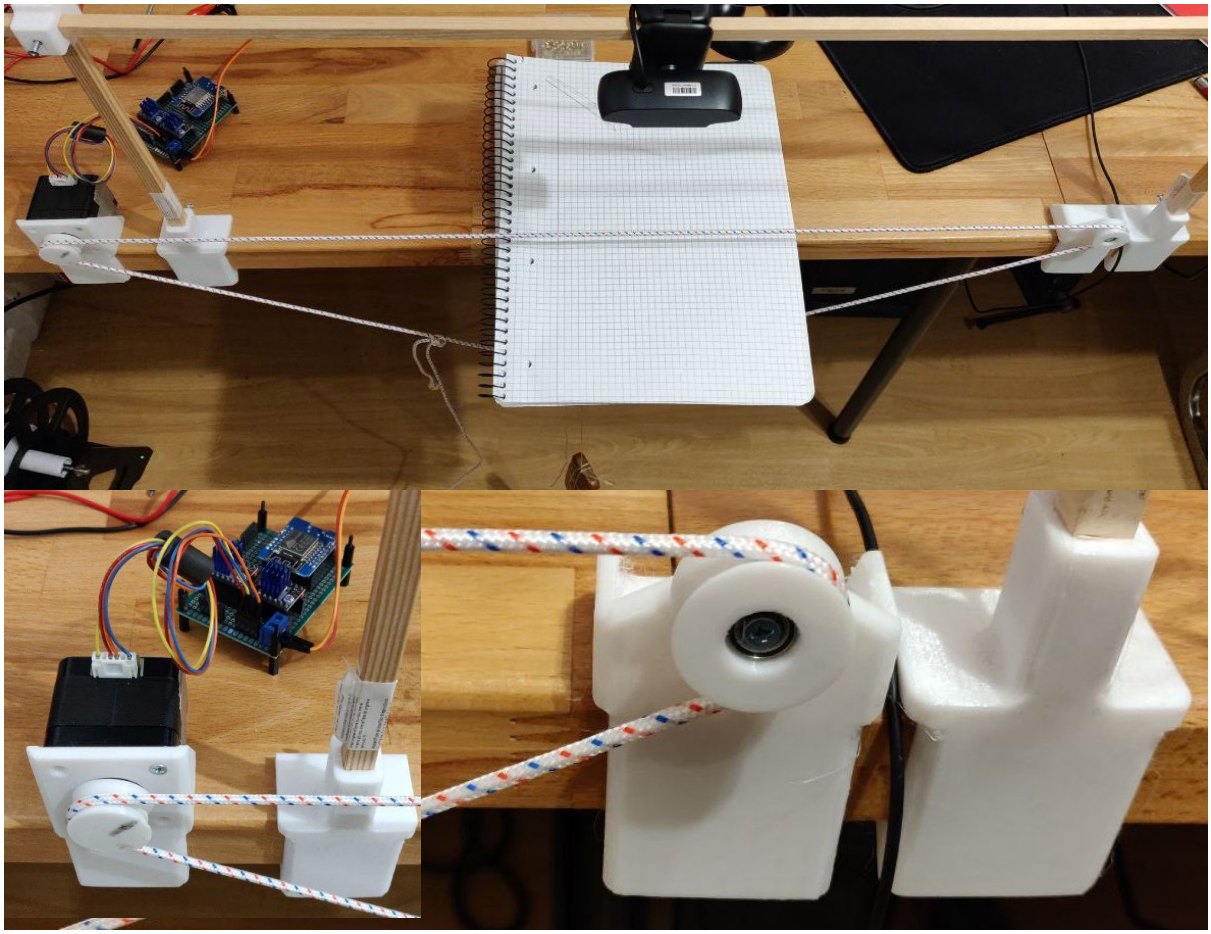
Meilenstein 2 Projekt AWP 3D Rechnersehen/ Projekt Intelligente Systeme: Umsetzung Teil 1

Die aktuelle Phase des Projektes dreht sich um eine erste prototypische Umsetzung des linearen Encoders (CamLinEnc). Dazu wurden die Aufnahmen aus dem Testaufbau genutzt, welcher in Abbildung 1 dargestellt ist.



FAbbildung 1: Gesamtansicht mit Kamera, welche auf das Seil vor einheitlichem Hintergrund gerichtet ist (oben). Umlenkrolle mit Motor und Steuerung (links). Halterung für Umlenkrolle und Kameragestell (rechts).

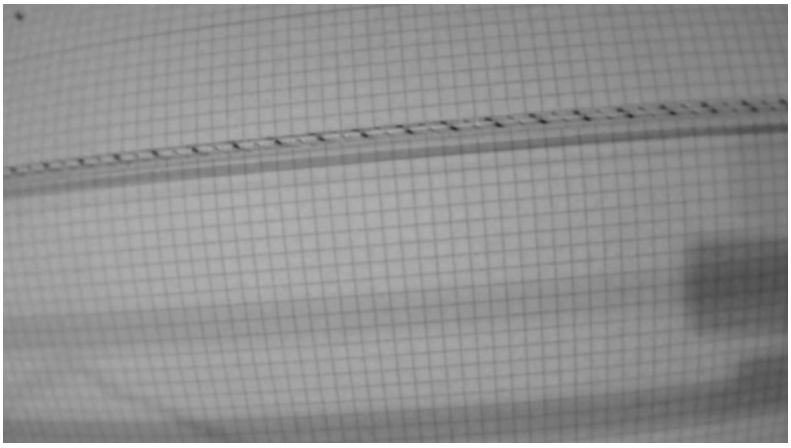


Abbildung 2: Ein Frame der Testaufnahme

Die mit einer 30 FPS, 80° Weitwinkel Webcam (RAPOO XW170[1]) aufgenommenen Videos des Seils, sind mit 720p aufgelöst. Was für eine verhältnismäßig niedrige Qualität des Videos sorgt. Die Aufnahmen sind des Weiteren verschwommen, verzeichnet und es sind Schatten auf dem Bild zu erkennen, wie in Abbildung 2 dargestellt. Da der CamLinEnc möglichst robust funktionieren soll, werden diese Aufnahmen für die Entwicklung der Software-Library verwendet.

In einem ersten Verarbeitungsschritt werden mithilfe des Canny-Algorithmus die Kanten des Seils bestimmt. Dafür wird hier die openCV Funktion cv2.Canny[2] verwendet.

Abbildung 3: Nach der Anwendung des Canny-Algorithmus

Über die in Abbildung 3 deutlich sichtbaren Kanten lassen sich Graden legen, um die Lage des Seils zu bestimmen. Dafür wird die openCV Funktion cv2.HoughLines[3] verwendet.
Alternativ könnte man für die Gradenbestimmung auch das Differenzbild aufeinanderfolgender Video-Frames verwenden, wie in Abbildung 4 zu sehen.
Zu beachten ist dabei aber, dass das Seil im Ruhezustand kein signifikantes Differenzbild erzeugt.

[1] https://www.rapoo-eu.com/de/product/xw170/#specifications-anchor
[2] https://docs.opencv.org/3.4/da/d22/tutorial_py_canny.html
[3] https://docs.opencv.org/3.4/d9/db0/tutorial_hough_lines.html

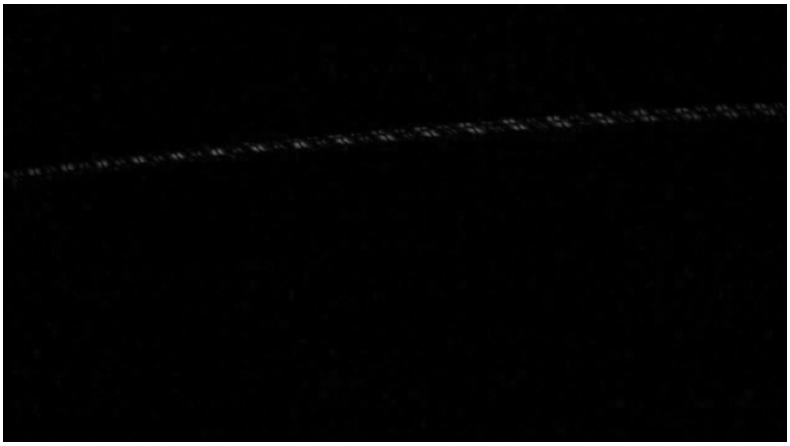


Abbildung 4: Das Differenzbild aufeinanderfolgender Frames

Um die Markierungen des Seils zu lokalisieren, wie in Abbildung 5 zu erkennen, wird die Blob-Detection Funktion cv2.SimpleBlobDetector[4] verwendet.

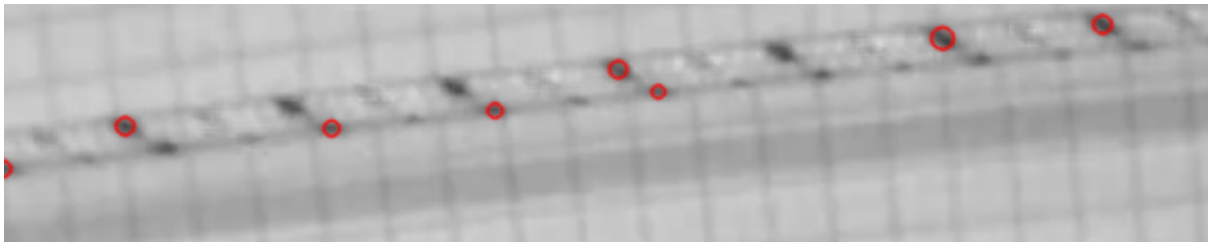


Abbildung 5: Die roten Kreise zeigen die erkannten Markierungen Auf dem Seil.

Im nächsten Entwicklungsschritt wird mithilfe der bekannten Lage des Seils und einer perspektivischen Transformation ein Ausschnitt des Bildes definieren, welcher nur das Seil enthält. Mit der sogenannten Region-of-interest und den erkannten Markierungen soll dann im darauffolgenden Entwicklungsschritt die Bewegung des Seils encodiert werden.

[4] https://learnopencv.com/blob-detection-using-opencv-python-c/