Projektbericht – „Spidercam“

Zusammenfassung unseres Projekts:

Nach einem sehr frühen Ausscheiden der deutschen Nationalmannschaft in der Gruppenphase der Weltmeisterschaft 2018, mussten wir uns ein neues Interessengebiet im Rahmen der Weltmeisterschaft suchen. Schon während der (wenn auch nur wenigen) Spiele der deutschen Nationalmannschaft fielen uns viele schöne Kameraperspektiven auf, die das Spielfeld aus allen Blickwinkeln zeigten und so wirkten, als ob die Kamera über das Spielfeld fliegen würde.

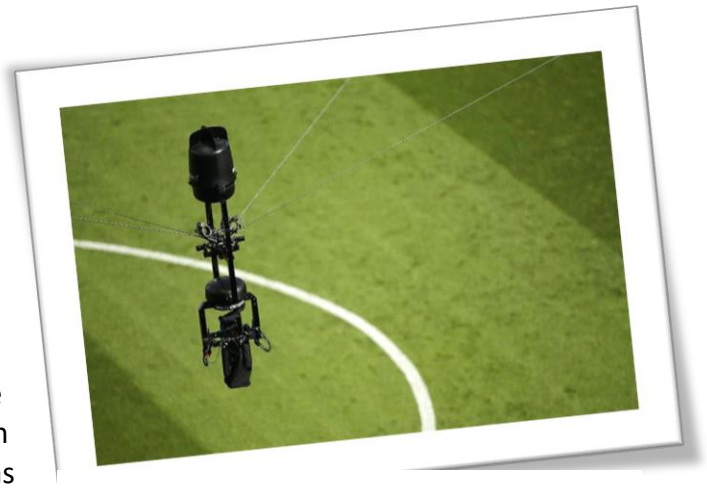

Abb. 1: Spidercam beim Fußball

So fanden wir schnell Interesse an der sog. „Spidercam“ und wollten mehr darüber erfahren. Deshalb entschieden wir uns, eine Spidercam mal im Kleinen nachzubauen und zu programmieren.

Einige dazu nötige Materialien hatten wir schon bereitgestellt bekommen. Die Rollen, über die das Seil laufen sollten, mussten wir allerdings noch selbst modellieren, da wir für diese eine bestmögliche Größe haben wollten. Außerdem mussten wir auch unsere handwerklichen Fähigkeiten einsetzen, um die Hardware aufzubauen, alles richtig zu verkabeln und die Schnüre ordentlich anzubringen. Eine weitere große Hürde war die korrekte und synchrone Steuerung der Motoren zu programmieren und das Bild der Kamera auf einen Bildschirm zu übertragen.

Den genauen Ablauf unserer Woche wollen wir hier darstellen.

Dienstag, 10.07.2018:

Nachdem wir uns in unserer Gruppe zusammengefunden hatten, starteten wir erst einmal damit, die oben schon aufgezählten Problemstellungen unseres Projekts und die Aufgabenschritte zu erarbeiten.

Dann teilten wir uns in kleinere Gruppen auf und modellierten die Spulen mit Hilfe des Programms „FreeCAD“, um sie dann im 3D-Drucker herstellen zu können. Dabei konstruierte jede Gruppe Spulen einer anderen Größe, damit wir die Optimalste verwenden konnten.

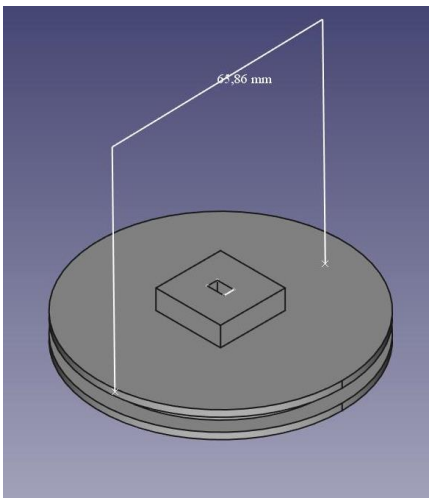

Abb. 2: Konstruktion der Spule mittels FreeCAD

Unser nächster Schritt war die Hardware. An einer Platte brachten wir den Raspberry Pi B+ (ein Minicomputer), die vier Motoren und vier Platinen (zur Steuerung) an und

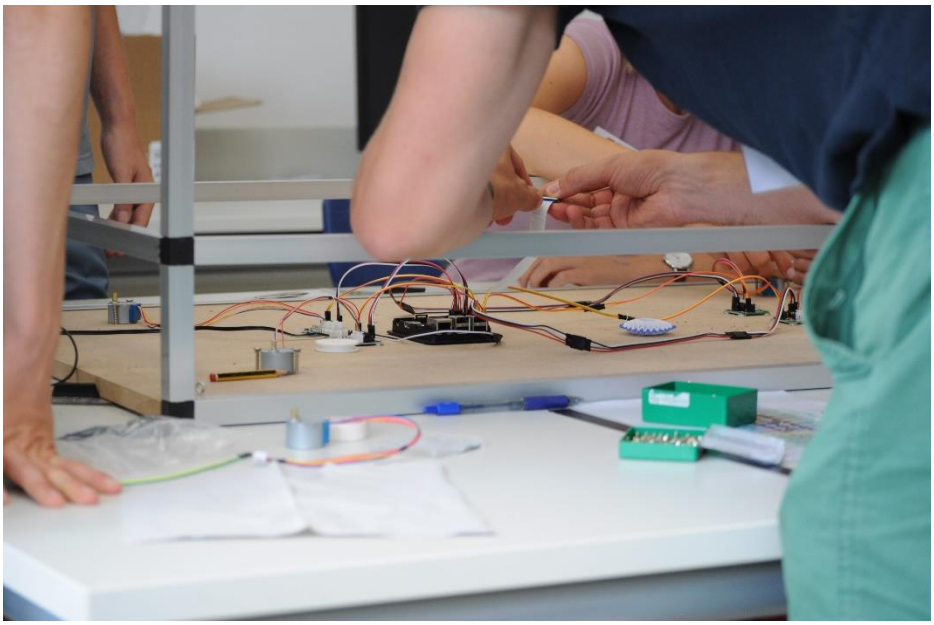

Abb. 3: Hardwareaufbau

verkabelten das Ganze. Als uns dies erfolgreich gelungen war, erweiterten wir unseren Aufbau zur Probe mit Testspulen, einem Kameramodul und Fäden.

Für einen guten Abschluss unseres ersten Tages hatte natürlich noch etwas Mathematik gefehlt. Also begannen wir uns zu überlegen, wie eine beliebige Position der Kamera mithilfe von Vektoren bestimmt werden kann und berechneten diese Vektoren mittels beispielhafter einfacher Werte. Um zu berechnen, wie mittels Vektoren ermittelt werden kann, wieviel Seil auf- bzw. abgewickelt werden muss, muss man die Länge der Vektoren subtrahieren:

$$W_1 = \begin{pmatrix}90\\0\\30\end{pmatrix},\ K_0 = \begin{pmatrix}45\\60\\25\end{pmatrix},\ K_1 = \begin{pmatrix}45\\60\\6\end{pmatrix}$$

$$\overrightarrow{W_1K_0} = \begin{pmatrix}45\\60\\25\end{pmatrix} - \begin{pmatrix}90\\0\\30\end{pmatrix} = \begin{pmatrix}-45\\60\\-5\end{pmatrix}$$

$$\left|\overrightarrow{W_1K_0}\right| = \sqrt{(-45)^2 + 60^2 + (-5)^2} = \sqrt{5650} = 75{,}17cm$$

(analog für $\left|\overrightarrow{W_1K_1}\right| = 78{,}75$cm)

Aus der Differenz von $\left|\overrightarrow{W_1K_1}\right|$ und $\left|\overrightarrow{W_1K_0}\right|$ ergibt sich, dass das Seil an dieser Stelle um 3,58 cm aufgewickelt werden muss, wobei W_1 einem der Pfosten und K_i den beiden Kamerapositionen entspricht.

Mittwoch, 11.07.2018:

In den nächsten Tag starteten wir mit einer GeoGebra-Simulation unserer Spidercam. In Gruppen konstruierten wir eine 3D-Graphik zur Darstellung dieser, um berechnen zu können, wie viel Seil bei einer Positionsänderung aufgerollt oder abgewickelt werden muss, sodass wir überprüfen konnten, ob unsere Programmierung korrekt ist. Außerdem half uns die Simulation, um uns unser Ziel besser vor Augen führen zu können.

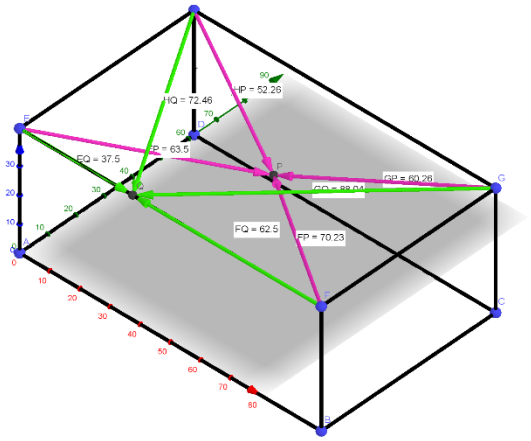

Abb. 4: GeoGebra-Simulation einer Spidercam

In unserer Darstellung verwendeten wir dazu zwei Punkte. Einer, an dem die Kamera sich aktuell befindet (dies ist der Punkt bei den lilafarbenen Vektoren) und einer, an den die Spidercam hinbewegen soll (dies ist der Punkt bei den grünen Vektoren).

Die Vektoren gehen jeweils von den Eckpunkten zur Kamera und symbolisieren die Seile. Eine Längeneinheit in unserer 3D-Graphik entsprach hierbei einem Zentimeter in der Realität. Die Differenzen der Seillängen, die wir benötigten, bekamen wir durch Subtraktion des Vektors der aktuellen Position mit dem jeweils zugehörigen der gewünschten heraus.

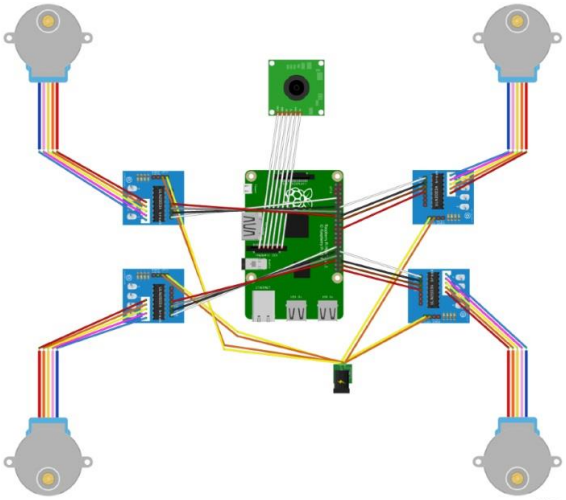

Abb. 5: Schaltplan unserer Spidercam

Mithilfe des Programms „Fritzing“ erstellten wir den Schaltplan der Hardware zur anschaulichen Darstellung unseres Projekts. Hierbei mussten wir die einzelnen Bauteile zunächst im Internet heraussuchen und anschließend importieren und modellieren.

Der Kern unseres Projekts ist der Raspberry Pi, der sich in der Mitte befindet und über Kabel mit den 4 Steuerungsplatinen (blau) verbunden ist. Diese werden mit Gleichspannung versorgt und sind jeweils mit einem der Motor verbunden, die sich in den Ecken befinden. Die Spidercam (grün) ist am Minicomputer angeschlossen und befindet sich in der Graphik über diesem.

Anschließend nahmen wir Messungen vor, um herauszufinden, welche der Spulen die Optimalste war. Zuerst rechneten wir mit dem Radius den Umfang der jeweiligen Spulen aus:

$$\textbf{Beispiel } 1\text{: Spule mit } r = 3{,}25 \text{ cm: } U = 2 * \pi * 3{,}25 \text{ cm} = 20{,}42 \text{ cm}$$

$$\textbf{Beispiel } 2\text{: Spule mit } r = 1{,}50 \text{ cm: } U = 2 * \pi * 1{,}50 \text{ cm} = 9{,}43 \text{ cm}$$

Der Umfang der Spulen entsprach der Länge der Fäden, die mit einer Umdrehung des Motors auf- bzw. abgewickelt werden konnte. Da bei gleicher Geschwindigkeit die größere Spule schneller mehr Seil abwickeln konnte, entschieden wir uns, diese für unser Modell zu

benutzen. Die Umfänge der Spulen benötigten wir auch zur Berechnung der Umdrehungen, um eine beliebige Länge des Seils auf bzw. abzuwickeln.

Nach einer kurzen Einweisung in die Funktionsweise eines Schrittmotors, programmierten wir in der Programmiersprache „Python“ die Drehung des Motors in beide Richtungen. Dabei arbeiteten wir mit einer Vorlage unserer Betreuer.

Anschließend lasen wir uns in die Syntax von Python ein und erstellten gemeinsam ein funktionales Modell für unser Programm. Wir stellten fest, dass wir drei Funktionen benötigten, um das Projekt umzusetzen (s. Abb. 4):

Die erste rechnet die benötigte Seillänge aus der aktuellen und der gewünschten Position mittels Vektoren aus (s. oben).

Aus diesem Wert wird die Anzahl der Umdrehungen der einzelnen Schrittmotoren berechnet.

Auch benötigten wir noch eine Funktion, die die entsprechenden Motoren ansteuert.

Im Anschluss versuchten dann nach und nach einige Gruppen, die Funktionen zu implementieren. Trotz einiger Misserfolge standen am Ende des Tages zumindest schon die ersten beiden Funktionen komplett.

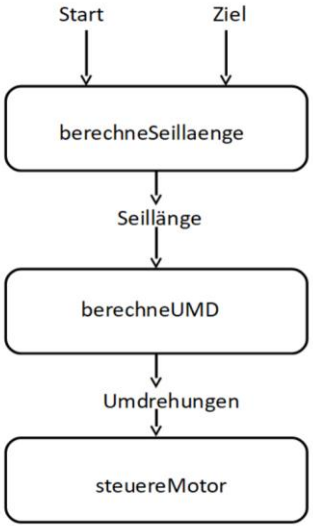

Abb. 6: Funktionen unseres Programms

Währenddessen beschäftigte sich ein Schüler mit der Ausgabe des Kamerabildes auf einem mittels HDMI-Kabel angeschlossenen Monitors.

Donnerstag, 12.07.2018:

Am Donnerstag setzten alle Kleingruppen die Arbeit an ihren Programmen fort. Nach einigen kleinen Änderungen stellten wir das Programm fertig. Anschließend testeten wir unser Ergebnis am Modell, wobei wir eine Enttäuschung hinnehmen mussten:

Es konnten zwar alle Motoren angesteuert werden, jedoch nur nacheinander, und nicht - wie gewünscht - gleichzeitig. Ein Teil von uns befasste sich dann weiter mit dem Modell, sei es die Steuerung der Motoren zu kalibrieren oder die Optimierung der Kabelführung und des Hardwareaufbaus. Dabei ergaben sich ein paar unvorhergesehene Probleme, die sich im Laufe des Vormittags aber lösen ließen. Der andere Teil übernahm das Problem der gleichzeitigen Ansteuerung der Motoren. Als geeignete Lösung stellten sich „Threads“ heraus. Dabei können mehrere Programmabschnitte gleichzeitig ausgeführt werden und so konnten wir unsere Motoren wie gewünscht steuern.

Im Weiteren ist ein Ausschnitt des Programmcodes in Python, mit dem der Motor in Abhängigkeit von den zuvor berechneten Umdrehungen gesteuert werden kann.

```
def steuereMotor(umdrehungen):

    # Imports Math, Threading
    import math
    import threading

    # Imports Schrittmotoren
```

```
import schrittmotor1 as sm1
import schrittmotor2 as sm2
import schrittmotor3 as sm3
import schrittmotor4 as sm4

# Setup der Schrittmotoren
sm1.setup()
sm2.setup()
sm3.setup()
sm4.setup()

# Steuerung der Schrittmotoren
if umdrehungen[0]>0:
    th1=threading.Thread(target=sm2.forward, args = (int(abs(umdrehungen[0])),))
    th1.start()
else:
    th1=threading.Thread(target=sm2.backward, args=(int(abs(umdrehungen[0])),))
    th1.start()
```

Nach einem Testlauf, bei dem alles was wir erreichen wollten, gut funktionierte, konnten wir entspannt zurück zu unserem Seminarzentrum laufen und abends an der Präsentation und dem Projektbericht arbeiten.

Freitag, 13.07.2018:

An unserem letzten Tag bastelten wir dann das Fußballfeld, optimierten unser Projekt nochmals und stellten den Projektbericht und die Präsentation für den Nachmittag fertig.

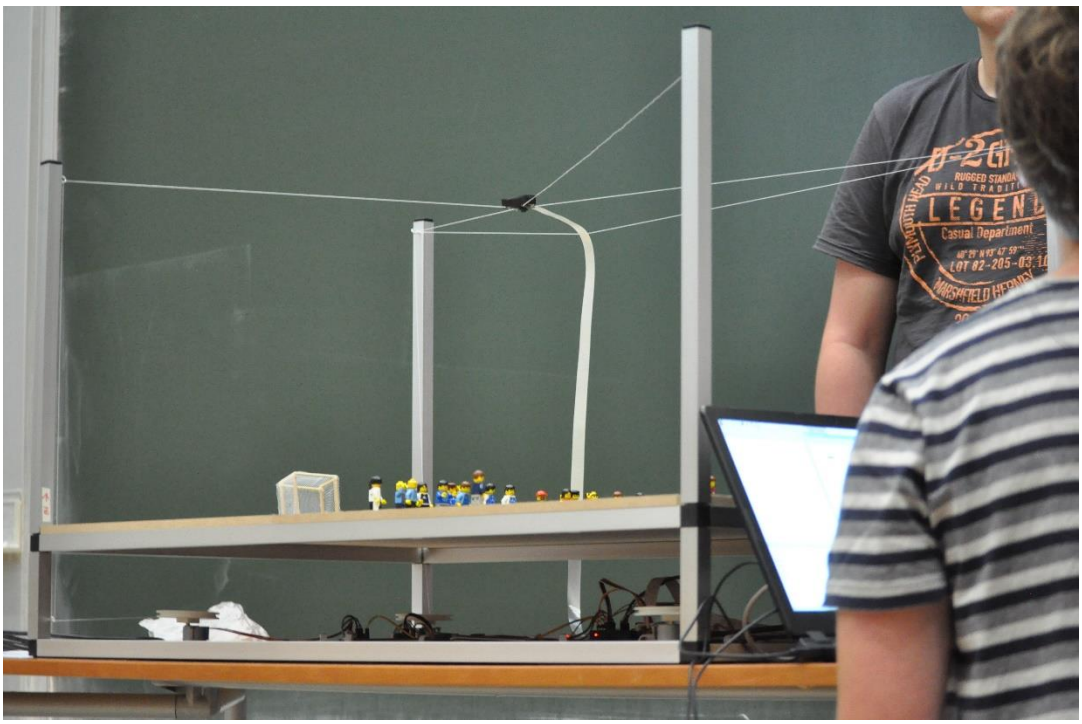

Abb. 7: Spidercam Modell