# Anforderungsdokument

## Must-Have-Liste:

### Grundlegende Navigation:

* + Programmierung von Bewegungsabläufen (vorwärts, rückwärts, drehen).
  + Fähigkeit, sich autonom auf vordefinierten Pfaden zu bewegen.

### Hinderniserkennung und -vermeidung:

* + Integration von Ultraschall- oder Infrarotsensoren zur Abstandsmessung.
  + Implementierung von Algorithmen zur Echtzeit-Hindernisvermeidung.

### Grundlegende Sensorik:

* + Nutzung von Sensoren zur Erkennung von Wänden und Personen.
  + Verarbeitung der Sensordaten zur Entscheidungsfindung.

### Hardware-Integration:

* + Verbindung des Raspberry Pi mit Motoren, Sensoren und ggf. Aktoren.
  + Sicherstellung der Energieversorgung und Stabilität des Systems.

### Programmierung in C++:

* + Verwendung von C++ zur Entwicklung effizienter und stabiler Software.
  + Strukturierter und kommentierter Code für bessere Wartbarkeit.

### Mechanischer Arm:

* + Entwicklung und Integration eines funktionierenden Greifarms.
  + Programmierung der Greiffunktion für das Aufnehmen und Ablegen.

### Dokumentation:

* + Ausführliche Beschreibung des Projektverlaufs und der technischen Umsetzung.
  + Erklärung der Herausforderungen und deren Lösungen.

## Nice-to-Have-Liste:

### Erweiterte Navigation:

• Implementierung von Karten und Pfadplanung (z.B. A\*-Algorithmus).

• Fähigkeit, dynamisch neue Routen zu berechnen.

### Maschinelles Lernen:

* + Einsatz von Machine-Learning-Algorithmen zur besseren Hinderniserkennung.
  + Training des Systems zur Erkennung spezifischer Objekte oder Personen.

### Kameraintegration:

* + Nutzung einer Kamera zur Bildverarbeitung.
  + Objekterkennung mittels Computer Vision (z.B. OpenCV).

### Benutzeroberfläche:

* + Entwicklung einer GUI zur Überwachung und Steuerung des Roboters.
  + Anzeige von Sensordaten und Systemstatus in Echtzeit.

### Netzwerkkommunikation:

* + Fernsteuerung oder -überwachung über WLAN oder Bluetooth.
  + Möglichkeit zur Aktualisierung der Software über das Netzwerk.

### Sicherheitsfunktionen:

* + Not-Aus-Schalter und Sicherheitsprotokolle.
  + Meldung von Fehlfunktionen

## Use-Cases

### Grundlegende Navigation

* **Akteure:** Nutzer, Roboter
* **Beschreibung:** Der Roboter soll sich entlang eines definierten Pfades bewegen.
* **Hauptszenario:**
* Nutzer startet den Roboter.
* Roboter liest den Pfad ein und beginnt zu navigieren.
* Bei Hindernissen stoppt der Roboter oder weicht aus.
* Roboter erreicht den Zielpunkt.
* **Alternative Szenarien:**
* Hindernis nicht erkannt: Roboter meldet Fehler und stoppt.
* Pfad nicht vollständig definiert: Roboter kehrt zum Ausgangspunkt zurück.

### Hinderniserkennung und -vermeidung

* **Akteure:** Nutzer, Roboter, Hindernis
* **Beschreibung:** Roboter erkennt und vermeidet Hindernisse während der Navigation.
* **Hauptszenario:**
* Roboter liest Sensordaten ein.
* Hindernis wird erkannt und lokalisiert.
* Roboter berechnet eine Ausweichroute.
* Roboter setzt die Navigation fort.
* **Alternative Szenarien:**
* Hindernis blockiert den Weg vollständig: Roboter meldet Fehler.
* Sensorfehler: Roboter wechselt in einen Sicherheitsmodus und stoppt.

### Greifarmsteuerung

* **Akteure:** Nutzer, Roboter, Objekt
* **Beschreibung:** Roboter nutzt den Greifarm, um Objekte aufzunehmen und abzulegen.
* **Hauptszenario:**
* Nutzer gibt den Befehl zum Aufnehmen eines Objekts.
* Roboter lokalisiert das Objekt mit der Kamera.
* Greifarm greift das Objekt.
* Roboter transportiert das Objekt und legt es am Zielort ab.
* **Alternative Szenarien:**
* Objekt nicht lokalisiert: Roboter meldet Fehler.
* Greifarmmechanik blockiert: Roboter versucht, das Problem zu lösen oder meldet Fehler.

### Objekterkennung mit KI

* **Akteure:** Nutzer, Roboter, Kamera
* **Beschreibung:** Roboter identifiziert ein bestimmtes Objekt mithilfe von OpenCV.
* **Hauptszenario:**
* Roboter aktiviert die Kamera.
* KI analysiert das Bild und identifiziert das gewünschte Objekt.
* Objektkoordinaten werden an die Navigationssteuerung übermittelt.
* Alternative Szenarien:
* Objekt nicht erkannt: Roboter scannt die Umgebung erneut.
* Kameraausfall: Roboter meldet den Fehler.

### Systemintegration

* **Akteure:** Nutzer, Roboter
* **Beschreibung:** Alle Module arbeiten zusammen, um das gewünschte Ziel zu erreichen.
* **Hauptszenario:**
* Nutzer startet das Gesamtsystem.
* Roboter erkennt das Zielobjekt, navigiert, greift das Objekt und transportiert es zum Zielort.
* Roboter meldet erfolgreiche Ausführung.
* **Alternative Szenarien:**
* Ein Modul fällt aus: Roboter meldet den Fehler und wechselt in den Sicherheitsmodus.
* Ziel nicht erreichbar: Roboter kehrt zum Ausgangspunkt zurück.