Restschnittstelle für einen Kameraroboter

Individuelle Praktische Arbeit

Titelbild

**Lernender**: Maurice Meier  
**Hauptexperte**:   
**Berufsbildner**: Janusz Szymanski  
**Lehrbetrieb**: Fachhochschule für Technik FHNW  
**Startdatum**: 07.03.2022  
**Abgabedatum**:

# Änderungsnachweis

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Datum | Autor | Beschreibung |
| 0.1 | 07.03.2022 | Maurice Meier | Teil 1: Umfeld und Ablauf |
| 0.2 | 09.03.2022 | Maurice Meier | Teil 2: Informieren |

EINTRAGEN!!!!

Inhaltsverzeichnis

[1. Änderungsnachweis 1](#_Toc97473377)

[2. Abstract 4](#_Toc97473378)

[2.1. Ausgangslage 4](#_Toc97473379)

[2.2. Vorgehen 4](#_Toc97473380)

[2.3. Ergebnis 4](#_Toc97473381)

[3. Projektorganisation 5](#_Toc97473382)

[3.1. Involvierte Personen 5](#_Toc97473383)

[3.2. Projektmethode¨ 5](#_Toc97473384)

[3.3. Dokumentationsaufbau 5](#_Toc97473385)

[3.4. Datensicherung 5](#_Toc97473386)

[4. Aufgabenstellung 6](#_Toc97473387)

[4.1. Titel der Arbeit 6](#_Toc97473388)

[4.2. Ausgangslage 6](#_Toc97473389)

[4.3. Detaillierte Aufgabenstellung 6](#_Toc97473390)

[4.4. Mittel und Methoden 6](#_Toc97473391)

[4.5. Vorkenntnisse 6](#_Toc97473392)

[4.6. Vorarbeiten 6](#_Toc97473393)

[5. Zeitplan 7](#_Toc97473394)

[6. Arbeitsjournal 8](#_Toc97473395)

[6.1. Montag, 06.03.2022 Tag 1 8](#_Toc97473396)

[6.1.1. Probleme/Lösungen 8](#_Toc97473397)

[6.1.2. Bemerkungen 8](#_Toc97473398)

[7. Informieren 9](#_Toc97473399)

[7.1. Anforderungsanalyse 9](#_Toc97473400)

[7.2. System 9](#_Toc97473401)

[7.2.1. Camrobot 9](#_Toc97473402)

[7.3. Api Framework 9](#_Toc97473403)

[7.4. Reverse Proxy 9](#_Toc97473404)

[8. Planen 10](#_Toc97473405)

[9. Entscheiden 10](#_Toc97473406)

[10. Realisieren 10](#_Toc97473407)

[11. Kontrollieren 11](#_Toc97473408)

[11.1. Unit Tests 11](#_Toc97473409)

[11.2. Testfallspezifikationen 11](#_Toc97473410)

[11.2.1. Getesteter Teil 11](#_Toc97473411)

[11.3. Testprotokolle 12](#_Toc97473412)

[11.3.1. Testfall Protokoll TP-01 12](#_Toc97473413)

[12. Auswerten 13](#_Toc97473414)

[12.1. Mögliche Erweiterungen 13](#_Toc97473415)

[12.2. Fazit 13](#_Toc97473416)

[12.3. Persönliches Fazit 13](#_Toc97473417)

[13. Quellenverzeichnis 13](#_Toc97473418)

[13.1. Glossar 13](#_Toc97473419)

[13.2. Internetquellen 13](#_Toc97473420)

[13.3. Bildquellen 13](#_Toc97473421)

AKKTUALISIEREN

Teil 1: Obligatorisches Kapitel

# Kurzfassung

## Ausgangssituation

## Umsetzung

## Ergebnis

# Projektorganisation

## Involvierte Personen

**Hauptexperte:**

André Lichtsteiner  
+41 79 469 33 67¨  
andre.lichtsteiner@gmx.ch

**Nebenexperte:**

Jonathan Teig  
Rolltreppen Weg 3  
1235 Nullhausen   
+41 14 411 44 11

**Verantwortliche Fachkraft:**

Martin Gwerder  
Bahnhofstrasse 6  
5210 Windisch  
5.2B15  
+41 65 202 76 81

**Kandidat:**

Maurice Meier  
Bahnhofstrasse 6  
5210 Windisch  
5.2B06  
+41 79 152 52 74

## Projektmethode¨

Gewählte methode und grund

Für die Durchführung dieses Projekt wurde die Projektmethode IPERKA gewählt. IPERKA besteht aus den insgesamt 6 Phasen Informieren, Planen, Entscheiden, Realisieren, Kontrollieren und Auswerten. Aus deren Anfangsbuchstaben das Akronym IPERKA gebildet wurde. Durch diese Methode erhält man einen klaren Ablauf und Struktur über das gesamte Projekt hinweg. Ausserdem wurde diese Methode bereits in Projekten in der Schule vom Kandidaten verwendet. (Ich person ??)

## Dokumentationsaufbau

Struktur der Der Doku wie viele Teile was ist in diesen Teilen

Die Dokumentation ist insgesamt in 2 Teile gegliedert. Teil 1 Obligatorisches Kapitel beinhaltet allgemeine Informationen zur Arbeit sowie eine Kurzfassung, Auflistung aller beteiligten, Die Aufgabenstellung und Zeitplan mit Tagesjournalen.

Teil 2 Projekt-Dokumentation ist nach IPERKA strukturiert und aufgebaut. Jede Phase wird von einem Kapitel abgedeckt und darin die Tätigkeiten während der Phase dokumentiert und festgehalten. Am Ende befindet sich das Glossar so wie die Quellenverzeichnisse.

## Datensicherung

Wie wird das Projekt gesichert und wann

Die Datensicherung wird mithilfe von Täglichen commits auf ein Github repo so wie Das speichern des Projekts auf einem zusätzlichen USB Stick damit immer ein backup mit der momentan aktuellsten Version vorhanden ist.

# Aufgabenstellung

Bei diesen Angaben halte ich mich and die eingetragenen Informationen des Fachverantwortlichen auf PKOrg. Kleinere Änderungen wurden nur vorgenommen, um Rechtschreibfehle zu korrigieren oder die Formatierung anzupassen.

## Titel der Arbeit

Restschnittstelle für einen Kameraroboter

## Ausgangslage

CamBot ist ein für diese Arbeit neu entwickelter Roboter. Er ist in der Lage eine Kamera, um einen 3D Drucker frei und repetierbar zu positionieren. Dieser Roboter soll in OctoPrint einbindbar sein und 3D-taugliche Daten sowie Hyperlapses (Zeitraffer mit Kamerabewegung) liefern, die mittels der in der Intel Realsense d435 und Meshroom (Fotogrammmetrie) gewonnen werden. Diese Arbeit wird parallel mit der Arbeit von Semjon Buzdin durchgeführt. Weil Semjon die 3D-Rekonstruktion macht und Maurice die Kontrolle des Roboters wird für die Arbeit eine API vorgegeben.

## Detaillierte Aufgabenstellung

Ziel dieser Arbeit ist es CamBot zu steuern. Dazu wird ein Dienst namens "cambotmanager" geschrieben. Der Dienst läuft auf einem Raspberry PI mit Raspian (oder Ubuntu) und stellt für die Ansteuerung des CamBots eine API auf der Basis von Rest nach aussen zur Verfügung.

Die RestAPI ist spezifiziert auf Swagger unter der URL <https://app.swaggerhub.com/apis/mgwerder/cambot/0.0.1>.

Der Roboter ist mittels G-Codes ansteuerbar (GRBL-Basis; https://github.com/grbl/grbl/wiki).   
Der Roboter hat 3 Achsen. Die A-Achse ist die Rotationsachse um dem Roboter um den 3D-Drucker zu steuern (Kreis; ca. 270°; X-Achse auf GRBL; 1mm entspricht einem Grad). Die Y-Achse steuert den Roboter in der Höhe (Linearer Spindelantrieb; ca. 40cm; Y-Achse auf GRBL). Die B-Achse steuert den Kamera-Tilt des Roboters (Rotation; ca. -90°-90°; Z-Achse auf GRBL; 1mm entspricht einem Grad).

Ziel ist es die oben spezifizierte API zu realisieren (Swagger-Link) und zu schützen.

Alle Calls, die ein Zipfile retournieren haben ein Zipfile, das Strukturiert ist wie folgt: <taskname> --- snapshots --- <iso8601-timestamp>--- metadata.ini and images (png or jpg) Im File Metadata sind mindestens folgende Informationen enthalten:  
 - Zeit des Snapshots   
- Filename und Typ des Snapshots (Achtung: Es könnte Snapshots mit Tiefen und RGB-Informationen gleichzeitig geben)   
- Genaue Position des Roboters  
 - Status am Ende des Vorganges (OK oder Fehlercode und Text)

Zusätzlich ist eine kleine UI vorhanden (Web) welches es erlaubt (ebenfalls Passwortgeschützt) den Roboter manuell zu steuern und seine Werte anzuzeigen.

Die vom Roboter abgeholten Bilder werden mit den Metadaten in der EXIF-Struktur angereichert.

## Mittel und Methoden

PK ORG

## Vorkenntnisse

* Maurice hat ca. ein Monat im Voraus Zugriff auf den Roboter und kann bereits mit ihm arbeiten
* Er hat im Monat vor der IPA mit Python gearbeitet (1.5d pro Woche) und vorher auch schon damit Erfahrungen gesammelt (Namentlich: Flask, Zip-Files und RestAPI).
* Er kennt den Raspberry-PI

## Vorarbeiten

* Der Kandidat hat eine kurze Ausbildung mit Zustandsmaschinen vor der Arbeit gehabt.
* Der Kandidat hat bereits einen Reverse-Proxy realisiert und darüber eine Benutzerauthentifizierung gemacht auf der Basis von Apache2.
* Der Kandidat hat bereits den Roboter mittels Python angesteuert.
* Der Kandidat hat bereits eine sehr einfache RestAPI auf Python realisiert.
* Der Kandidat hat bereits Dateihandling geübt (Files erstellen, verschieben, löschen)
* Der Kandidat hat bereits Daten in die EXIF-Informationen geschrieben.

# Zeitplan

EINFÜGEN

# Arbeitsjournal

## Montag, 06.03.2022 / Tag 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 2h | 2h | Zeitplan | Zeitplan erstellen | Zeitplan wurde erstellt |
| 2h | 2h | Informieren | Anforderungsanalyse | Anforderungen wurden aus der Aufgabenstellung abgeleitet. |
| 20 min | 10min | Datensicherung | GitHub erstellen | Github repo wurde erstellt. |
| 2h | 2h | Informieren | System Aufbau festhalten | Das System wurde analysiert und als Diagramm festgehalten unter ([Informieren/Systemaufbau](#_Systemaufbau)) |
| 1h 20 min | 2h | Dokumentation | Titel und Teil 1 | Titel und Layout sowie ein Grossteil von Teil 1 Fertig. |
| 15 min | 10 min | Tages Journal | Häutiges Journal einfüllen | Journal wurde geführt |
| 5 min | 5 min | Datensicherung | Alles auf GitHub und USB-Sichern | Daten wurden Kurz vor 17:00 auf GitHub gepushed und auf USB gespeichert. |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
| Was Muss alles im UI möglich sein? | Allfällige Fragen wurden festgehalten, damit diese Matin Gwerder gestellt werden können. |
| Eine Eigene .ini Datei Pro Bild oder Pro Zipfile? | **||** |
|  |  |

### Bemerkungen

Martin Gwerder erteilte uns den Auftrag für die IPA bereits am FR. Abend (04.03.2022) da er am ersten Tag verhindert war.  
  
Der Start verlief wie geplant. Ich konnte nahezu alle Dinge, welche ich mir vorgenommen habe, fertig stellen bis auf 2 Teile im 1.Teil der Dokumentation.

## Mittwoch, 08.03.2022 / Tag 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 1.5h | ~1.5h | Experten Besuch | Experten Besuch um 08:30-10:00 | Alle offenen Fragen geklärt.  Individuelle Bewertungskriterien definiert.  Präsentation Datum festgelegt |
| 2h |  | Informieren | Informieren fertig stellen |  |
| WnZ. |  | Planen | Mit Planen beginnen |  |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
| Fragen von Montag, 06.03.2022 sind im Tagesjournal. | Beim Expertengespräch mit Martin Gwerder geklärt |
| Termin mit Martin Gwerder  Anforderungs Analyse, Entscheidungen, aufkommende Fragen | Termin mit Martin Gwerder am Donnerstag 10.03.2022 15-16 Uhr |
|  |  |

### Bemerkungen

An diesem Tag war das erste Expertengespräch zusammen mit Herr Lichtsteiner und Martin Gwerder. Im Verlauf dieser Besprechung konnte ich alle meine offenen Fragen klären so wie Unklarheiten bei der Auftragsgebung und den Individuellen Bewertungskriterien klären.

Teil 2: Projekt-Dokumentation

# Informieren

## Anforderungsanalyse

Die Anforderungen wurden aus der Aufgabenstellung im PKOrg abgeleitet und nach den verschiedenen Bestandteilen des Projekts sortiert z.B. Reverse Proxy.   
Jede Anforderung erhielt eine eindeutige Bezeichnung, diese richten sich nach dem folgenden Schema

* ANF-RA-#: Anforderungen an die Rest API
* ANF-UI-#: Anforderungen an das UI
* ANF-D-#: Anforderungen an die Datenstruktur

|  |  |
| --- | --- |
| Anforderung | Beschreibung |
| ANF-RA-1 | Die RestAPI verfügt über einen Zugriffsschutz mit Basic oder Digest-Authentifikation via Reverse-Proxy. |
| ANF-RA-2 | Die Rest API verfügt über einen Zugriffsschutz aufgrund der autorisierten IP-Adressen. |
| ANF-RA-3 | Die Rest API ist nach Spezifikation auf Swagger implementiert |
| ANF-UI-1 | Die UI Muss mit einem Passwort geschützt werden |
| ANF-UI-2 | Über muss der momentane Status ersichtlich sein |
| ANF-UI-3 | Der Roboter soll über das UI zurückgesetzt werden können. |
| ANF-UI-4 | Das Aktuelle Inventar soll im UI aufgelistet sein |
| ANF-UI-5 | Elemente aus dem Inventar sollen über das UI heruntergeladen werden können. |
| ANF-D-1 | Die Bilder sind mit Metadaten in der EXIF-Struktur angereichert |
| ANF-D-2 | Die Bilder werden von der API als Zipfile retourniert |
| ANF-D-3 | Im Zip File ist eine metadata.init Datei vorhanden |
| ANF-D-4 | Im File metadata.init sind mindestens folgende Informationen enthalten: - Zeit des Snapshots - Filename und Typ des Snapshots (Achtung: Es könnte Snapshots mit Tiefen und RGB-Informationen gleichzeitig geben) - Genaue Position des Roboters - Status am Ende des Vorganges (OK oder Fehlercode und Text) |
| ANF-D-5 | Die Bilder werden von der API als Zipfile retourniert |
| ANF-D-6 | Alle Calls, die ein Zipfile retournieren haben ein Zipfile, das Strukturiert ist wie folgt: <taskname> --- snapshots --- <iso8601-timestamp>--- metadata.ini and images (png or jpg) |
|  |  |

## Systemaufbau

Das System ist so wie im unten gezeigten Diagramm aufgebaut. Es soll aus einem Raspberry 3, auf welchem ein Reverse Proxy eingerichtet ist. Über diesen soll man auf den Cambotmanager oder das UI zugreifen können. Der Cambotmanager dient als Verbindung zum Cambot. Von aussen wird entweder über den Cambotprocessor welcher in der IPA von Semjon Buzdin erarbeitet wird oder manuell zugegriffen.

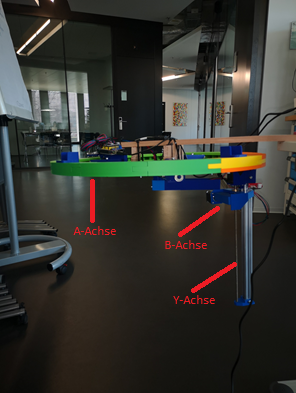
Diagram

Description automatically generated

### Camrobot

Der Cambot besitzt 3 Achsen, mit welchen die Kamera positioniert wird. Die A-Achse ist die Rotationsachse, um die Kamera um den 3d-Drucker zu rotieren und entspricht der X-Achse in GRBL. Ein Lineare Spindelantrieb denkt die Y-Achse ab. Die B-Achse kontrolliert die Neigung der Kamera und kann in GRBL über die Z-Achse angesteuert werden.

A picture containing floor, indoor

Description automatically generated

## Api Framework

Während des Informierens sammelte ich auch Infos über verschiedene Frameworks um mit Python API’s zu erstellen. Die Infos habe ich von verschiedenen Webseiten beschafft, diese habe ich im Quellenverzeichnis festgehalten.

Infos über verschidene pyhon api frameworks/ info herkunft

### Django Rest

Mit Django Rest können einfach Rest-APIs erstellt werden. Django ist umfangreich dokumentiert und besitzt eine aktive Onlinecommunity. Django Rest unterstützt auch mehrere integrierte Authentifizierungsrechtlinien.

### Flask Restful

Flask ist ein einfaches Framework mitwelchem schnelle Rest-APIs erstellt werden. Genau wie Django besitzt auch Flask eine umfangreiche Dokumentation so wie eine Aktive Onlinecommunity. -Flask hat den zusätzlichen Vorteil

### Fast API

Fast API ist eine der schnellsten API-Frameworks. Das Fast bezieht sich auf die Anzahl Queries pro Sekunde, sondern auch die benötigte Zeit, um eine laufende API zu erstellen.

## Reverse Proxy

Um eine gute Entscheidung zu treffen informierte ich mich auch über verschiedene Möglichkeiten einen Reverse Proxy auf einem Respberry Pi zu erstellen.

Infos über verschiedene Reverse Proxys für raspberry/ info herkunft

### NGINX VS Apache

NGINX und Apache sollen in etwa die gleiche Komplxität beim einrichten besitzen. NGINX sei allerdings besitzt NGINX einen kleineren Memory Fussabdruck als Apache und ist somit leistungs freundlicher. Für einem Beitrib auf einem Raspberry wäre das von Vorteil da wir etwas begrentzt sind.

https://serverfault.com/questions/143238/nginx-vs-apache-as-reverse-proxy-which-one-to-choose

# Planen

## Cambotmanager

## Reverse Proxy

## UI

# Entscheiden

Festhalten der getroffenen Entscheidungen so wie Kommunikation mit Arbeitgeber

# Realisieren

# Kontrollieren

## Unit Tests

## Testfallspezifikationen

### Getesteter Teil

#### Testfall Tf-GT-01

|  |  |
| --- | --- |
| Testfall | Tf-GT-01 |
| Anforderung |  |
| Kurzbeschreibung |  |
| Anforderung |  |
| Eingabe | Erwartete Ausgabe |
|  |  |

## Testprotokolle

### Testfall Protokoll TP-01

#### Kennung

#### Testumgebung

#### Durchgeführte Testfälle

|  |  |  |
| --- | --- | --- |
| Testfall | Ergebnis | Bemerk |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

#### Kommentar¨

#### Autogramm

# Auswerten

## Mögliche Erweiterungen

## Fazit

## Persönliches Fazit

# Quellenverzeichnis

## Glossar

## Internetquellen

* <https://geekflare.com/de/python-frameworks-for-apis/> (API Framework)

## Bildquellen