**Systems Engineering - Projekt**

**Dokumentation**

***„ Telepresence with Virtual Reality Headset “***

**Dozent: Dipl. Ing. Peter Puschmann**

**Gruppe: Tim, Nieter (s0533317@htw-berlin.de)**

**Christian, Tollas (s0533318@htw-berlin.de)**

Inhalte

[1. Rasberry Pi Compute Module 3](#_Toc443419629)

[1.1 Videopipline Script 3](#_Toc443419630)

[1.2 RasberryServoControl 4](#_Toc443419631)

[2. Windows Host Notebook 6](#_Toc443419632)

[2.1 OculusTelemetrySender 6](#_Toc443419633)

[2.2 CameraToRiftRecv 7](#_Toc443419634)

# 1. Rasberry Pi Compute Module

Auf dem Raspberry gibt es zwei Komponenten.

* Die Videopipeline die das Bild der Kameras mittels raspivid und Gstreamer überträgt.
* Das RaspberryServoControl, welches über eine Netzwerkverbindung angesteuert wird und die Telemetriedaten der Oculus Rift für die drei Servos der 3-Achsen-Konstruktion übersetzt.

## Videopipline Script

Das Script für die Videoübertragung hat folgenden Inhalt.

*#!/bin/bash*

*raspivid -t 0 -w 1280 -h 720 -fps 48 -b $3 -o - | gst-launch-1.0 -e -v fdsrc ! h264parse ! rtph264pay pt=96 config-interval=1 ! udpsink host=$1 port=$2*

**Usage:**

1\_Thread\_1Camera\_Gstreamer.sh **[IP-Adresse] [Port] [Bitrate]**

* **[IP-Adresse]:** IP-Adresse des Zielsystems.
* **[Port]:** Port
* **[Bitrate]:** Zielbitrate für den Encoder und die Übertragung. Muss angepasst werden falls die Übertragungsrate im WLAN aufgrund von Störungen nicht erreicht werden kann.

**Beispiel:**

*“1\_Thread\_1Camera\_Gstreamer.sh 192.168.1.88 5000 4000000”*

**Aufbau:**

*raspivid:*

Raspberry Programm welches die Kameradaten über den CSI Bus ausliest.

* + **–t 0:** Gibt die Übertragungszeit an. 0 = Unendlich
  + **–w 1280 –h 720:** Gibt die Auflösung an. 1280x720 Pixel (Breite\*Höhe)
  + **–fps 48:** Gibt die Framerate an. 48 Frames pro Sekunde.
  + **–b $3:** Gibt die Bitrate für den Encoder und die Übertragung an. Setzt die Bitrate auf den Wert der mit Parameter $3 übergeben wurde.
  + **–o -:** Output des Streams an Gstreamer

*Gst-launch-1.0:*

Programm welches die Kameradaten aufnimmt und per UDP Stream an das Zielsystem sendet.

* **–e –v:** Verbose. Gibt Debuginformationen aus.
* **fdsrc:** Liest von einem unix File-Descriptor. Hier: Nimmt den von raspivid übergebenen Video-Stream als Quelle.
* **h264parse:** Parser für h264 Stream
* **rtph264pay:** Teilt die h264 Frames in RTP Pakete
* **udpsink host=$1 port=$2:** Gibt als Senke ein Netzwerkhost an. Sendet die Daten dann als UDP Pakete

## RasberryServoControl

Das RaspberryServoControl hört auf dem **Port 5555** auf eingehende Verbindungen.

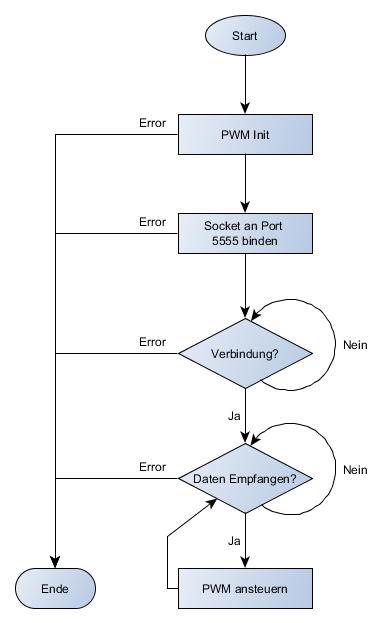
Bei erfolgreich etablierter Verbindung, werden die Daten der Oculus Rift in folgendem Format erwartet.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Byte 0 | Byte 1 | Byte 2 | Byte 3 | Byte 4 | Byte 5 | Byte 6 | Byte 7 |
| Länge Frame | CMD | Roll, LSB | Roll, MSB | Pitch, LSB | Pitch, MSB | Yaw, LSB | Yaw, MSB |

Wobei **CMD = 0x01h** sein muss.

Nach erfolgreichem Empfangen eines Frames werden die Daten von Roll, Pitch und Yaw auf die jeweiligen Servos übertragen. Der Wertebereich geht dabei von 0 bis 180, da die Servos nur für einen 180° Bereich ausgelegt sind.

Zur Feinjustierung der **Servos** kann in der **GlobalConst.h** Datei ein Offset für die Servos angegeben werden.



# 2. Windows Host Notebook

Auf dem Windows System gibt es zwei Softwarekomponenten die über eine GUI miteinander verbunden sind.

* Der OculusTelemetrySender liest die Sensordaten der Oculus Rift mittels OculusSDK aus und überträgt sie an das Raspberry Pi.
* CameraToRiftRecv ist das Empfangsprogramm für den Video-Stream des Raspberry Pi. Es empfängt die Daten und bereitet sie in einer 3D Umgebung auf um mit der Oculus betrachtet werden zu können.

## 2.1 OculusTelemetrySender

**Usage:**

TeleVRTelemetrySender.exe **[IP-Adresse] [Port]**

* **[IP-Adresse]:** IP-Adresse des Zielsystems.
* **[Port]:** Port auf dem Zielsystem

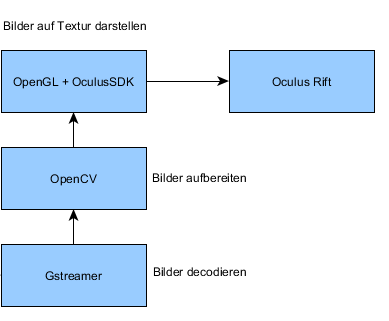
**Beispiel:**

*„TeleVRTelemetrySender.exe 192.168.1.107 5000“*

Der OculusTelemetrySender verbindet sich mittels UDP Socket mit dem angegebenen Zielsystem und überträgt kontinuierlich alle 25 ms die Sensordaten der Oculus in folgendem Format.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Byte 0 | Byte 1 | Byte 2 | Byte 3 | Byte 4 | Byte 5 | Byte 6 | Byte 7 |
| Länge Frame | CMD | Roll, LSB | Roll, MSB | Pitch, LSB | Pitch, MSB | Yaw, LSB | Yaw, MSB |

## 2.2 CameraToRiftRecv



Die OpenGL Anwendung basiert auf den Beispielen in <https://github.com/OculusRiftInAction/OculusRiftInAction>.

Die Anwendung dient als Empfänger für den Video-Stream der von dem Raspberry Pi gesendet wird. Zum Empfangen kommt auch hier Gstreamer zum Einsatz, welches die Bilder decodiert.

Anschließen werden die Bilder so aufbereitet das sie von OpenCV verwendet werden können. Von OpenCV werden die Bilder dann an OpenGL übergeben wo sie als Textur auf ein vor dem Nutzer schwebenden Fenster dargestellt werden.