



## Inhaltsverzeichnis

Allgemeines .....	3
Unser Projekt „LeapMotionRobot“ .....	3
Erklärung .....	3
Welche Anforderungen sollen erfüllt werden? .....	3
Vorgehensweise .....	3
Die Hardware.....	3
LeapMotion .....	3
Der Arduino .....	3
Das Motorshield .....	4
Das Bluetoothshield .....	4
Der Distanzmesser.....	4
Der „Roboter“ .....	4
Die Software .....	4
C#.....	4
Erklärung .....	4
Anforderungen an das Programm .....	4
Code.....	4
Erklärung/Interpretation des Codes.....	4
Arduino .....	6
Erklärung .....	6
Anforderungen an das Programm .....	6
Code.....	6
Erklärung/Interpretation des Codes.....	6
Testläufe .....	6
Testlauf 1 .....	6
Testlauf 2 .....	6
Testlauf 3 .....	6
Testlauf 4 .....	6
Testlauf 5 .....	7
Testlauf 6 .....	7

# Allgemeines

## Unser Projekt „LeapMotionRobot“

### Erklärung

Ein von uns entworfener Roboter soll ferngesteuert werden können. Als Antrieb dienen zwei Gleichstrommotoren, welche über einen Arduino angesteuert werden können. Ebenfalls zum Ansprechen der Motoren wird ein Motorshield benötigt, welches als Steckbrett verwendet wird und auf den Arduino hinaufgesteckt wird. Für die Kommunikation wird auf der Arduino-Seite ein Bluetoothshield verwendet, welches die Bewegungsdaten in Empfang nimmt.

Die Fernkommunikation erfolgt clientseitig mittels einer LeapMotion (wird noch erklärt), die über Bluetooth Bewegungsdaten der Hand weiter an das Bluetoothshield sendet.

### Welche Anforderungen sollen erfüllt werden?

Der Roboter soll nur mithilfe einer Hand gesteuert werden. Je nach Neigung der Hand (nach links oder rechts bzw. nach vorne oder hinten) soll der Roboter entweder nach links oder rechts bzw. nach vorne oder zurück fahren. Ist die Hand nicht geneigt, soll der Roboter stehen bleiben.

In weiterer Folge soll noch realisiert werden, dass der Roboter mit verschiedenen Geschwindigkeiten fahren kann.

Des Weiteren soll der Roboter gegen kein Hindernis (Mauer, Stuhl, Tisch, etc.) fahren.

### Vorgehensweise

Zu Beginn musste recherchiert werden, was alles notwendig war, um das geplante Projekt starten zu können. Dazu wurde auch ein Pflichtenheft erstellt. Dann wurden die Arbeiten aufgeteilt in Arduino – Programmierung, C# - Programmierung (für LeapMotion) und zusammenfügen der Hardware.

Anfangs wurden Testprogramme geschrieben und getestet, diese wurden dann schrittweise zusammengefügt. Fehler wurden solange ausgebessert, bis die Muss-Kriterien erfüllt waren.

Danach wurden noch die Kann-Kriterien ausgearbeitet und in die bereits existierenden Programme implementiert.

## Die Hardware

### LeapMotion

Die LeapMotion ist ein Sensor, entwickelt von der gleichnamigen Firma, welcher die Neigung, Position, Konturen und Bewegungen einer menschlichen Hand erkennen kann. Über eine spezielle Software werden die angeführten Aspekte am Desktop angezeigt.

Die LeapMotion kann man mit einer Kinect – Steuerung (für die Spielekonsole Xbox) verglichen werden.

### Der Arduino

Für unser Projekt wird ein Arduino Uno verwendet.

Dieser Mikrocontroller bekommt Bewegungsdaten der Hand über das Bluetoothshield, welche dann der Arduino in Steuersignale für die Motoren umwandelt. Diese Steuersignale gibt der Arduino danach weiter an das Motorshield.

## Das Motorshield

Dies ist eine spezielle Hardware, welche die Steuersignale, die vom Arduino weitergegeben wurden, verarbeitet und an die Motoren weiterleitet. Je nach Signal werden beide Motoren eingeschaltet, nur der rechte Motor oder nur der linke Motor.

## Das Bluetoothshield

Ist die Schnittstelle zwischen dem Mikrocontroller (Arduino) und dem PC. Das Bluetoothshield nimmt die Daten vom PC entgegen und gibt diese an den Arduino weiter.

## Der Distanzmesser

Der Distanzmesser ist ein Infrarotsensor, welcher ein Hindernis erkennen soll. Durch einen Infrarotstrahl wird die Distanz zu einem Hindernis gemessen. Je nach Abstand zum Hindernis wird durch den Distanzmesser eine LED zum Leuchten gebracht, welche signalisiert, dass das Hindernis nicht mehr weit entfernt ist. Wird der Abstand noch kleiner, dann wird der Roboter zum Stillstand gebracht und dieser kehrt dann automatisch um.

## Der „Roboter“

Alle Teile zusammengesetzt, ausgenommen von der LeapMotion, inklusive anderer benötigter Hardware (Batterien, Kabel, Räder, etc.) ergeben den Roboter.

## Die Software

### C#

#### Erklärung

Auf Basis von C# wurde eine Software entwickelt, welche die Gesten der Hand über dem LeapMotion-Controller einliest, interpretiert und über Bluetooth an den Arduino sendet.

#### Anforderungen an das Programm

Am Anfang soll das Programm eine Bluetooth-Verbindung mit dem Roboter (Arduino) aufbauen, wobei der ausgehende Port des PCs vom Benutzer eingegeben werden kann. Das Programm ist ein Konsolenprogramm.

Die Position der Hand wird mit ihren konkreten Positionswerten laufend eingelesen. Aufgrund dieser Werte soll in Folge ein Code generiert werden, welcher an den Arduino gesendet wird.

Natürlich ist auch Exception-Handling einzubauen, welches absichert, dass das Programm im Falle von nicht vorgesehenen Benutzerinteraktionen nicht abstürzt.

#### Code

Dass eine Kommunikation zwischen zwei oder mehreren Geräten über einen speziellen Code funktionieren muss, ist wahrscheinlich jedem klar.

Von uns wurde für die Kommunikation LeapMotion (bzw. PC) und Arduino dafür ein eigener Code entworfen, der im C#-Programm zusammengestellt und über Bluetooth gesendet wird.

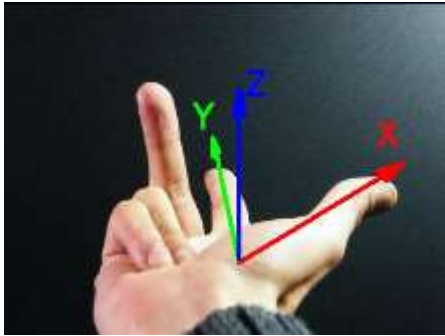
#### Erklärung/Interpretation des Codes

Der im C#-Programm erstellte und im Arduino-Programm interpretierte Code sieht wie folgt aus:

AxxxBxxxC

An erster Stelle, also zwischen A und B, steht die Zahl für die Geschwindigkeit, welche abhängig von der Neigung der Hand ist.

Zur Erklärung des Codes werden folgende Achsen der Hand herangezogen:



Ist die Hand mit den Fingerspitzen maximal nach unten und mit dem Arm maximal nach oben geneigt, d.h. Fingerspitzen in Richtung negativer Z- und Arm in Richtung positiver Z-Achse, so ist der erste Wert zwischen A und B 0. Umgekehrt, wenn die Fingerspitzen maximal in Richtung positiver Z- und der Arm in Richtung negativer Z-Achse geneigt ist, so ist der Wert 200. Befindet sich die Hand genau in Position der Y-Achse, also ist die Hand in Richtung Y-Achse gerade, so ist der Wert

100.

Der zweite Wert zwischen B und C ist analog dazu aufgebaut. Dieser gibt an, ob, in welche Richtung und wie stark gelenkt werden soll. Ist die Hand maximal nach links geneigt, d.h. die X-Achse maximal in Richtung positiver Z-Achse, so ist der Wert 0. Umgekehrt ist der Wert, wenn die Hand maximal nach rechts, d.h. die X-Achse maximal in Richtung der negativen Z-Achse geneigt ist, 200. Ist die Hand gerade, also in Richtung der X-Achse waagrecht, so ist dieser Wert 100.

Die Zahl 100 beschreibt also für beide Fälle das Neutrum.

Um eine gute Steuerung gewährleisten zu können, muss um den Neutralbereich (100) eine Toleranzzone eingebaut werden, da es menschlich kaum möglich ist, die Hand exakt waagrecht zu halten. Ebenso muss ab einer gewissen Neigung alles darüber bzw. darunter als Maximum betrachtet werden, da sonst ein einwandfreies Funktionieren des LeapMotion-Controllers nicht gewährleistet ist. Diese Toleranz- bzw. Maximumsgrenze wurde ausgemessen.

Sinnvolle Werte:

Toleranzbereich Pitch (Vor/Zurück): 10

Toleranzbereich Roll (Lenken): 12

Maximalwert Pitch: 50

Maximalwert Roll: 50

Der Code ist ein Prozentwert, da jeder Teil auf einer Spanne von 100 beruht. Das heißt, in der Berechnung des Codes müssen für Pitch die Spanne von 40 auf 100, für Roll die Spanne von 38 auf 100 ausgeweitet werden.

## Arduino

### Erklärung

### Anforderungen an das Programm

### Code

### Erklärung/Interpretation des Codes

## Testläufe

### Testlauf 1

Beim 1. Testlauf wurde getestet, ob das 1. Prototypprogramm am Arduino läuft. Dieses soll die Motoren links, rechts, vor und zurück fahren lassen. Dazu wurden im Programm Testdaten erzeugt und verwendet. Über den Serial Monitor wurde anschaulich gemacht, welcher Motor bei welchen Daten welches Signal bekommen würde. Sobald diese Routine funktionstüchtig war, wurde der Code für weitere Zwecke verwendet.

### Testlauf 2

Im zweiten Testlauf wurde die Routine des Arduinos bereits mit den Motoren getestet. Dazu wurden Testdaten erzeugt, die schon den fertigen Code, wie er von der LeapMotion kommen soll, simulieren. Anfangs kamen einige Schwierigkeiten zum Vorschein, da die Verarbeitung nicht richtig funktionierte. Nach Fehlersuche und Programmänderungen kamen wir schlussendlich zum gewünschten Ergebnis und die Motoren taten das, was sie laut Routine machen sollten.

Der Roboter konnte nun:

- Sich vor bewegen
- Nach links und rechts fahren

### Testlauf 3

In diesem Test wurde die Bluetooth – Connectivity getestet. Arduinoseitig und auch am PC wurden dazu jeweils ein Programm geschrieben, welche mit den Hauptprogrammen zusammengefügt wurden. Das C# - Programm soll Testdaten erzeugen und über Bluetooth an den Arduino (genauer: zum Bluetoothshield) senden. Dieser nimmt die Daten entgegen und verarbeitet diese so, dass die Motoren das machen, was sie machen sollen.

Das Senden und Empfangen der Daten funktionierte bereits beim ersten Versuch einwandfrei, nur die Verarbeitung am Arduino schlug noch fehl.

Der Fehler konnte jedoch rasch behoben werden. Somit konnten die Motoren des Roboters über den PC angesprochen werden.

### Testlauf 4

Nun wurde versucht, über die LeapMotion den Roboter bzw. dessen Motoren anzusprechen. Ein C# - Programm soll die Daten der LeapMotion übernehmen, diese so verarbeiten, dass sie dem von uns entworfenen Code entsprechen (in der Form AxxxBxxxC) und anschließend an den Arduino übermitteln.

Anfangs ergaben sich noch einige Probleme, da der Roboter überhaupt nicht das machte, was er machen sollte.

Nachdem das Hauptprogramm des Arduinos umgeschrieben wurde, funktionierte die Routine einwandfrei. Der Roboter konnte nun über die LeapMotion ferngesteuert werden, somit war unser SOLL-Kriterium erfüllt.

### **Testlauf 5**

Vor diesem Testlauf wurden diverse Programmverbesserungen vorgenommen. Nun war es möglich, je nach Neigung der Hand schneller bzw. langsamer zu fahren. Je weiter die Hand nach vorne geneigt wird, desto schneller fährt der Roboter nach vorne.

Ebenfalls in die Routine mit eingebaut wurde die „Lenkintensität“. Je weiter die Hand zur Seite geneigt ist (in beide Richtungen), desto enger wird der Kurvenradius gewählt.

Der Test lieferte uns ein positives Ergebnis, somit war schon ein erstes KANN-Kriterium erfüllt worden.

### **Testlauf 6**

Beim 6. und letzten Testlauf wurde der Infrarotdistanzmesser am Roboter angebracht. Fährt der Roboter auf ein Hindernis (Wand, Tür, etc.) zu, so erkennt dies der Sensor und hält den Roboter an. Des Weiteren dreht sich der Roboter selbstständig um, wenn er vor einem Hindernis steht. Aufgrund des erfolgreichen Testlaufes wurde ein weiteres KANN-Kriterium erfüllt.

Am Roboter wurde ebenfalls eine LED angebracht. Diese fängt zu blinken an, sobald sich der Roboter einem Hindernis nähert. Sobald der Roboter kurz vor dem Hindernis ist, bleibt dieser automatisch stehen, wie oben beschrieben, und gleichzeitig leuchtet die LED durchgehend. Dieser Test verlief ebenfalls nach Wunsch und somit wurde das letzte KANN-Kriterium erfüllt.