Version

1

Maturaprojekt 2022/23

Sebastian Haider, Philip Jessner, Simon Frauenschuh  
Betreuer: Dipl. Ing. Karl Heinz Steiner

Ball On Plate

Maturaprojekt 2023

Ball On Plate

© Sebastian Haider, Philip Jessner, Simon Frauenschuh

HTBLuVA Salzburg

Itzlinger Hauptstraße 30

5020 Salzburg

Kontakt: simon\_frauenschuh@icloud.com

# Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, keine anderen als die angegebene Quellen und Hilfsmittel benutzt, und die den benutzten Quellen wörtlich- und inhaltlich entnommenen Stellen als Solche gekennzeichnet habe.

Salzburg, am XXXXXXXXXXXXXXXXXXXXXXXXX

Verfasser:



# DIPLOMARBEIT Dokumentation

|  |  |
| --- | --- |
| Namen der Verfasser | Sebastian Haider, Philip Jessner, Simon Frauenschuh |
| Jahrgang \ Schuljahr | 5BHEL \ 2022/23 |
| Thema der Diplomarbeit | Entwicklung einer Echtzeitauswertung und -berechnung für Laufbahnen bewegter kugelförmiger Objekte  Realisierung einer Konstruktion zur gezielten Steuerung von Kugeln auf ebenen Oberflächen |
| Realisierung | Über Sensoren eingelesene Parameter (Position der Kugel, Neigung der Platte, …) wird die aktuelle sowie zukünftige Position der Kugel bestimmt. Daraufhin werden zwei Servomotoren, und folglich auch die Konstruktion, entsprechend der gewünschten Kugellaufbahn bewegt. |
| Ergebnisse |  |
| Möglichkeiten der Einsichtnahme in die Arbeit | <https://github.com/SimonFrauenschuh/BallOnPlateJava> <https://github.com/SimonFrauenschuh/BallOnPlateC> <https://github.com/SimonFrauenschuh/BallOnPlateFusion> |

|  |  |  |
| --- | --- | --- |
| Approbation (Datum / Unterschrift) | Prüfer | Direktor Abteilungsvorstand |

# DIPLOMA THESIS Documentation

|  |  |
| --- | --- |
| Authors | Sebastian Haider, Philip Jessner, Simon Frauenschuh |
| From academy year | 5BHEL \ 2022/23 |
| Assignment of tasks | Development of a real-time evaluation and calculation for the tracks of moving spherical objects  Realization of a construction for the targeted control of balls on flat surfaces |
| Realization | Parameters read in by sensors (position of the ball, inclination of the plate, ...) are used to determine the current and future position of the ball. Then two servomotors, and consequently also the construction, are moved, according to the desired ball track. |
| Results |  |
| Accessibility of diploma thesis | <https://github.com/SimonFrauenschuh/BallOnPlateJava> <https://github.com/SimonFrauenschuh/BallOnPlateC> <https://github.com/SimonFrauenschuh/BallOnPlateFusion> |

|  |  |  |
| --- | --- | --- |
| Approval (Date / Sign) | Examiner | Head of College Head of Department |

# Inhaltsverzeichnis

[Eidesstattliche Erklärung i](#_Toc87794990)

[DIPLOMARBEIT Dokumentation ii](#_Toc87794991)

[DIPLOMA THESIS Documentation iii](#_Toc87794992)

[Inhaltsverzeichnis iv](#_Toc87794993)

[Einleitung 1](#_Toc87794994)

[Grundansatz 1](#_Toc87794995)

[Benutzerinteraktionen 2](#_Toc87794996)

[Zielsetzung und Aufgabenverteilung 2](#_Toc87794997)

[Sebastian Haider 2](#_Toc87794998)

[Philip Jessner 2](#_Toc87794999)

[Simon Frauenschuh 2](#_Toc87795000)

[Messtechnik 3](#_Toc87795001)

[Wahl der Programmiersprache 3](#_Toc87795002)

[Datenerfassung / Messtechnik 4](#_Toc87795003)

[Gyroskop 4](#_Toc87795004)

[Touchfolie 4](#_Toc87795005)

[C – Code (Logik) 5](#_Toc87795006)

[Wahl der Programmiersprache 5](#_Toc87795007)

[Prinzip der Berechnung / Datenerfassung 5](#_Toc87795008)

[Berechnungen 6](#_Toc87795009)

[Kugelposition 6](#_Toc87795010)

[Gyroskop 6](#_Toc87795011)

[Touchpanel 6](#_Toc87795012)

[Web-Applikation (Back- & Frontend) 7](#_Toc87795013)

[Wahl der Softwaresysteme 7](#_Toc87795014)

[Java EE 7](#_Toc87795015)

[WildFly 7](#_Toc87795016)

[Xhtml, jsf und css 8](#_Toc87795017)

[Backend 8](#_Toc87795018)

[Frontend 8](#_Toc87795019)

[Mechanische Konstruktion 9](#_Toc87795020)

[Wahl der Fertigungsmethode 9](#_Toc87795021)

Kapitel

1

# Einleitung

Das Maturaprojekt “Ball On Plate“ lässt eine Kugel vordefinierte Wege über eine Oberfläche rollen und ermöglicht punktgenaue Bewegung des Körpers sowie die Vorhersage zukünftiger Bewegungen.

Z

iel des Projekts ist es, eine Kugel, die auf einer ebenen Oberfläche mit bekanntem Reibungskoeffizienten liegt, automatisiert entsprechend der Nutzereingabe zu bewegen, bzw. ihre Bewegungen zu berechnen und eine dementsprechende Bewegung der Oberfläche (im Weiteren meist Platte genannt) einzuleiten, um den vordefinierten Pfad (Kreis, stabilisieren auf einem Punkt, …) zu folgen.

icon key

1. Valuable information
2. Test your knowledge
3. Keyboard exercise
4. Workbook review

HIER EIN FOTO DES PROJEKTS EINFÜGEN!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

## Grundansatz

Während der Entwicklung wurden verschiedenste Möglichkeiten der Aufenthalts- und Bewegungsberechnung der Kugel genutzt. Generell galt jedoch, die Position der Kugel nicht nur auszulesen und somit der Realität immer „hinterherzuhinken“ (siehe [Prinzip der Berechnung / Datenerfassung](#_Datenerfassung_/_Messtechnik)), sondern zu berechnen und die Sensordaten der Touchfolie nur zur Bestätigung zu nutzen. Gleiches Prinzip wurde bei der Positionsfeststellung der Platte gewählt. Hierbei wird auf den im Servocontroller gespeicherten Wert zugegriffen, und nur bei einer zu großen Abweichung des vom Gyroskop weitergegebenen Wertes Maßnahmen zur Korrektur eingeleitet, bzw. das Programm abgebrochen zur manuellen Fehlerauswertung.

## Benutzerinteraktionen

Bei dem Projekt wurde vollständig auf physische Komponenten zur Ein- und Ausgabe wie beispielsweise Taster, kabelgebundenes Display, … verzichtet. Hier wurde mit Java EE, PostgreSQL und WildFly auf ein vollkommen Cloud- bzw. Enterprise orientiertes System gesetzt, das ortsungebundene Steuerung sowie zusätzlichen Projektumfang und somit Professionalität gewährt. Zudem wurde so die einfache Erweiterbarkeit gewährleistet, wobei bei steigendem Funktionsumfang keinerlei Änderungen der Hardware vonnöten sind. Da die Steuerung der Kugel automatisiert ist, wird dem Nutzer lediglich ein vordefinierter Funktionsumfang geliefert, aus dessen verschiedenen vorgegebenen Bewegungsabläufen wählen kann. Zudem wird eine 2D Simulation über den Browser ausgegeben, um auch nicht vor Ort anwesenden Benutzern ein Live-Feedback zu ermöglichen.

## Zielsetzung und Aufgabenverteilung

Sebastian Haider

* Konstruktion der mechanischen Grundkonstruktionen (Fusion)
* Konstruktion und Fertigung der benötigten elektronischen Bauteile (Platine (Eagle))
* Realisierung des (grafischen) Frontends (xhtml, jsf, css)

Philip Jessner

* Erstellung des physikalischen Modells
* Ableitung der von Herrn Steiners zur Verfügung gestellten Formel zur Berechnung der Bewegung
* Konstruktion des 4 Draht – I2C Konverters zur Auswertung der Touchpaneldaten (Eagle)
* Realisierung der Verbindung des C-Programms mit der Datenbank
* Entwicklung eines geeigneten Algorithmus zur Vorausberechnung der Kugelposition und benötigten Plattenstellung

Simon Frauenschuh

* Entwicklung des Backends (Java EE, Liquibase) und Frontends (xhtml, jsf)
* Entwicklung einer geeigneten Datenstruktur und Erstellung ebenjener (PostgreSQL)
* Realisierung der Verbindung und Ansteuerung der Sensoren (Gyroskop, Servotreiber, Touchpanel) und Motoren (Servotreiber)
* Entwicklung eines geeigneten Algorithmus zur Vorausberechnung der Kugelposition und benötigten Plattenstellung

Kapitel

2

# Messtechnik

Dieser Teil der Dokumentation befasst sich ausschließlich mit messtechnisch relevanten Problemen und Messverfahren sowie der Datenerfassung (physikalisch gesehen). Auf die Auswertung der Daten wird näher im Kapitel [C – Code (Logik)](#_C_–_Code) eingegangen.

A

ufgrund des großen Umfangs dieses Kapitel wird lediglich auf die für die Realisierung des Projekts notwendigen Probleme und Erkenntnisse eingegangen. Grundprinzipien werden dadurch nur kurz angeschnitten, können jedoch über die verlinkten Quellen nachgelesen werden.

Im weiteren Verlauf dieses Kapitels wird ebenfalls noch ein kurzer Einblick in die Gründe für die Wahl bestimmter Auswertungsmethoden und Messtechniken gegeben. Durch die im Verlaufe des Projekts stark ansteigende benötigte Genauigkeit werden manche Messmethoden mehrmals beschrieben, da sich die Art der Auswertung / Ansteuerung geändert hat.

## Wahl der Programmiersprache

Die ersten Prototypen des Programms wurden der Einfachheit halber mit Python bewerkstelligt. Die Programmstruktur wurde jedoch bald zu komplex und hardwareintensiv, weshalb auf das ressourcenarme C gewechselt wurde, welches es auch ermöglichte, hardwarenäher zu programmieren, was insbesondere der Auswertung der Gyroskopdaten zugutekam, bei denen nun Bitshifting verwendet werden konnte (siehe Funktion „readWord2c“ (lib/gyroscope.h)). Die Verbindung zu den einzelnen Komponenten via I2C konnte ebenso problemlos wie rasch realisiert werden. Der fehlende integrierte Funktionsumfang war für dieses Projekt nicht wirklich von Bedeutung, da der Hauptteil aus Berechnungen besteht, die im Funktionsumfang mitgeliefert sind. Fehlende Funktionen / Bibliotheken wurden aus dem Internet heruntergeladen (pca9685.h) oder selbst geschrieben (alle restlichen).

## Datenerfassung / Messtechnik

Die Datenerfassung beruht auf der Annahme, dass im Regelfall keine Fehler auftreten (siehe [Prinzip der Berechnung / Datenerfassung](#_Datenerfassung_/_Messtechnik)).

Gyroskop

THEORIE HINTER DER AUSLESE MIT GYROSKOP (FUNKTIONSWEISE (MASSEN, DIE SICH BEWEGEN, MAGNETFELD DER ERDE, …)

Touchfolie

In diesem Unterkapitel wird erläutert, wieso ein resistives- und nicht ein kapazitives Touchpanel mit einer vergleichsweise geringen Aktivierungskraft gewählt wurde.

Kapitel

3

# C – Code (Logik)

Dieser Teil der Dokumentation befasst sich vorwiegend mit der Realisierung der Berechnung und Ausführung der Bewegungen, sowie mit den Zusammenhängen zu dem Kapitel [Messtechnik](#_Messtechnik)

D

a die Berechnung der Kugellaufbahn vollständig auf einem Raspberry Pi durchgeführt werden, wurde, wie im Kapitel [Wahl\_der\_Programmiersprache](#Wahl_der_Programmiersprache) näher beschrieben, für alle Berechnung auf das hardwareschonende C gesetzt. Dies umfasst alle Eingaben über den I2C Bus, die Ausgabe über I2C an die Motoren sowie das Hosten eines WildFly Servers für die Web-Applikation (siehe [Web\_Applikation](#Web_Applikation)).

Im weiteren Verlauf dieses Kapitels wird ebenfalls noch ein kurzer Einblick in die Gründe für die Wahl bestimmter Bus-Systeme und Auswertungsmethoden gegeben. Dies dient jedoch nur zur Erläuterung getroffener Entscheidungen aus softwaretechnischer Sicht zu den Messmethoden, die unter [Datenerfassung / Messtechnik](#_Datenerfassung_/_Messtechnik_1) näher erörtert wurden.

## Wahl der Programmiersprache

Die ersten Prototypen des Programms wurden der Einfachheit halber mit Python bewerkstelligt. Die Programmstruktur wurde jedoch bald zu komplex und hardwareintensiv, weshalb auf das ressourcenarme C gewechselt wurde, welches es auch ermöglichte, hardwarenäher zu programmieren, was insbesondere der Auswertung der Gyroskopdaten zugutekam, bei denen nun Bitshifting verwendet werden konnte (siehe Funktion „readWord2c“ (lib/gyroscope.h)). Die Verbindung zu den einzelnen Komponenten via I2C konnte ebenso problemlos wie rasch realisiert werden. Der fehlende integrierte Funktionsumfang war für dieses Projekt nicht wirklich von Bedeutung, da der Hauptteil aus Berechnungen besteht, die im Funktionsumfang mitgeliefert sind. Fehlende Funktionen / Bibliotheken wurden aus dem Internet heruntergeladen (pca9685.h) oder selbst geschrieben (alle restlichen).

## Prinzip der Berechnung / Datenerfassung

Die Datenerfassung beruht auf der Annahme, dass im Regelfall keine Fehler auftreten. Dies spart während der Datenverarbeitung Zeit, da die Motoren schon beginnen können, sich in die gewünschte Position zu bewegen. Sollte jedoch wider Erwarten ein Fehler auftreten, wird – je nach Art und Schweregrad – das Programm abgebrochen und / oder eine Benutzeraktion über das Webinterface gefordert.  
  
Die Position der Konstruktion wird über den Speicher des Servotreibers ausgelesen und mit den Werten des Gyroskops gegengeprüft. Da jedoch aufgrund der Funktionsweise eines Gyroskops – im Rahmen des Budgets und der Rentabilität – nicht die benötigte Genauigkeit erreicht werden kann, wird hier nur auf eine gewisse Toleranz geprüft (beispielweise Wert des Servotreibers +/- 5 Grad), um Fehler wie eine Verkeilung o.Ä. zwar zu erkennen, jedoch nicht der Wert des Gyroskops verwendet.

Ein ähnliches Prinzip wird bei der Auslesung der Touchfolie verwendet. In den Anfangsversionen wurden die Werte der Touchfolie zur Steuerung verwendet. Aufgrund der steigenden Anforderung an die Verarbeitungsgeschwindigkeit wurde im Projektverlauf auf selbiges Prinzip wie das des Gyroskops gewechselt. Im Gegensatz zum Gyroskop kann hier die benötigte Genauigkeit garantiert werden, durch die verarbeitungsbedingte Zeitverschiebung kann hier jedoch nur kontrolliert und nicht ausgelesen werden.

## Berechnungen

Kugelposition

Gyroskop

Touchpanel

Kapitel

4

# Web-Applikation (Back- & Frontend)

Dieser Teil der Dokumentation befasst sich vorwiegend mit der Realisierung der Benutzerinteraktion bzw. Visualisierung über eine Java EE Plattform, sowie mit der Datenbankstruktur, -Verbindung zum Java Code und -Realisierung.

I

n diesem Kapitel wird der komplette Vorgang der Entwicklung des Back- und Frontends dokumentiert und auf einzelne, für die Realisierung des Projekts essentielle, Vorgänge näher eingegangen.

Die Wahl Struktur der Datenbank, sowie damit einhergehende Nach- und Vorteile werden in diesem Kapitel behandelt. Die Entscheidung für PostgreSQL basiert auf während des Pflichtpraktikums bei RedBull gesammelter Erfahrung, nach der für die nächsten Jahre eine Struktur, die den Kriterien PostgreSQLs entspricht, sinnvoll ist, und wird dementsprechend nicht näher erläutert, sondern lediglich die gewählten Dateitypen.

## Wahl der Softwaresysteme

Java EE

Auch, wenn es einfacher zu realisierende Programmiersprachen, oder vorgefertigte Editoren, die ebenso plattformunabhängige Programmierung geboten hätten, gegeben hätte, wurde Java EE aufgrund seiner Etablierung in der Enterprise Umgebung, und somit Zwecks der Übung zu späterem beruflichen Zwecke gewählt. Der damit einhergehende Mehraufwand konnte in Kauf genommen werden, da die Diplomarbeit zu dritt gemacht wurde, und somit die Ressourcen vorhanden waren. Die dadurch gewonnene Möglichkeit zur Realisierung tausender Verbindungen sowie schnellerer Datenverarbeitung wurde nicht genutzt, ist jedoch für eine zukünftige Entwicklung (evtl. Bereitstellung als Reality-Onlinespiel) geeignet und muss nur marginal umgeschrieben werden.

WildFly

Hier gilt Selbiges wie bei der [Wahl von JavaEE](#Wahl_JavaEE). Der Server wurde aufgrund seiner Etablierung in der Wirtschaft gewählt. Der gewonnene Nutzen durch die zusätzlich zur Verfügung stehenden Optionen wird nur insofern genutzt, da das Projekt auf einen „echten“ Webserver deployt werden könnte (Progammierungsstruktur wie bei Salzburg AG und RedBull).  
Auf einfachere Systeme (z.B. Apache Tomcat) konnte verzichtet werden, da der Mehraufwand bei dermaßen kleinen Projekten nicht nennenswert ist.

Xhtml, jsf und css

Beim Frontend wurde ebenfalls auf den Branchenstandard für nicht-grafiklastige Oberflächen gesetzt. Xhtml erzeugt die benötigte Struktur, welche wiederum in das „jsf“-Format konvertiert wird. Css wird lediglich dazu eingesetzt, die Verschachtelung der einzelnen Oberflächen zu realisieren.

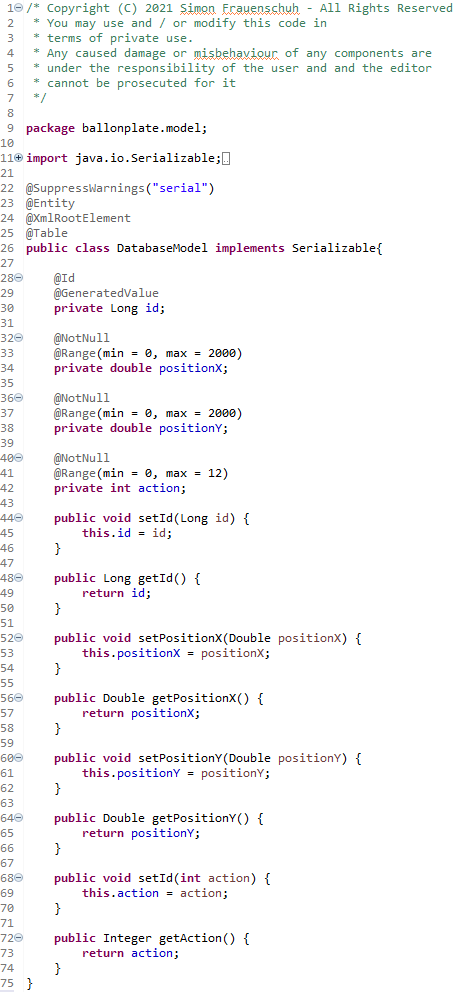
## Backend

In folgenden Unterkapiteln wird ein detaillierter Einblick in die Entwicklung mit Java EE und anderen verwendeten Techniken gewährt.

Programm

Das komplette Programm ist stateless. Das bedeutet, dass bei einem unvermittelten Wechsel des Servers durch einen Load-Balancers die Funktionalität des Programms nicht beeinträchtigt ist, da alle Daten persistent auf einer Datenbank gespeichert sind. Lokale Variablen existieren zwar, werden jedoch für die nächste Instanz (den nächsten Aufruf durch einen Benutzer / eine Datenbank) nicht benötigt. Die Instanziierung und Verknüpfung der Klassen erfolgt, wie in Java EE gefordert, nicht programmierergesteuert, sondern wird von Maven über die Annotationen gehandhabt.

Datenbank



Java Model Klasse

In diesem Unterkapitel wird erläutert, warum die Datenbankstruktur gewählt wurde.

Der Wert „id“ wird automatisch generiert und immer um den wert „1“ inkrementiert. Gestartet wird bei der ID 1, da die ID 0 für einen Datenbanktest benötigt wird.  
Gewählt wurde der Typ „long“, da so eine lange Funktionalität ohne Rücksetzen der Datenbank gewährleistet wird.

„positonXEst“ und „positionYEst“ sind die von der Weboberfläche (oder dem Benutzer, je nach gewähltem Modus) gewünschten Werte.  
Der Datentyp „float4“ wurde gewählt, da die Genauigkeit bereits um ein Vielfaches höher ist, als die in echt Erreichte.

„positonXReal“ und „positionYReal“ sind die realen, vom C-Programm eingetragenen Werte.  
Für die Wahl des Datentyps gilt Selbiges, wie für die gewünschten X- und Y-Positionen.

„mode“ verweist auf einen anderen Table mit den Werten ID (serial2 (Auto)) und Mode (int2). Er beschreibt die gewünschte Funktion (Kreisbewegung, fixe Postion).  
Aufgrund des geringen Funktionsumfangs ist hier in kleiner Datentyp gewählt worden.

„error“ verweist auf einen anderen Table, mit den Werten ID (serial2 (Not Auto)) und Error (int2). Er wird von dem C-Programm gespeist, sollte ein Fehler auftreten.  
Für die Wahl des Datentyps gilt Selbiges, wie für die Moduswahl.

## Frontend

Kapitel

5

# Mechanische Konstruktion

Dieser Teil der Dokumentation befasst sich ausschließlich mit der Realisierung und den mechanischen Problemen der für das Projekt unabdingbaren mechanischen Grundlage.

t

o save time in the future, print a copy of this document. Click **Print** on the **File** menu, and press ENTER to receive all eight pages of examples and instructions. With the printed document in hand, position yourself in normal view to see the style names next to each paragraph. Scroll through the document, and write the style names next to the paragraphs (press CTRL+HOME to reposition yourself at the beginning of the document).

To create a drop cap for the lead paragraph, like the example above, select the letter T, and then type a new letter.

## Wahl der Fertigungsmethode

The “icon key” at left was produced by using the Heading 8 style for the words “icon key” and the List Bullet 5 style for the text below—which uses a Wingdings symbol for the bullet character. To change the bullet symbol, click **Bullets and Numbering** on the **Format** menu. Click **Modify**, and then click the **Bullet** button. Select a new symbol, and then click **OK** twice.