie auch über andere Kugeln werden vorerst nicht beachtet. Die Ausgabe soll auch nicht unbedingt über den Projektor erfolgen, eine textuelle Präsentation soll hier genügen.

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 geht 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 wird aber vorläufig weder in Projekt-2 noch in der darauffolgenden Bachelor-Thesis von Relevanz sein.

3.0.1 Planung

Die Planung beinhaltet eine Auflistung der Ziele nach Deadline sowie Bearbeiter.

Ziel	Datum	Bearbeiter
Evaluation Beamer & Kamera	08.03.2021	Lukas & Luca
Überlegungen zum Aufbau	11.03.2021	Lukas & Luca
Entscheid Beamer- Kameratyp	11.03.2021	Lukas & Luca
Theorie der Kamera-Kalibrierung erarbeiten	18.03.2021	Lukas
Beschreibung Such-Algorithmus	18.03.2021	Luca
Theorie der Beamer-Kalibrierung erarbeiten	25.03.2021	Lukas & Luca
Aufbau des Systems	01.04.2021	Lukas & Luca
Definitive Kalibrierung	08.04.2021	Lukas & Luca
Kugel erkennen & Klassifikation	15.04.2021	Luca
Fine-Tuning Kugel erkennen & Klassifikation	29.04.2021	Luca
Marker - Transformation in internes Koordinatensystem	29.04.2021	Lukas
Resultate festhalten Erkennung, Klassifikation & Transformation	06.05.2021	Lukas & Luca
Einfache Suche implementieren	03.06.2021	Lukas & Luca

4 Billiard-AI

In diesem Kapitel werden die gewählte Architektur wie auch die bisher erarbeiteten theoretischen Ansätze behandelt. Weiterhin wird auf verschiedene Aspekte des Aufbaus und dessen Problematik eingegangen.

4.1 Aufbau und Messung

Die Kamera wie auch der Projektor werden mittels eines Baugerüsts über dem Tisch platziert. Dieses ermöglicht die nötige Flexibilität, um die optischen Objekte unabhängig voneinander optimal auszurichten. TODO: Bild einfügen

Der Tisch ist an den Seiten leicht schräg, was ein genaues Messergebnis verunmöglicht. Weiterhin ist er an den Kanten nicht fest und kann leicht deformiert werden. Um diese Ungenauigkeitsfaktoren möglichst gering zu halten, wurden die gegenüberliegenden Tischkanten mit je zwei Holzstücken (Abbildung 4.1, 1), deren Breite individuell über einen Messschieber millimetergenau bestimmt wurde, ausgestattet. Nun kann die Distanz zwischen den Holzstücken gemessen und anschliessend die Breite der Hölzer dazuaddiert werden. Die Messung selbst (Abbildung 4.1, 2 und 3) wird über ein Laser-Messgerät[1] durchgeführt. Die Messungenauigkeit wird mit $\pm 1.5mm$ angegeben. Die Abbildung 4.1 zeigt die Messung in X-Richtung, diese Messung wird ebenfalls in gleicher Art und Weise in Y-Richtung durchgeführt.

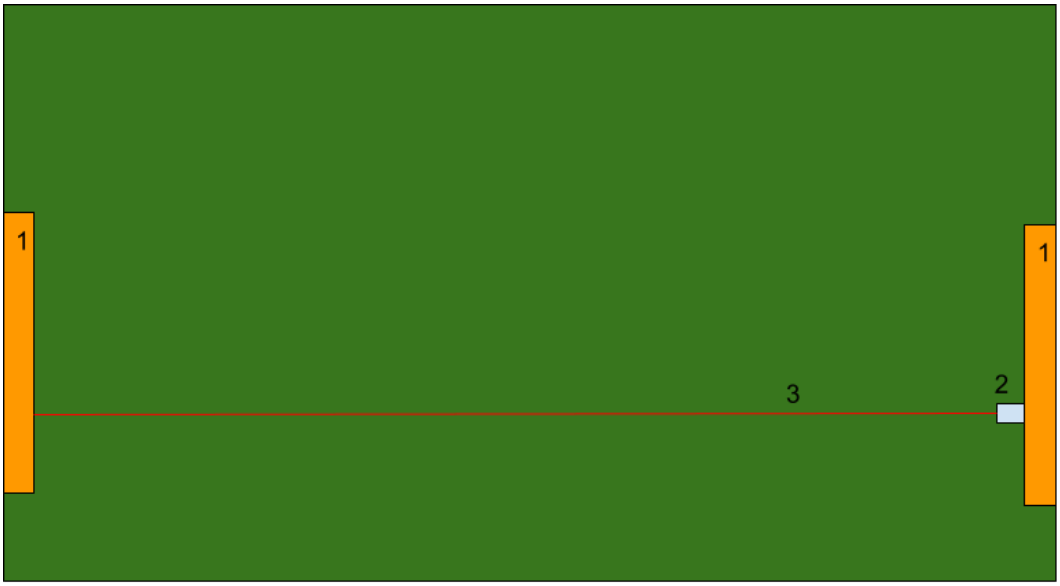


Abbildung 4.1: Messung - Tisch

Um die Messung der Kugeln zu erläutern, muss zuerst das Koordinatensystem eingeführt werden. Die Kugeln selbst werden über eine Kamera aufgenommen und deren Position daher auch in einem Pixelkoordinatensystem bestimmt. Dies ist aus mehreren Gründen ungünstig. Einerseits soll die allgemeine Abhängigkeit zur Kamera vermieden werden, andererseits ist das Pixelkoordinatensystem für die Visualisierung nicht gut geeignet.
Die Kugeln werden also in ein Modellkoordinatensystem übersetzt[2]. Bei diesem befindet sich der Ursprung in der Mitte des Tisches und die X- wie auch die Y-Achse bilden die Breite und Höhe in Millimeter ab.

Um nun die Position der Kugel zu bestimmen, wird zuerst deren Abstand zu den Banden gemessen (siehe Abbildung , 1 und 2). Dies geschieht wie beim Ausmessen des Tisches über ein Holzstück und das Lasermessgerät. Wurden beide Distanzen bestimmt, so kann der Mittelpunkt der Kugel berechnet werden, indem noch der Radius, welcher durch einen Messschieber

[1] Die genauen Spezifikationen sind im Anhang 8.1 ersichtlich.
[2] Siehe Kapitel 4.3

bestimmt wurde, dazuaddiert wird. Es gilt nun, diesen Punkt in das Modellkoordinatensystem zu übersetzen. Dazu wird jeweils die halbe Breite wie auch Höhe (beachte den Ursprung des Modellkoordinatensystems) von dem Mittelpunkt abgezogen. Abschliessend werden noch die korrekten Vorzeichen über den Quadranten bestimmt.

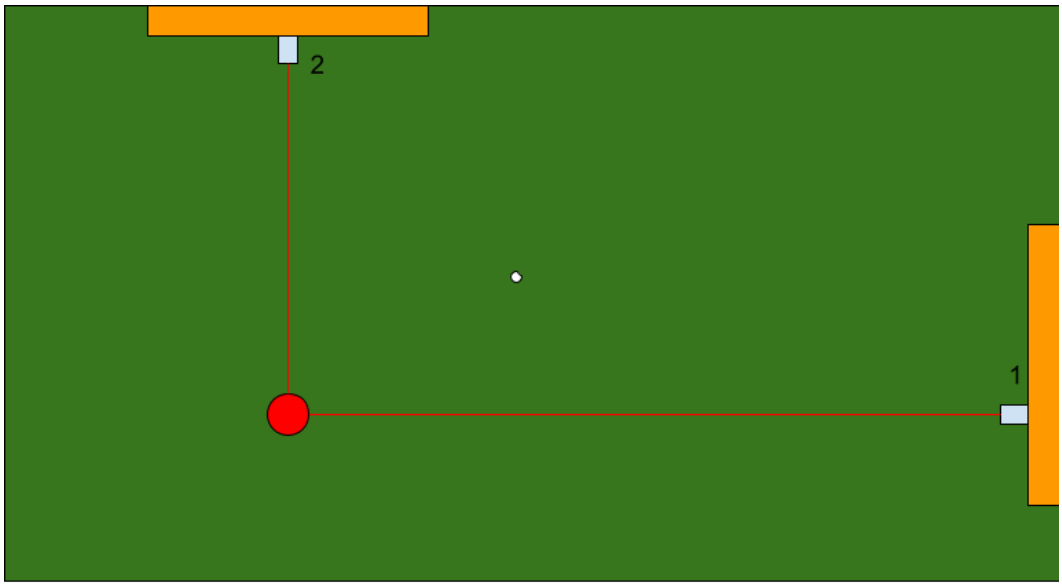


Abbildung 4.2: Messung - Kugel

4.2 Architektur

Die eigentliche Funktionalität wird in einer Core-Library untergebracht, welche nativ in C++ entwickelt wird. Das Endprodukt soll aber weitaus mehr zu bieten haben, wie aus den Zielen ersichtlich wird. Deswegen wird in Unity eine Interaktionsmöglichkeit geschaffen, worüber der Benutzer einerseits die Resultate visualisiert erhält und andererseits der Core-Library seine nächsten Schritte mitteilen kann. Um dies zu erreichen, wird eine weitere native Komponente erstellt, welche die Interaktion zwischen Unity und der Core-Library ermöglicht. Es ist dies eine C/C++-Bibliothek, die ein C-Interface bereitstellt, welches in Unity geladen wird. Unity selbst erhält ebenfalls eine Abstraktionsschicht, um die Nativen auf die Applikationsmodelle zu mappen. Eine Darstellung kann in Abbildung 4.3 entnommen werden.

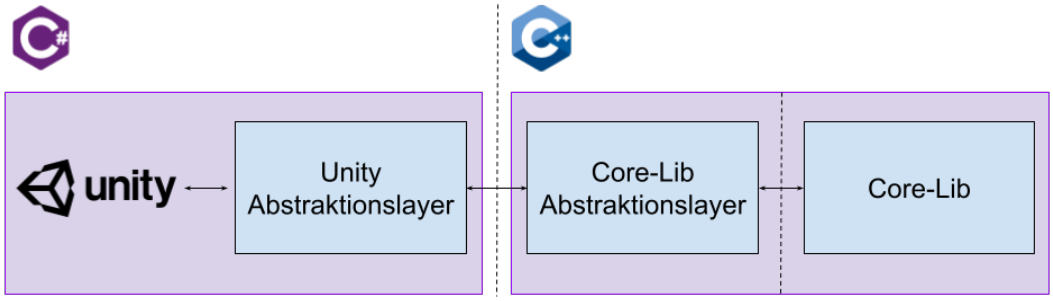


Abbildung 4.3: Grobarchitektur - Applikationsumgebung

Die Core-Library selbst besteht aus mehreren Teilstücken. Es sind dies namentlich mitsamt Funktionalität die folgenden:

billiard_capture Erfasst den aktuellen Spielstand und stellt diesen im OpenCV-Format bereit.

billiard_detection Erstellt aus dem Spielstand eine interne Repräsentation, welchen den Status beschreibt.

billiard_search Verwendet den aktuellen Status wie auch eine zusätzliche Such-Beschreibung, um einen optimalen Stoss zu berechnen.

billiard_physics Stellt Funktionalität bereit, um physikalische Berechnungen durchzuführen.

4.3 Von Pixel zu Modellkoordinaten

TODO: Write ArUco

5 Resultate

Das Kapitel beinhaltet die Beschreibung der Genauigkeiten der Messresultate wie auch der Kugeldetektion.

5.1 Detektionsgenauigkeit

Ein Mass für die Genauigkeit der Detektion bildet die absolute Differenz zwischen der erwarteten wie auch der effektiv detektierten Anzahl der Kugeln. Hierbei werden die Positionen nicht berücksichtigt, die Erkennungsgenauigkeit dieser wird im Kapitel 5.2 beschrieben und millimetergenau bestimmt. Dazu wurden diverse Testbilder[1] aufgenommen.

Bild	Anzahl erwartete Kugeln	Anzahl detektierte Kugeln	Absoluter Fehler
8.1	10	9	10 - 9 = 1

Tabelle 5.1: Detektionsgenauigkeit

Der durchschnittliche Fehler beträgt: ? [Kugeln]

5.2 Übersetzungsgenauigkeit der erkannten Kugeln

Es gilt zwei Aspekte zu beachten. Einerseits soll die Genauigkeit der Übersetzung von Pixel- zu Modellkoordinatensystem, andererseits auch die Genauigkeit der Detektion eines Kugelmittelpunkts bestimmt werden. Um Letzeres zu erreichen, muss Ersteres bekannt sein. Ist bekannt, welcher Fehler bei der Übersetzung geschieht, kann erkannt werden, wie gross der Fehlerbeitrag der Detektion des Kugelmittelpunkts ist. Dies ist wichtig, um bei grösseren Abweichungen von der erwarteten Fehlertoleranz am richtigen Ort anzusetzen. Die Gesamttoleranz der beiden möglichen Fehlerquellen liegt bei 3mm. Diese Fehlertoleranz $F(x, y)$ beschreibt einen Radius um den erwarteten Kugelmittelpunkt. Der Radius setzt sich aus dem Fehler in X- wie auch in Y-Richtung zusammen. Er wird also wie in Gleichung 5.1 beschrieben.

$$F(x, y) = \sqrt{x^2 + y^2} | T(x, y) \leq 3 \tag{5.1}$$

Zusätzlich muss noch der Fehler des Messgeräts berücksichtigt werden, welcher mit $\pm 1.5mm$ angegeben wird. Somit liegt der Gesamtfehlerradius bei $4.5mm$, wobei allfällige Messfehler von Hand nicht berücksichtigt werden.

Um die Genauigkeit zu messen, wurden mehrere Testbilder[2] erstellt. Auf diesen ist jeweils eine Kugel ersichtlich, von der wie in Kapitel 4.1 beschrieben die Modellkoordinate bestimmt wird. Diese Messung dient als Wahrheit zur Verifikation der Berechnungen.

In einem ersten Schritt muss also die Genauigkeit der Umrechung der Koordinatensysteme bestimmt werden. Dazu wird manuell der erwartete Kugelmittelpunkt in Pixelkoordinaten pro Testbild bestimmt. Dieser Punkt wird anschliessend umgerechnet und es wird der absolute Fehlerwert in Millimeter angegeben.

Der durchschnittliche Messfehler beträgt: 1.956 [mm]

In einem zweiten Schritt wird nun noch die Detektion der Kugeln miteinbezogen. Es wird also nicht mehr manuell der Kugelmittelpunkt bestimmt, dies übernimmt nun ebenfalls die Core-Library. Der Fehler wird nun nochmals gleich berechnet wie bei Tabelle 5.2 unter Verwendung derselben Testbilder (siehe Tabelle 5.3, Spalte „Gesamter absoluter Fehler“). Abschliessend kann der Fehler der Detektion in Millimeter über die absolute Differenz des Fehlerwerts ohne Detektion und des Fehlerwerts mit Detektion bestimmt werden (siehe Tabelle 5.3, Spalte „Absoluter Fehler Detektion“).

[1] Alle Testbilder sind im Anhang 8.2.1 ersichtlich
[2] Alle Testbilder sind im Anhang 8.2.2 ersichtlich

Bild	Pixelkoordinaten	Erwartete Modellkoordinaten	Detektierte Modellkoordinaten	Absoluter Fehler
8.2 8	[629, 743]	[-374.85, -216.15]	[-373.308, -214.712]	[-1.542, -1.438], 2.108mm
8.3 9	[814, 287]	[-180.85, 283.15]	[-178.844, 281.705]	[-2.006, 1.445], 2.472mm
8.3 10	[1273, 413]	[337.85, 142.15]	[336.345, 140.874]	[1.505, 1.276], 1.973mm
8.3 11	[1561, 621]	[664.15, -91.85]	[661.979, -93.1876]	[2.171, 1.3376], 2.549mm
8.3 12	[475, 639]	[-560.85, -112.85]	[-561.435, -112.964]	[0.585, 0.114], 0.596mm
8.4 13	[980, 537]	[5.85, 0.15]	[6.50545, 1.31031]	[-0.655, -1.160], 1.332mm
8.4 14	[1640, 225]	[751.15, 350.15]	[750.001, 352.548]	[1.149, -2.398], 2.659mm

Tabelle 5.2: Messresultate Koordinatensystem

Bild	Erwartete Modellkoordinaten	Detektierte Modellkoordinaten	Gesamter absoluter Fehler	Absoluter Fehler Detektion
8.2	[13, 13]	[9, 9]	[4, 4], 5.65mm	5.65mm - 4.24mm = 1.41mm

Tabelle 5.3: Messresultate Detektion

Der durchschnittliche Fehler der Detektion beträgt: ? [mm]
Der durchschnittliche Gesamtmessfehler beträgt: ? [mm]

6 Weitere Arbeiten

7 Fazit

Abbildungsverzeichnis

Tabellenverzeichnis

Glossar

Versionskontrolle

Version	Datum	Beschreibung	Autor
0.1	01.03.2021	Eröffnung Dokumentation	Luca Ritz
0.2	04.03.2021	Definition Ziele	Lukas Seglias & Luca Ritz

8 Anhang

8.1 Messgeräte

Es werden die verwendeten Messgeräte und alle bekannten Spezifikationen erläutert.

8.1.1 Messschieber

TODO: Messgerät beschreiben

8.1.2 Lasermessgerät

TODO: Messgerät beschreiben

8.2 Testbilder

8.2.1 Detektionstestbilder

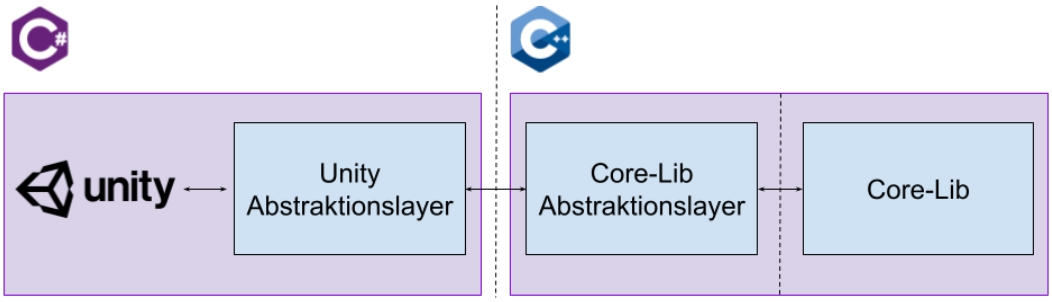


Abbildung 8.1: Detektionsgenauigkeit - Testbild 1

TODO: Add testbilder

8.2.2 Übersetzungsgenauigkeit

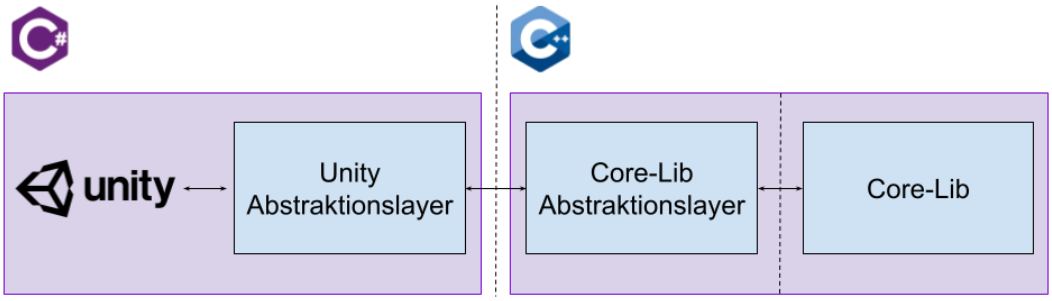


Abbildung 8.2: Übersetzungsgenauigkeit - Testbild 1

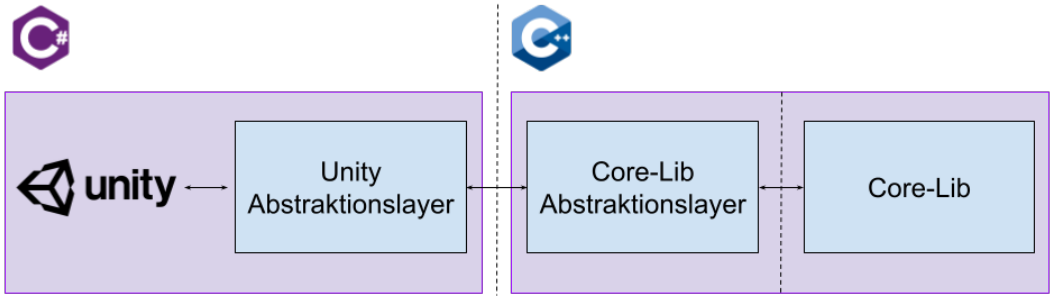


Abbildung 8.3: Übersetzungsgenauigkeit - Testbild 2

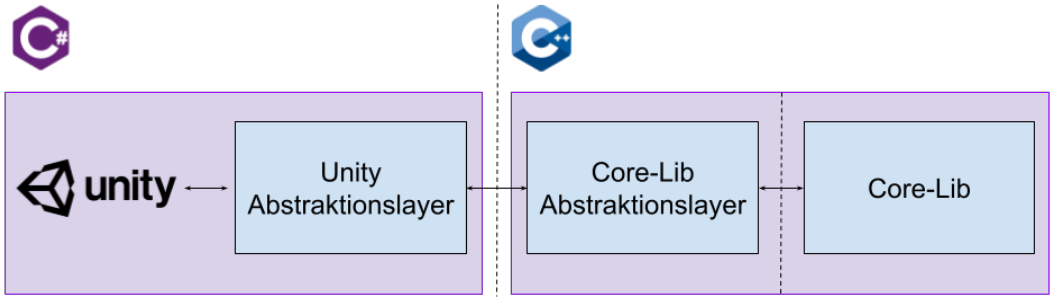


Abbildung 8.4: Übersetzungsgenauigkeit - Testbild 3

TODO: Add testbilder