Tätigkeitsbericht Simon Bodner

LeapMotionRobotTeam

8.10.2014

- Überarbeitung des Pflichtenheftes mit Alexander Hoppichler
- Festlegung der Arbeitsaufteilung (mein Teil: Programmierung der LeapMotion)
- Entwurf des Codes für die Datenübermittlung zwischen LeapMotion (bzw. PC) und Roboter (bzw. Arduino) – Code siehe Pflichtenheft
- Recherche zur Verwendung und Programmierung der LeapMotion

15.10.2014

- Überarbeitung Pflichtenheft (hauptsächlich Beschreibung des verwendeten Codes für die Übermittlung von Daten)
- Einblick in GitHub
- Recherche zur Verwendung und Programmierung der LeapMotion

22.10.2014

- Einlesen in Programmierung der Leap-Motion
- Verfassen von Kurzanleitung zur Benützung von GitHub

29.10.2014

- Beispielprogramm von der LeapMotion-SDK erzeugt, Libraries hinzugefügt
- Programmierung von grundlegenden Funktionen der LeapMotion

03.11.2014

- Erstellung eines Beispielprogramms zum Einlesen von Handposition (ohne Test)

05.11.2014

- Definition von maximalen und minimalen Neigungen der Hand
- Festlegen eines sinnvollen Toleranzbereiches um den Nullpunkt (Handposition gerade)
- Programmierung: Einlesen von benötigten Winkeldaten von LeapMotion, Beginn Programmierung der Methode für die Erzeugung des zu sendenden Codes.

06.11.2014

- Programmierung der Methode für das Erzeugen des zu sendenden Codes. Die von der LeapMotion eingelesenen Werte müssen mit einem Toleranzbereich um den Nullpunkt versehen werden und auf einen Prozentwert (ganzzahlig) in ihrem Gültigkeitsbereich (z.B. Toleranzbereich 10, maximale Neigung der Hand 50 → Gültigkeitsbereich/Prozentbereich zwischen 10 und 50) berechnet werden.
  Noch Fehler in Berechnung.

12.11.2014

- Erzeugen des zu sendenden Codes über C#-Programm funktioniert
- Fehlersuche (Fehler war in Berechnung des Codes. Werte von LeapMotion eingelesen: Basis ist für Vor und Zurück, sowie Links und Rechts 100 (0: minimal bzw. Links, 200: maximal bzw. Rechts))

19.11.2014

- Simulation von Testdaten (da uns LeapMotion dieses Mal nicht zur Verfügung stand)
- Hilfe und Test beim Erstellen des Arduinoprogramms mit der gesamten Gruppe für die Entgegennahme und Interpretation der Werte
- Test von Zusammenspiel von C#- und Arduino-Programm – erfolgreich. Ausgabe der Motoren (noch nicht mit wirklichen Motoren) in TeraTerm.

26.11.2014

- Troubleshooting bzgl. Senden der Daten von LeapMotion zu Arduino
- Fehler bei der Übertragung, allgemeine Fehler

3.12.2014

- Troubleshooting bzgl. Senden der Daten von LeapMotion zu Arduino. Arduino empängt gesendete Daten nicht korrekt. Anscheinend liegt dies am Delay im Arduino- bzw. Thread.Sleep im C#-Programm.
  Meilensteine für nächste Einheit: Korrektes Aufspalten der Daten am Arduino

10.12.2014

- Erfolg beim Aufspalten des gesendeten Codes am Arduino:
  Code wird als String gesendet und über die C-Methode strtok aufgespalten. Anschließend werden die beiden Array-Einträge mithilfe von atof von String in int umgewandelt. Nun liegen am Arduino die beiden notwendigen Werte wie gewünscht vor. Es folgt die Interpretation.

Bsp.:

```
Input:    A100.0B100.0

Motor vor
```

A100.0 bedeutet nach vorne fahren (100 ist Grenze – 99 wäre bereits rückwärts
B100.0 bedeutet geradeaus fahren (keine Lenkung)

```
Input:    A177.0B18.0


Motor vor
Motor left
```

A177.0 → nach vorne fahren
B18.0 → nach links lenken (linker Motor aus, rechter Motor ein)

```
Input:    A57.0B144.0


Motor zuruck
Motor rechts
```

A57.0 → rückwärts fahren
B144.0 → nach rechts lenken

```
Input:    A42.0B107.0


Motor zuruck
Motor rechts
```

A42.0 → rückwärts fahren
B107.0 → nach rechts lenken

- Löten von manchen Stellen des Roboters

17.12.2014

- Erstellen eines Testlaufs über den C#-Code (vor, lenken, zurück, lenken).

Nächster Meilenstein: Fertigstellung der Hardware (kalte Lötstellen, brüchige Verbindungen etc), Testläufe

7.1.2015

- Löten einer Bruchstelle
- Test mit LeapMotion, Problem: Daten kommen am Arduino richtig an, Roboter interpretiert sie aber falsch (statt zurück z.B. stopp)

Nächster Meilenstein: Roboter soll korrekt nach vorne und hinten fahren

14.1.2015

- Ansteuerung des Roboters – es funktioniert: nach vorne, hinten Fahren sowie Lenken. Sollkriterium ist erreicht (fahren mit konstanter Geschwindigkeit).

Weitere Schritte: Geschwindigkeit des Roboters soll von Neigung der Hand vorgegeben werden können. – bis 28.1.2015

21.01.2015

Fehlersuche bzgl. heute nicht funktionierender Bluetooth-Verbindung – am Ende funktionierte wieder alles.

28.1.2015

Es funktioniert mittlerweile, den Roboter mit verschiedenen Geschwindigkeiten zu steuern – je nach Lage der Hand. Auch Lenken funktioniert abhängig von der Handschräge unterschiedlich schnell. Damit ist das erste Kannkriterium bereits erreicht.

Weitere Vorgehensweise:
Umsetzung der zwei weiteren Kannkriterien (Abstandsmessung über Infrarotsensor und Stehenbleiben bevor Auffahren auf ein Hindernis bzw. Leuchten einer Warn-Led bei geringem Abstand zu einem Hindernis.

Nächster Meilenstein nächste Woche:
Aufrüsten des Roboters um einen Infrarotsensor

4.2.2015
Roboter aufrüsten um Infrarotsensor und Led (Blinken wenn Abstand zu Hindernis zu klein)
In C#-Programm werden dem Benutzer nun alle verfügbaren COM-Ports aufgelistet, er kann einen davon auswählen.

Meilenstein nächste Woche: Test, ob Funktionalität mit Infrarotsensor (Stop und Drehung bei zu wenig Abstand) funktioniert

25.02.2015
Verfassen der Dokumentation über das C#-Programm und den erstellten Code.
Nächste Woche: Weiterführen der Dokumentation