Software Dokumentation

Artillery

# Abstract

Dieses Dokument,

# Inhaltsverzeichnis

[I Abstract 2](#_Toc439806472)

[II Inhaltsverzeichnis 3](#_Toc439806473)

[III Einleitung 5](#_Toc439806474)

[III.1 Änderungen / Version 5](#_Toc439806475)

[III.2 Abkürzungen 5](#_Toc439806476)

[III.3 Gültigkeit und Zweck 6](#_Toc439806477)

[III.4 Ablage, Verweise 6](#_Toc439806478)

[III.5 Mitgeltende Unterlagen 6](#_Toc439806479)

[III.6 Verteiler und Freigabe 6](#_Toc439806480)

[IV Die Software 8](#_Toc439806481)

[IV.1 Ort und Inbetriebnahme 8](#_Toc439806482)

[IV.2 Struktogramm des Main 9](#_Toc439806483)

[IV.3 Verwendete Klassen und deren Parameter und Methoden 12](#_Toc439806484)

[IV.4 Dokumentation Testfälle 13](#_Toc439806485)

[IV.5 Buglist bei der Verison V01.00.00b 13](#_Toc439806486)

[V Zielerreichung im Projekt 14](#_Toc439806487)

[V.1 Must Kriterien 14](#_Toc439806488)

[VI Systemelemente 14](#_Toc439806489)

[VI.1.1 Spielelemente 14](#_Toc439806490)

[VI.2 Need Kriterien 15](#_Toc439806491)

[VI.2.1 Spielelemente 15](#_Toc439806492)

[VI.3 Nice to Have Kriterien 15](#_Toc439806493)

[VI.3.1 Systemelemente 15](#_Toc439806494)

[VI.3.2 Spielelemente 16](#_Toc439806495)

[VII Erfüllung der Anforderungen 17](#_Toc439806496)

[VII.1 Funktionale Anforderungen 17](#_Toc439806497)

[VII.1.1 Need Funktionen 18](#_Toc439806498)

[VII.2 Daten Schnittstellen 19](#_Toc439806499)

[VII.2.1 Need Daten Schnittstellen 19](#_Toc439806500)

[VII.3 Benutzer Schnittstellen 19](#_Toc439806501)

[VII.4 Leistungsmerkmale 20](#_Toc439806502)

[VIII Anhänge 21](#_Toc439806503)

[IX Unterschriften 22](#_Toc439806504)

# Einleitung

## Änderungen / Version

|  |  |  |  |
| --- | --- | --- | --- |
| **Task:** | **Wer:** | **Wann:** | **Bemerkung:** |
| Erstellt  V00.00.01 | WR | 04.01.2016 |  |
| Review  V00.00.02 | JK | 05.01.2016 | Ok |
| Anpassungen  V01.00.00 | WR | 05.01.2016 |  |

## Abkürzungen

|  |  |
| --- | --- |
| **Abkürzung:** | **Bedeutung:** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## Gültigkeit und Zweck

Das Dokument ist dient der Dokumentation des Matlab Projektes und ist für die Software Version 01.00.00 gültig. Zweck dieses Dokumentes ist es eine exakte Beschreibung der Software „Artillerie“ zu liefern den Stan gemäss lastenheft aufzuzeigen und etwaige Bugs zu beschreiben.

## Ablage, Verweise

Die Ablage des gesamten Projektes befindet sich in einem Git- Repository. Berechtigung für das Repository wird allen beteiligten erteilt. Das Projekt unterliegt der „GNU General Public License / Version 3, 29 June 2007“

**HTTPS clone URL**: https://github.com/fhnwRaphi/fhnwMatlProject

## Mitgeltende Unterlagen

|  |  |  |
| --- | --- | --- |
| **Bezeichnung** | **Autor / Datum** | **Bemerkung** |
| Skript mds, 2  Modellierungstechniken,  S. 8 | T. Petermann | Beispiel Schiefer Wurf mit Luftwiederstand |
| Skript mech, Dynamik des Massenpunktes, S.9 | R. Lamprecht | Beispiel Schiefer Wurf |
| Skript matl, | N. Hofmann  P. Coray | gesamtes Skript aus der Vorlesung |
| Latsenheft V01.00.00 | J. Koch  R.Waltenspül | Im Anhang befindlich |

## Verteiler und Freigabe

Der Verteiler des Projektes ist wie folgt:

* Ablage im Git- Repository
* norbert.hofmann@fhnw.ch
* patrick.coray@fhnw.ch
* joel.koch@students.fhnw.ch
* raphael.waltenspuel@students.fhnw.ch
* Abgabe in Printform

Die Bewertung des Projektes erfolgt durch mindestens einen der folgenden Dozenten:

* Prof. Dr. Norbert Hofmann
* Dr. Patrick Coray

# Die Software

## Ort und Inbetriebnahme

Die Software wird in elektronischer Form in folgendem Ordner ausgeliefert:

…AblagePfad…\fhnwMatlProject\srcOO

Dieser Ordner muss vor der Ausführung des Programmes in den Matlab Pfad hinzugefügt werden. >> Rechtsklick auf Ordner srcOO >> Add to Path >> Folder and Subfolders.

Die Dokumentation befindet sich im Ordner:

… AblagePfad …\fhnwMatlProject \Projektmanagement

## Struktogramm des Main

## Verwendete Klassen und deren Parameter und Methoden

## Dokumentation Testfälle

1. Aufstarrtest des Programms
2. Eingabe aller Parameter aus /F30/. (Ohne gegenseitige Abhängigkeiten)
3. Abbrechen nach Eingabe Parameter.
4. Fehlhandlung. Neben Buttons klicken, neben Spielfeld klicken.
5. Eingabe unrealistischer Parameter.
6. Parabel mathematisch nachprüfen. Weite, maximale Höhe.
7. Eingabe währen Schussanimation.
8. Punktezähler. Zählt dieser korrekt hoch.
9. Abbruch / Neustart des Programmes
10. Nächstes Level starten.
11. Win/Loss Bedingungen testen.

## Buglist bei der Verison V01.00.00b

# Zielerreichung im Projekt

Nachfolgend wird beschrieben welche grob umschriebenen Kriterien, welche im Lastenheft in Kapitel 4 definiert sind, implementiert wurden.

## Must Kriterien

# Systemelemente

* Zwei Spieler können gegeneinander spielen.
  + Wurde implementiert
* Die Spieler spielen rundenbasiert an einem Rechner.
  + Wurde implementiert
* Das Spiel ist 2-dimensional.
  + Wurde implementiert

### Spielelemente

* Die Landschaft wird pro Spielrunde neu berechnet.
  + Wurde implementiert
* Die Landschaft wird zufällig generiert.
  + Wurde implementiert
* Der Wind weht und ändert in Abhängigkeit der Zeit die Stärke.
  + Wurde implementiert, wobei die Runde als Zeiteinheit gilt
* Die Spieler werden zufällig platziert.
  + Wurde implementiert
* Jeder Spieler hat einen Tank.
  + Wurde implementiert
* Der Spieler kann die Parameter der Kanone ändern. Diese sind:
  + Stärke der Treibladung (Skalare Grösse)
  + Winkel des Kanonenrohrs (Winkel in Grad)
* oder der Spieler kann diese über eine Mauseingabe per Geschicklichkeit steuern.
  + Wurde implementiert
* Der Spieler gibt den Feuerbefehl
  + Wurde implementiert
* Das Spiel zeichnet die exakte Flugparabel unter Berücksichtigung des Luftwiderstandes. Dies in einer 2 dimensionalen Ebene
  + Wurde implementiert
* Trifft das Geschoss die Umgebung, so bildet sich dort einen Krater.
  + Wurde implementiert
* Trifft das Geschoss den Gegenspieler so wird dieser zerstört. Das Spiel ist zu ende.
  + Wurde implementiert

## Need Kriterien

### Spielelemente

* Das Spiel geht über mehrere Runden.
  + Wurde implementiert
* Trifft der Spieler erhält einen Punkt und eine neue Spielrunde beginnt.
  + Wurde implementiert, wenn alle anderen Spieler zerstört sind.
* Der Spieler der hinten liegt fängt an. In der ersten Runde wird gelost.
  + - Nicht implementiert, es wird immer gelost
* Die Kanonen haben eine Toleranz. Die Schuss- Parabel unterliegt einer Streuung
  + - Nicht implementiert

## Nice to Have Kriterien

### Systemelemente

* Das Spiel kann über das Netzwerk / Internet gespielt werden
  + - Nicht implementiert
* Es können mehr als zwei Spieler Teilnehmen.
  + Wurde implementiert
* Das Spiel kann sowohl Rundenbasiert nacheinander, als auch Rundenbasiert gelichzeitig gespielt werden. (Gleichzeitig heisst, wenn alle Spieler ihren Zug eingegeben haben wird dieser ausgeführt.
  + - Nicht implementiert
* Das Spiel bietet Soundeffekte
  + - Nicht implementiert
* Das Spiel ist 3-dimensional
  + - Nicht implementiert

### Spielelemente

* Der Spieler kann verschiedene Geschosse wählen. Diese können sein:
  1. Grösser Sprengkraft, dafür Schwerer.
  2. Streumunition
  3. …
     + Nicht implementiert, vorbereitet
* Die Kanone kann sich im Verlauf des Spiels erhitzen, dies erhöht die Ungenauigkeit und kann bis zum Ausfall der Kanone führen.
  + - Nicht implementiert, vorbereitet
* Es können sich Schockeffekte bilden. Trifft ein Geschoss in der unmittelbaren Umgebung so muss der Gegenspieler einen Spielzug aussetzen.
  + - Nicht implementiert, vorbereitet
* Die Spieler können mehrere Tanks haben.
  + - Nicht implementiert, vorbereitet
* Die Tanks haben individuelle Beschädigungen
  + - Nicht implementiert, vorbereitet
* Das Spiel bietet weiter Aktionen wie
  1. Fahren
  2. Reparieren
     + Nicht implementiert, vorbereitet
* Weitere Gadgets sind immer möglich…

# Erfüllung der Anforderungen

Die detaillierte Implementation der Anforderung gemäss Lastenheft Kapitel 7 ist unten beschrieben. Die erfüllten Anforderungen wurden Grün Markiert, die nicht erfüllten Anforderungen wurden Rot Markiert. Anmerkungen sind in Blau.

## Funktionale Anforderungen

Nachfolgend sind die funktionalen Anforderungen der Must Elemente beschrieben.

1. Der User startet das Programm in Matlab über den Befehl >> ARTILLERY
2. Das Programm öffnet das Main Menue.
3. Im Main Menue können die grundlegenden Parameter eingestellt werden. Die grundlegenden Parameter sind: Spielmodi (Geschicklichkeit; Taktik); Windstärke (low, medium, high); Einstellung der Berghöhe. (low, medium, high);
4. Starten des Spiels über einen Button.
5. Eine zweidimensionale Landschaft wird zufällig generiert. Dabei wird die gewünschte Berghöhe berücksichtigt.
6. Der Wind weht mit einer Stärke. Der Wind wird im Diagramm als ~~Zahl~~ und Richtungsvektor angezeigt. Der die Windstärke hängt von den eingestellten Parametern Windstärke ab. Anstelle der Zahl wurde dem Windshape eine veränderliche Farbe implementier.
7. Die generierte Landschaft entspricht in der x Achse zwischen 500m und 2000m ca. 1000m
8. Der Panzer jedes Spielers wird zufällig platziert. Dies aber auf der Landschaftsoberfläche und ~~auf einem flachen Plateau.~~ Auf das flache Plateu wurde bewusst verzichtet um später ein realistischen fahren zu ermöglichen.
9. Im Spielmodi Geschicklichkeit, erscheint ein Powerbalken in der Mitte des Feldes.
10. Im Spielmodi Taktik, erscheint ein Eingabefeld für Treibladungsstärke und Winkel. Als Eingabefeld wurde ein Slider verwendet.
11. Im Spielmodi Geschicklichkeit, kann der Spieler welcher am Zug ist, die Parameter der Kanone ändern. Für den Winkel geschieht dies über die Lage des Mauszeigers in Bezug zum Standort der Kanone. Für die Stärke geschieht dies über die Zeit welche der Spieler die Taste drückt.
12. Im Spielmodi Taktik, kann der Spieler welcher am Zug ist, die Parameter der Kanone ändern. Dies geschieht über die Eingabe der Werte für Treibladung in Joul und Winkel in Grad in einem Eingabefeld. Den Feuerbefehl gibt der Spieler durch drücken einer Taste.
13. Das Spiel zeichnet nun ausgehend von dem schiessenden Spieler die exakte Flugparabel des Projektils unter Berücksichtigung des Luftwiderstandes. Es soll dafür eine möglichst exakte Flugbahn gezeichnet werden. Es werden die Methoden aus den Modulen Mechanik und Modellieren dynamischer Systeme verwendet
14. Trifft das Geschoss die Umgebung, so bildet sich dort einen Krater. Dies geschieht durch das ändern des Landschaftsvektor an dieser Koordinate.
15. Trifft das Geschoss den Gegenspieler so wird dieser zerstört, der Spieler welcher trifft, erhält einen Punkt und eine neue Runde beginnt. Im Multiplayer mode erhält der als Leztes verbleibende spieler einen Punkt.

### Need Funktionen

Nachfolgend sind die funktionalen Anforderungen der Need Elemente beschrieben.

1. Ergänzend zu Punkt /F30/ Auswahl / Einstellen der Parameter: Spielmodi (Geschicklichkeit; Taktik); Windstärke (low, medium, high), Windböen (low, medium, high); Anzahl Runden (0 bis 100 St.); Einstellung der Berghöhe. (low, medium, high); Weiter wurde die Wahl eines Planeten auf welchem man spielen kann vorbereitet.
2. Ergänzend zu Punkt /F60/ Der Wind weht und ändert in Abhängigkeit der Zeit die Stärke. Der Wind wird im Diagramm als ~~Zahl~~ und Richtungsvektor angezeigt. Der die Windstärke hängt von den eingestellten Parametern Windstärke ab. ~~Die Änderung der Windstärke während einer Runde hängt von den eingestellten Parametern Windböen ab~~ Die Änderung der Windstärke während einer Runde wird mit einer Standardabweichung mal der zu beginn eingestellten Windstärke neu berechnet.
3. In den Ecken des Spielfeldes werden die Punktestände angezeigt.
4. Befindet sich das Spiel in der ersten Runde, oder herrscht Gleichstand so ermittelt das Programm zufällig welcher Spieler startet.
5. ~~Herrscht kein gleichstand, so starten der Spieler der zurückliegt.~~

## Daten Schnittstellen

Nachfolgend sind die Anforderungen und das Datenmanagement der Must Elemente beschrieben.

1. Das Spiel hat feste Grundeinstellungen.
2. Die Grundeinstellung der Daten sind in jeweiligen Klassen beschrieben. Dies soll das Handling mit grossen Mengen an Spielparametern erleichtern. Zum einen können diese dann einfach als Objekt übergeben werden zum andern können die Parameter einfach über Methoden eingestellt werden.

### Need Daten Schnittstellen

Nachfolgend sind die Anforderungen und das Datenmanagement der Need Elemente beschrieben.

1. ~~Die Spielzüge werden als Datei gespeichert. (Z. Bsp. .txt;.csv;.xml)~~

## Benutzer Schnittstellen

1. Das Spiel wird in einer GUI Dargestellt
2. Die Eingabe kann über die Maus erfolgen
3. ~~Die Eingabe kann über die Tastatur erfolgen.~~

## Leistungsmerkmale

1. Das Spiel muss flüssig laufen. Das bedeutet die Kometenflugbahn darf nicht stocken. Pausen in welchen nichts passiert und keine Eingabe möglich ist sind zu vermeiden.
2. Die Berechnung der Parabel soll < 3s. dauern. Die Animation des geschhosses gehört zum Effekt des Spieles. Die Berechnung der Parabel läuft sehr schnell <1s ab.

# Anhänge

* In Papierform: Ausdruck der Softwarecode
* In elektronischer Form Anhang in Ordner srcOO
  1. Main Function Artillery()
  2. Klasse Figure
  3. Klasse Landscape
  4. Klasse GameParameter
  5. Klasse GameStates
  6. Klasse Player
  7. Klasse FlightPath
* In Papierform: Ausdruck des Lastenheftes
* In elektronischer Form Anhang in Ordner Projektmanagement

# Unterschriften

**Student**

Herr Joel Koch

……………………………………………………………..

Ort, Datum Unterschrift

**Student**

Herr Raphael Waltenspül

……………………………………………………………..

Ort, Datum Unterschrift