Hochschule Konstanz
Fakultät Elektrotechnik
und Informationstechnik

Baumer

Embedded Systems – Radar Scanner Reference Box

Team BRT

Professor: Prof. Dr. Boris Böck
Firmenbegleiter: Daniel Ritzmann, (Julian Achatzi)

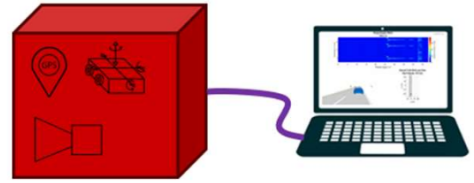
Agenda / Inhaltsverzeichnis

- Einleitung
  - Aufgabenstellung / Anforderungen
- Aufbau Hardware
- Aufbau Software
  - Firmware Radar Scanner Reference Box
  - PC-Visualisierung
  - Kamera Datenerfassung
- Retrospektive
  - Offene Punkte
  - Herausforderungen im Team

Einleitung

Aufgabenstellung

Radar Scanner Reference Box – Spezifikation



Kamera

Muss:

- Auflösung: ≥1MP
- Framerate: ≥10Hz
- Öffnungswinkel: ≥ 120°

PC Software

Muss:

- Empfangen und Speichern des Videostreams und der Lage
- Nachträgliche Visualisierung der Kamerabilder und der Lage

Kann:

- Echtzeit Visualisierung der Kamerabilder und der Lage

Lage

Muss:

- Output der Position in kartesischem Raum (x,y,z Koordinate)
- Output der Ausrichtung (Rotation) in Roll-Nick-Gier Winkel
- Optimierung der Positionsgenauigkeit mittels Fusion der GPS und der IMU Daten

System

Muss:

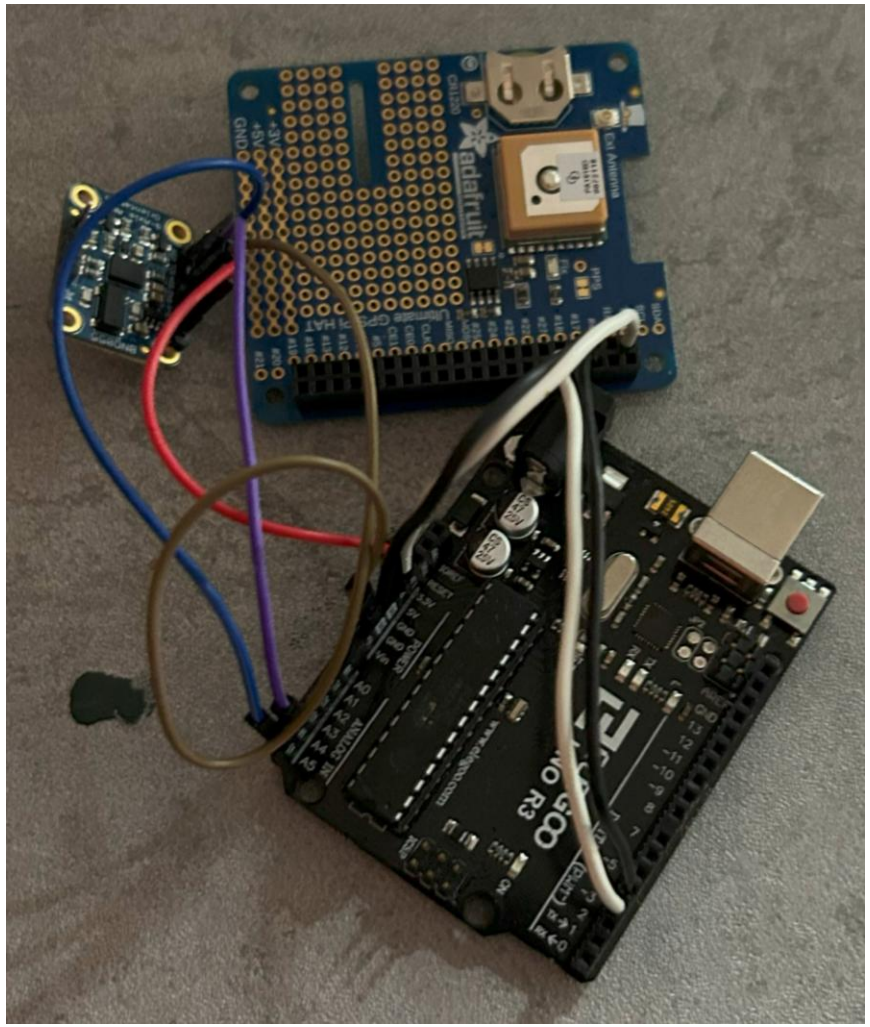
- Video sowie Lagedaten müssen mit einem relativen Zeitstempel versehen sein (Keine absolute Zeit nötig)
- Versorgungsspannung und Schnittstelle über USB (oder Ethernet)
- Robustheit, sodass die Box an einen Traktor montiert werden kann (nicht für Dauereinsatz, und bei trockenem Wetter)

Hardware

GPS & IMU

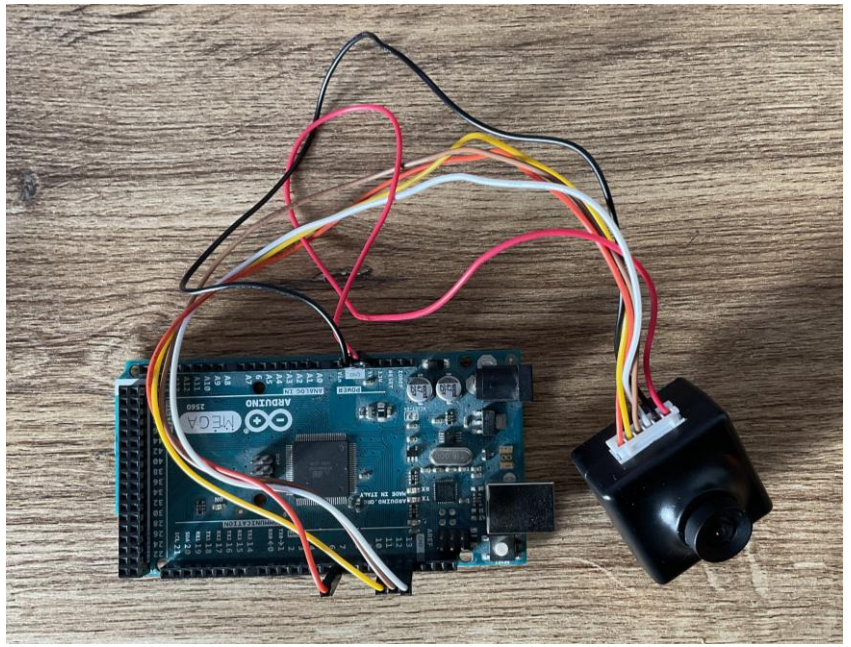
- Rapid Prototyping
  - Einbindung Sensoren in Arduino
  - Auswertung der Messdaten

- Verifizieren der Messdaten in Python

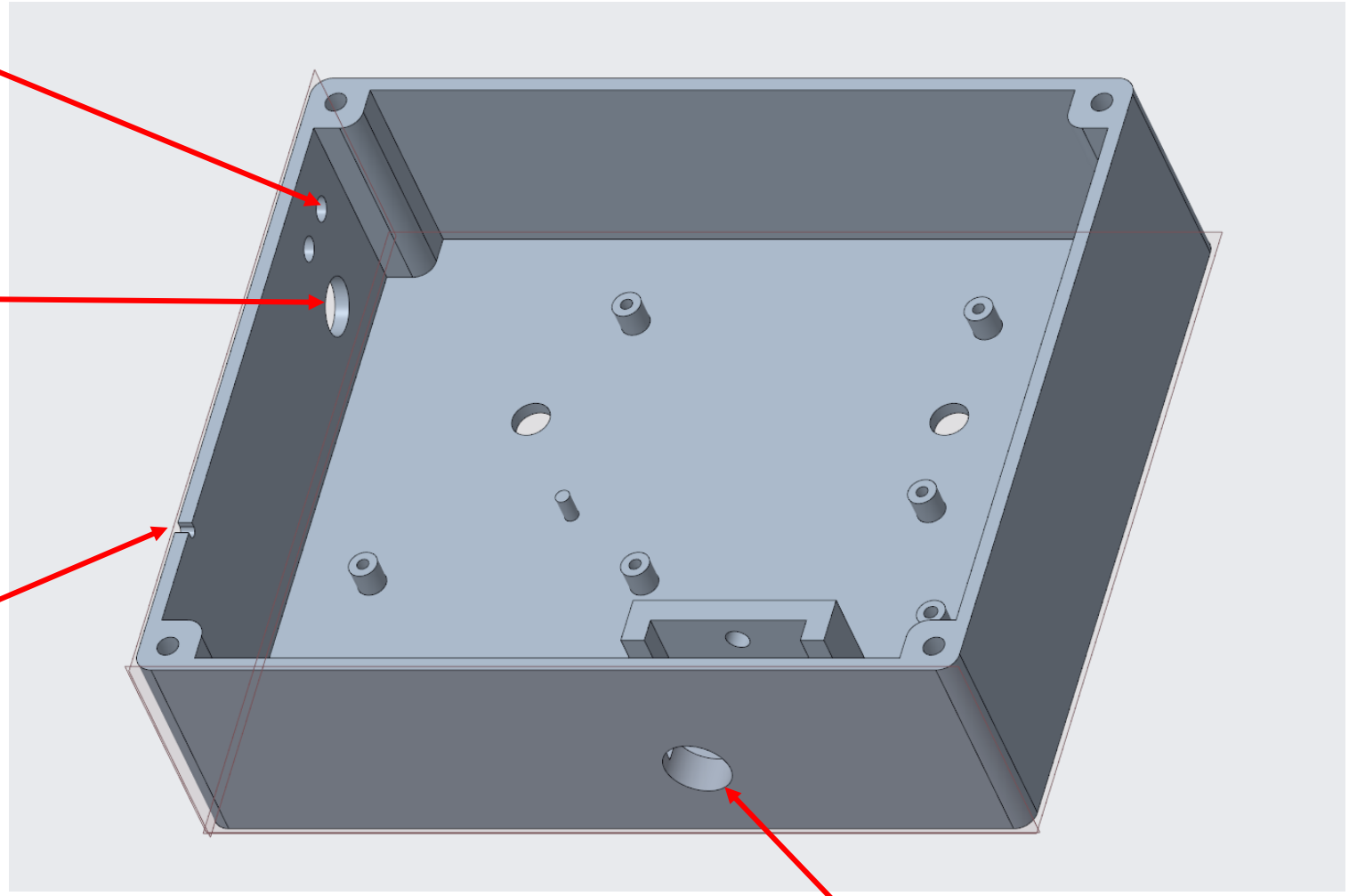


Kamera

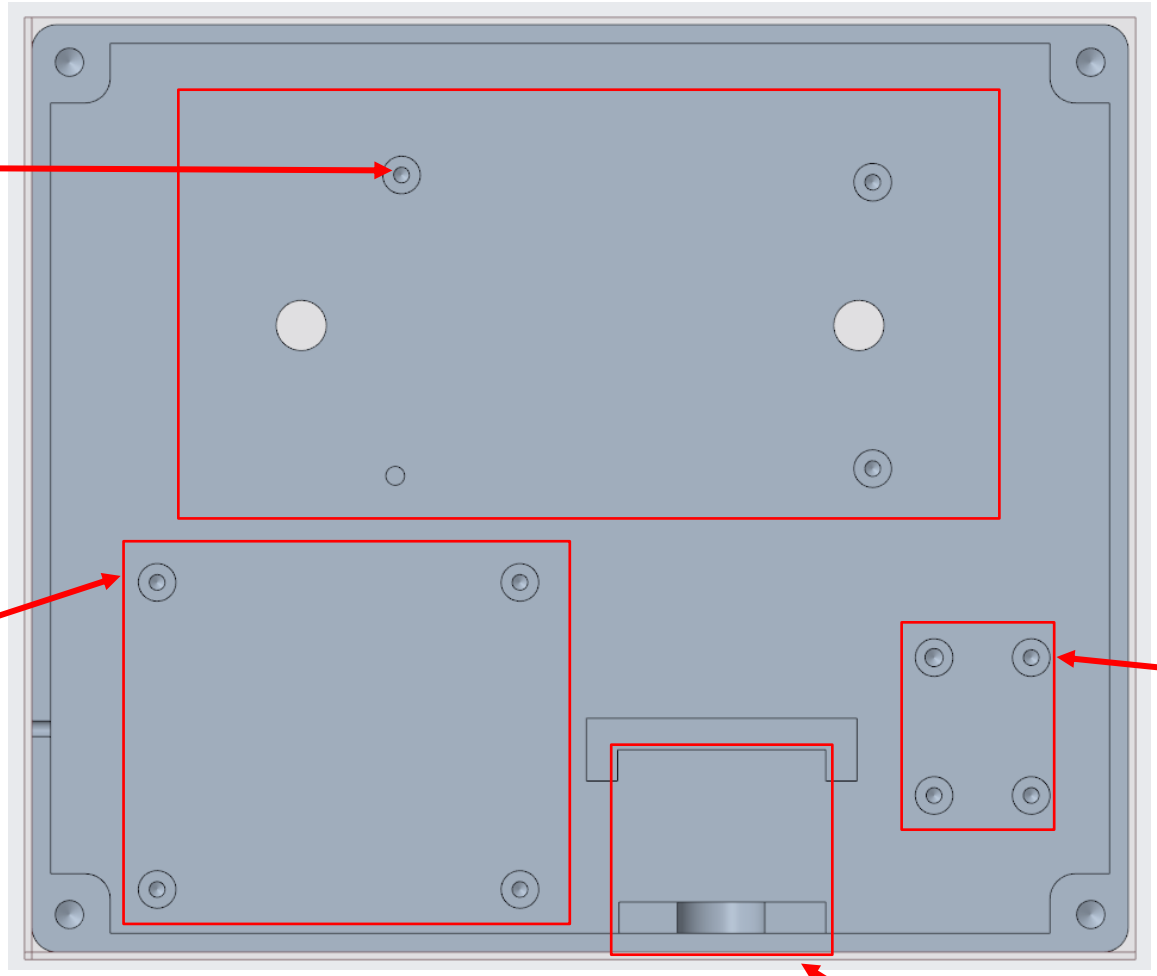
- **Rapid Prototyping**
  - OV5640
    - I2C Kamera mit DCMI
    - Keine geeignete Bibliothek für den STM32
  - SPI Kamera
    - Arducam Mega
    - Datenverarbeitung in der Kamera
    - Jpeg Daten an Kontroller über SPI
    - Erste Test mit Arduino
    - Auslesung der Kamera Bilds über Software



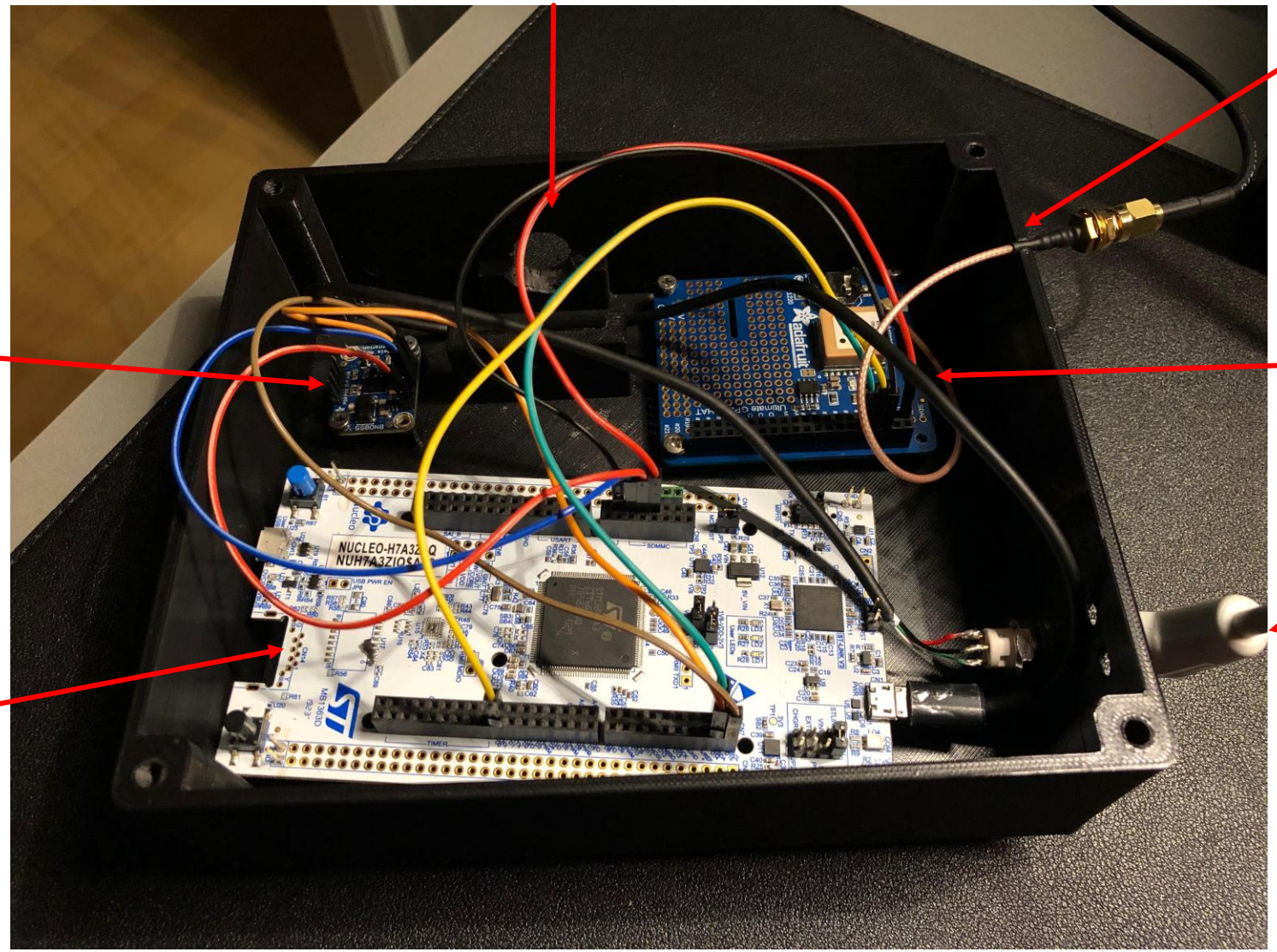
Gehäuse


Status LED
Kabeldurchführung
USB mit
Zugentlastung
M12 T-kodiert
Kabeldurchführung
Antenne
Durchführung Kameraobjektiv

Gehäuse


Befestigung STM32
mittels M3 Schrauben
Befestigung GPS
mittels M3 Schrauben
Befestigung IMU
mittels M3 Schrauben
Befestigung Kamera

Gehäuse

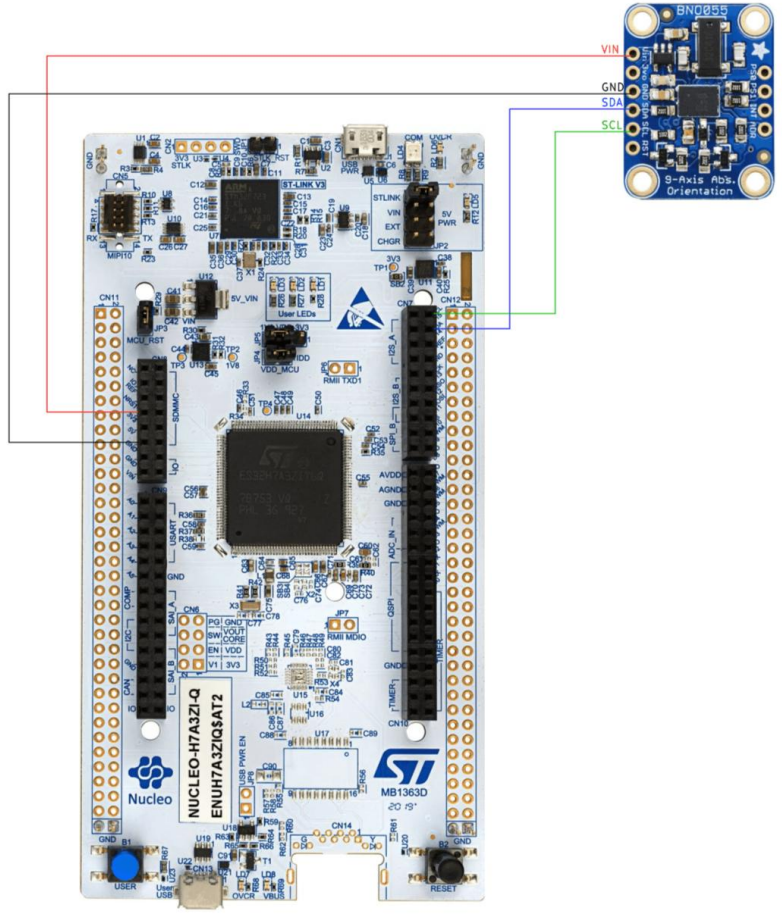

Befestigung Kamera
Kabeldurchführung Antenne
IMU
GPS
M12 T-kodiert
STM32

Software

IMU-Datenerfassung

- **Protokoll:** I2C
- **Ausgelesene Daten:**
  - **Orientierung:** Roll, Pitch, Yaw
  - **Lineare Beschleunigung:** X, Y, Z
  - **Kalibrierungsstatus:** sys, gyro, accel, mag

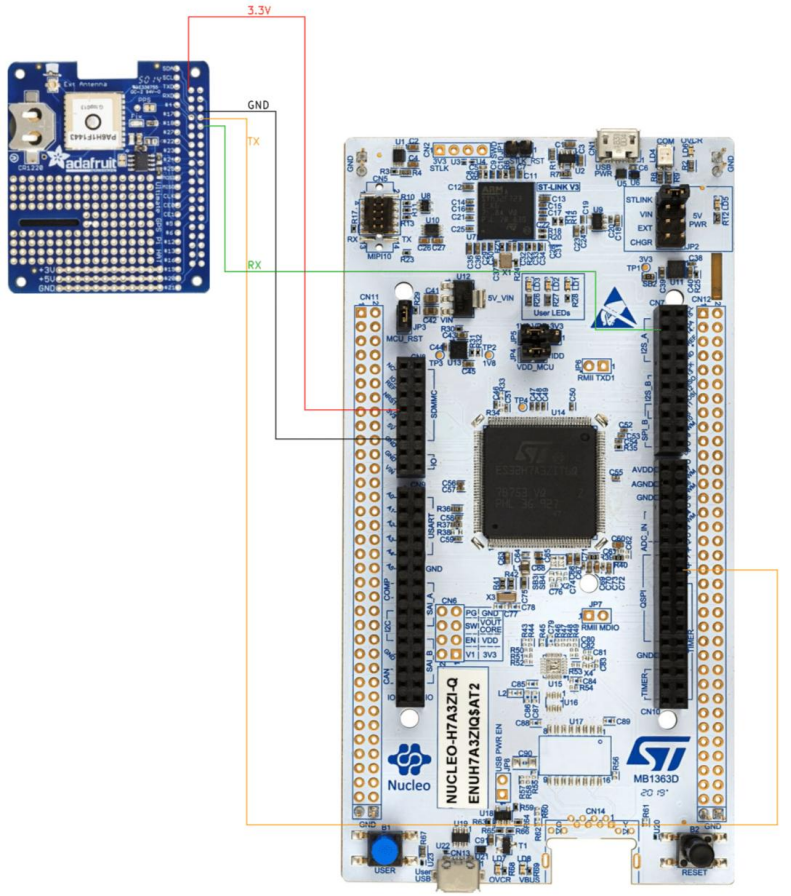
- **Daten-Beispiele:**
  - Orient: x= 195.12, y= 4.44, z= -8.25
  - Linear: x= 0.01, y= -0.03, z= 0.17
  - sys= 2, gyro= 3, accel= 3, mag= 3



GPS-Datenerfassung

- **Protokoll:** UART
- **Ausgelesene Daten:**
  - **$GNGGA:** Breitengrad, Längengrad, Uhrzeit
  - **$GNRMC:** Breitengrad, Längengrad, Uhrzeit, Datum

- **Daten-Beispiele:**
  - $GNGGA,193353.000,4738.6438,N,00910.8235,E,2,15,0.72,432.7,0,M,,*75
  - $GNRMC,193353.000,A,4738.6438,N,00910.8235,E,0.01,292.55,301124,,,D*73

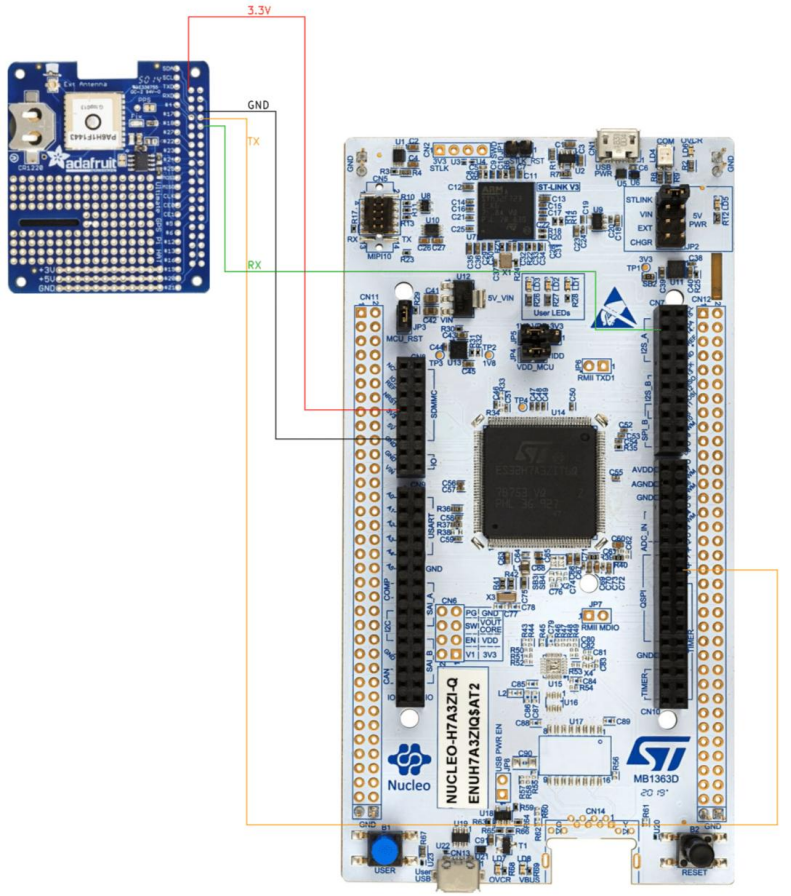


GPS-Datenerfassung

- **Protokoll:** UART
- **Ausgelesene Daten:**
  - **$GNGGA:** Breitengrad, Längengrad, Uhrzeit
  - **$GNRMC:** Breitengrad, Längengrad, Uhrzeit, Datum

- **Daten-Beispiele:**
  - $GNGGA,193353.000,4738.6438,N,00910.8235,E,2,15,0.72,432.7,0,M,,*75
  - $GNRMC,193353.000,A,4738.6438,N,00910.8235,E,0.01,292.55,301124,,,D*73

Uhrzeit Breitengrad Längengrad Datum

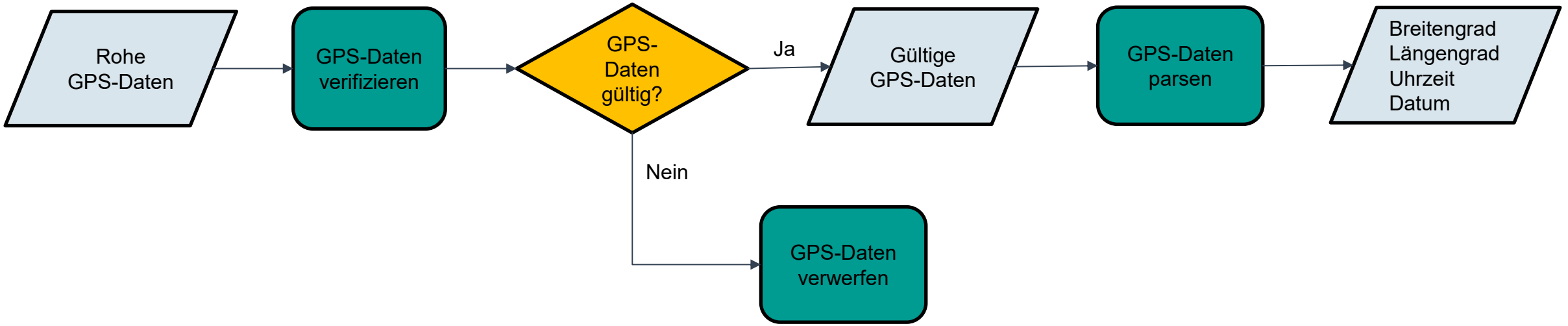


GPS-Datenverifizierung

- **Beispiel:** $GNGGA,193353.000,4738.6438,N,00910.8235,E,2,15,0.72,432.7,0,M,,*75\n
- **Validierungsschritte:**
  1. GPS-Nachricht beginnt mit $ und enthält * → Format korrekt
  2. Nachricht enthält weniger als 75 Zeichen → Länge OK
  3. Berechnung der Checksumme durch XOR-Operation über alle Zeichen zwischen $ und *, und dann Vergleich mit der angegebenen Checksumme (75) → Checksumme gültig

GPS-Datenverifizierung

- **Beispiel:** $GNGGA,193353.000,4738.6438,N,00910.8235,E,2,15,0.72,432.7,0,M,,*75\n
- **Validierungsschritte:**
  1. GPS-Nachricht beginnt mit $ und enthält * → Format korrekt
  2. Nachricht enthält weniger als 75 Zeichen → Länge OK
  3. Berechnung der Checksumme durch XOR-Operation über alle Zeichen zwischen $ und *, und dann Vergleich mit der angegebenen Checksumme (75) → Checksumme gültig

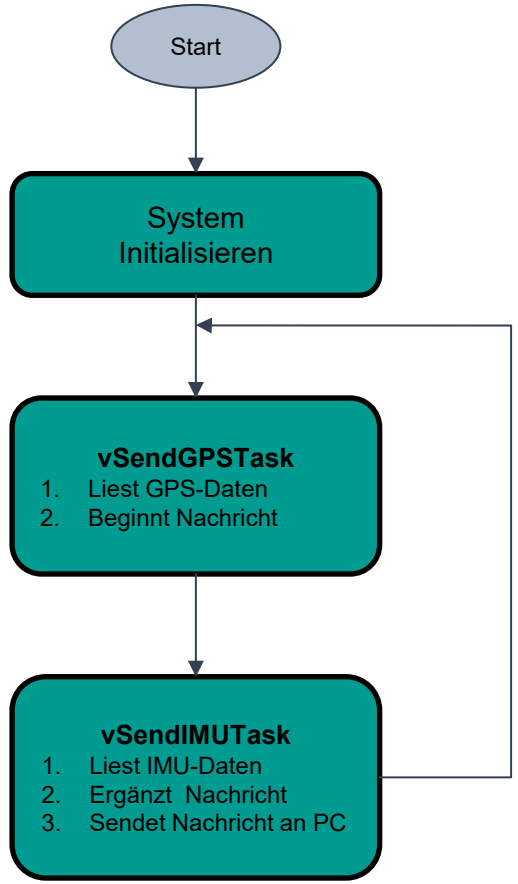


IMU- und GPS-Datenerfassung

- **Synchronisation der Daten zwischen IMU und GPS:**
  - Parallel laufende Tasks für IMU und GPS mit FreeRTOS
  - Verarbeitung der Daten in Echtzeit

- **vSendGPSTask:**
  1. Liest Standort- und Zeitdaten über UART (10 Hz)
  2. Startet die Nachricht mit GPS-Daten

- **vSendIMUTask:**
  1. Liest Beschleunigungs- und Orientierungsdaten über I2C (10 Hz)
  2. Ergänzt die Nachricht mit IMU-Daten
  3. Sendet die komplette Nachricht (GPS + IMU) über USB an den PC



Kalibrierung der IMU

- IMU muss theoretisch bei jedem Neustart kalibriert werden
- Kalibrierung läuft dauerhaft automatisch im Hintergrund
- Bei Bedarf müssen bestimmte Bewegungen ausgeführt werden
- Der aktuelle Status kann über ein Register eingesehen werden

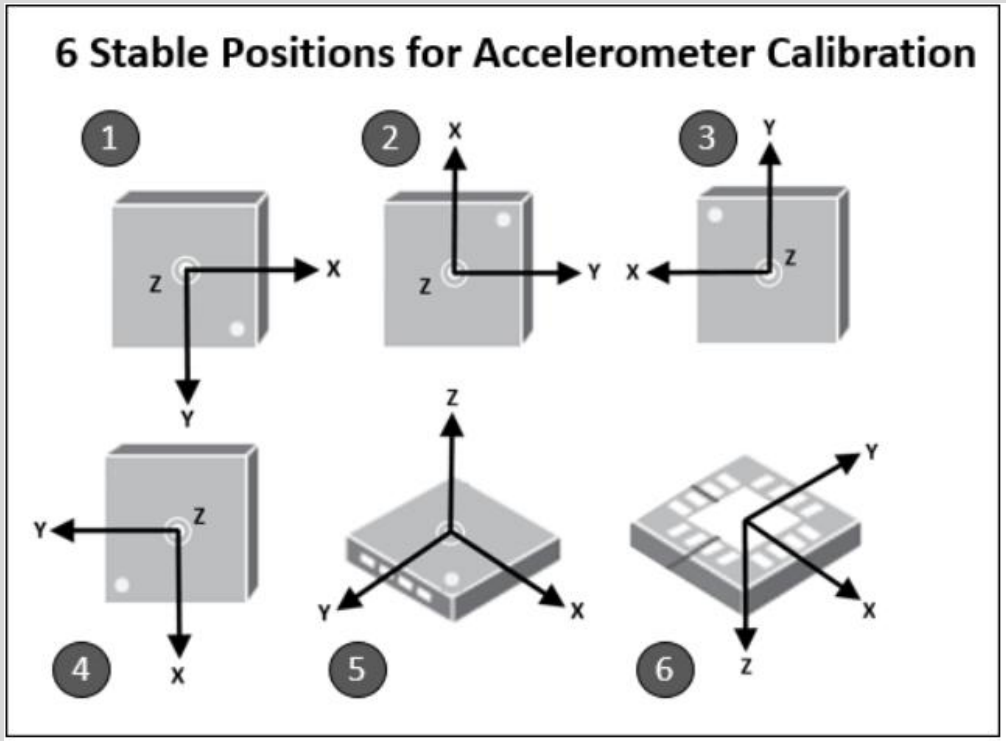
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Access	r	r	r	r	r	r	r	r
Reset	0	0	0	0	0	0	0	0
Content	SYS Calib Status <0:1>		GYR Calib Status <0:1>		ACC Calib Status <0:1>		MAG Calib Status <0:1>	

DATA	bits	Description
SYS Calib Status <0:1>	<7:6>	Current system calibration status, depends on status of all sensors, read-only Read: 3 indicates fully calibrated; 0 indicates not calibrated
GYR Calib Status <0:1>	<5:4>	Current calibration status of Gyroscope, read-only Read: 3 indicates fully calibrated; 0 indicates not calibrated
ACC Calib Status <0:1>	<3:2>	Current calibration status of Accelerometer, read-only Read: 3 indicates fully calibrated; 0 indicates not calibrated
MAG Calib Status <0:1>	<1:0>	Current calibration status of Magnetometer, read-only Read: 3 indicates fully calibrated; 0 indicates not calibrated

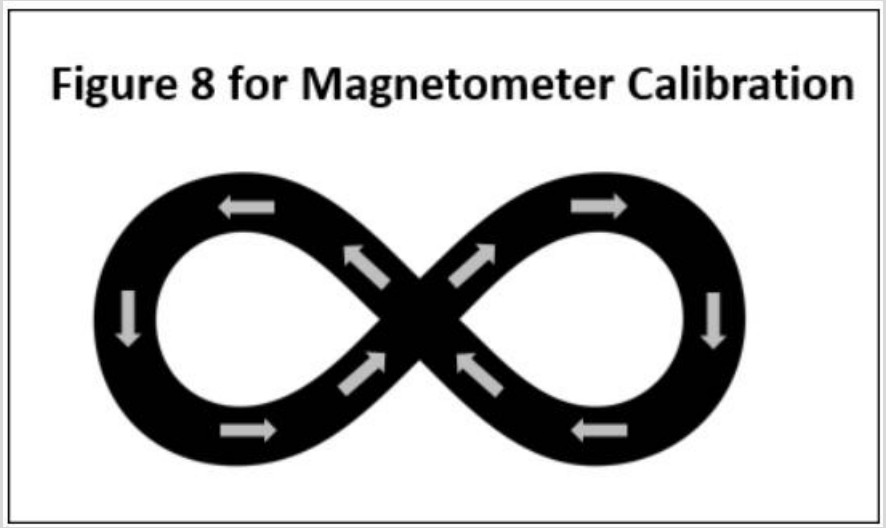
https://www.bosch-sensortec.com/media/boschsensortec/downloads/application_notes_1/bst-bno055-an007.pdf

Kalibrierung der IMU

Accelerometer



Magnetometer



Gyroskop

Stabile Positionierung für einige Sekunden

Kalibrierung der IMU

- Problem:
  - Sensor muss bei jedem Test neu kalibriert werden
  - Kalibrierung mit der gesamten Box eher umständlich
  - Aktueller Status muss einsehbar sein

- Lösung:
  - Aktueller Status wird in der Visualisierung angezeigt (wert 0 bis 3)
  - Bei Bedarf wird angezeigt, dass der eine Kalibrierung durchgeführt werden muss
  - Kalibrierungsdaten werden bei jedem Programmneustart einmalig geladen

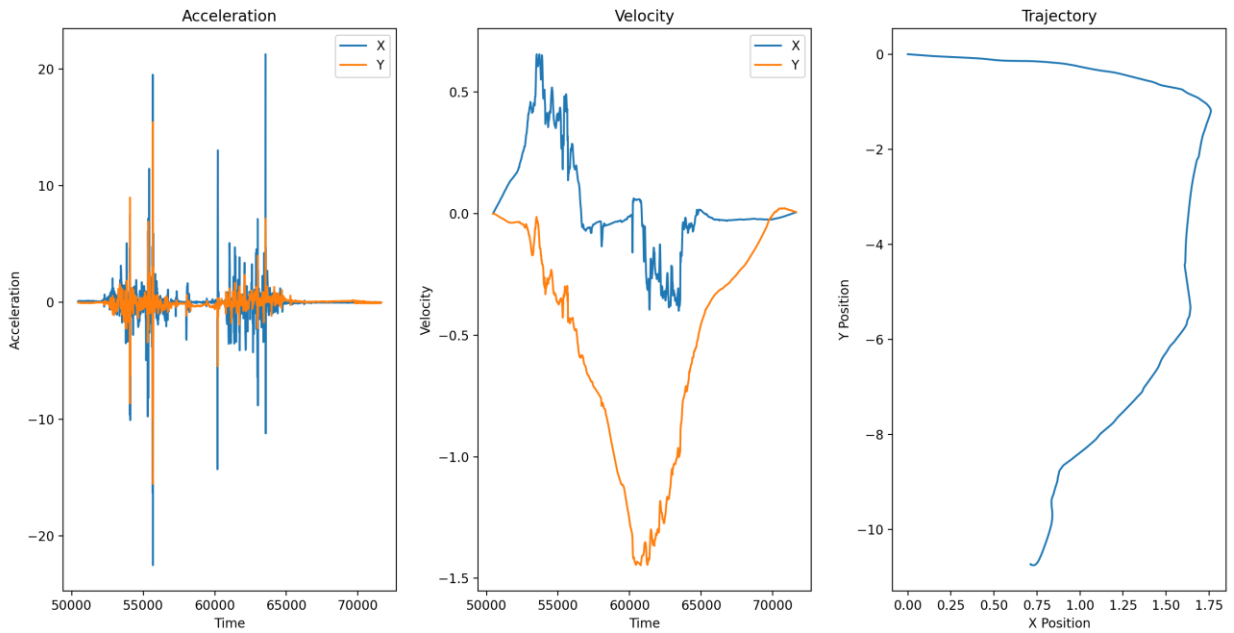
➡ manuelle Kalibrierung dadurch meist nicht notwendig

➡ Wenn doch wird dies Angezeigt und muss vom Nutzer durchgeführt werden

Fusion der Positionsdaten

- GPS-Sensor Genauigkeit maximal 2,5 m
- Verbesserung durch Fusion der IMU-Daten

- Test des IMU-Sensors
  - Beschleunigungsmessung nicht zufriedenstellend
  - Orientierungsmessung
    - Genauigkeit Magnetfeld 2,5°
    - Daten nutzbar

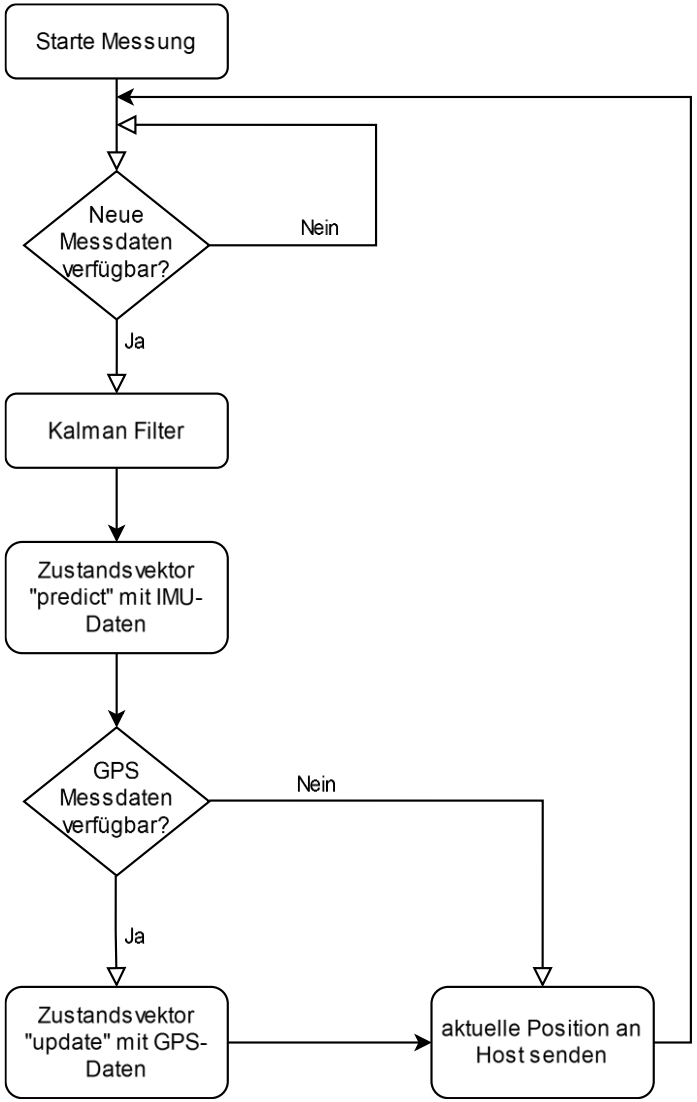


Fusion der Positionsdaten: Kalman-Filter

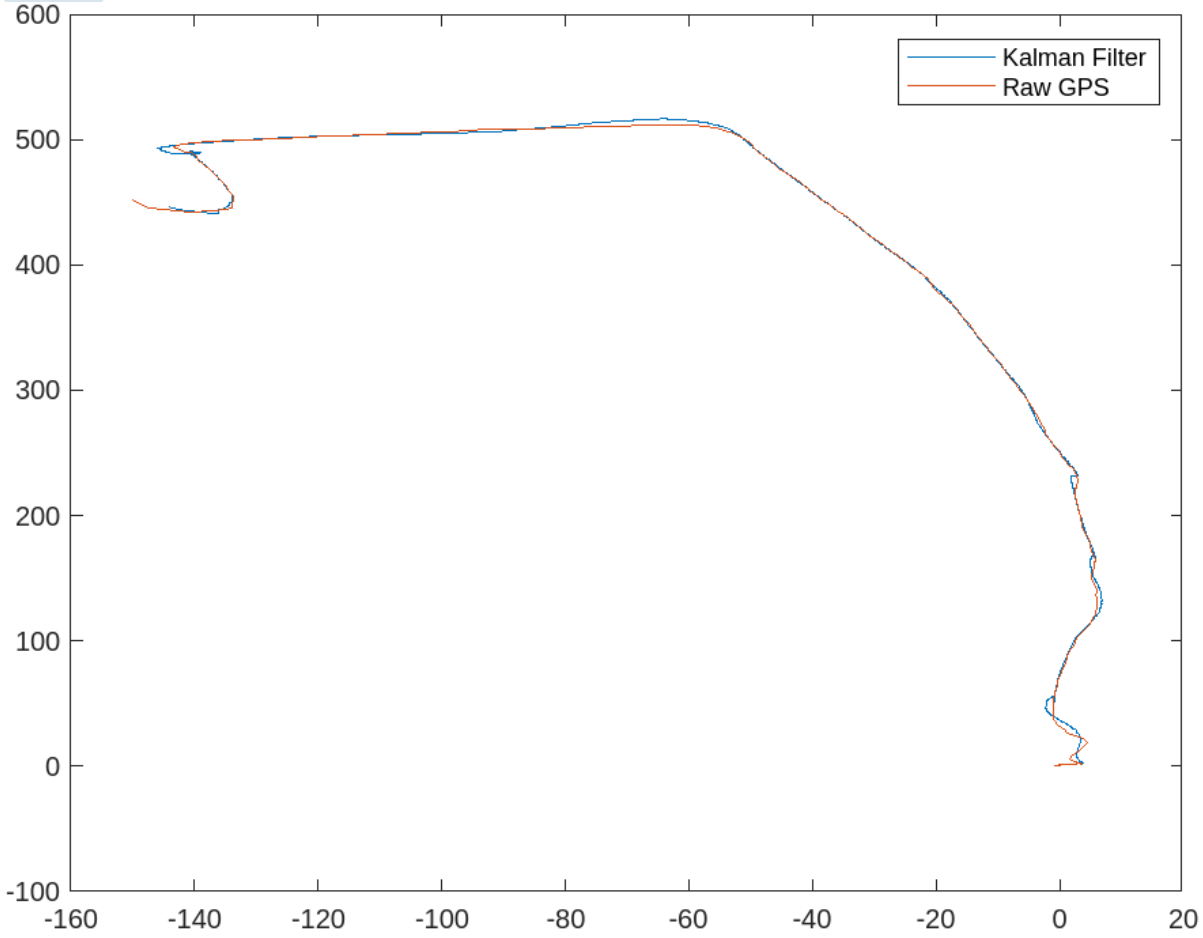
- Zwei Schritte:
  1. Schätzen des Zustands
  2. Aktualisieren mit Messwert

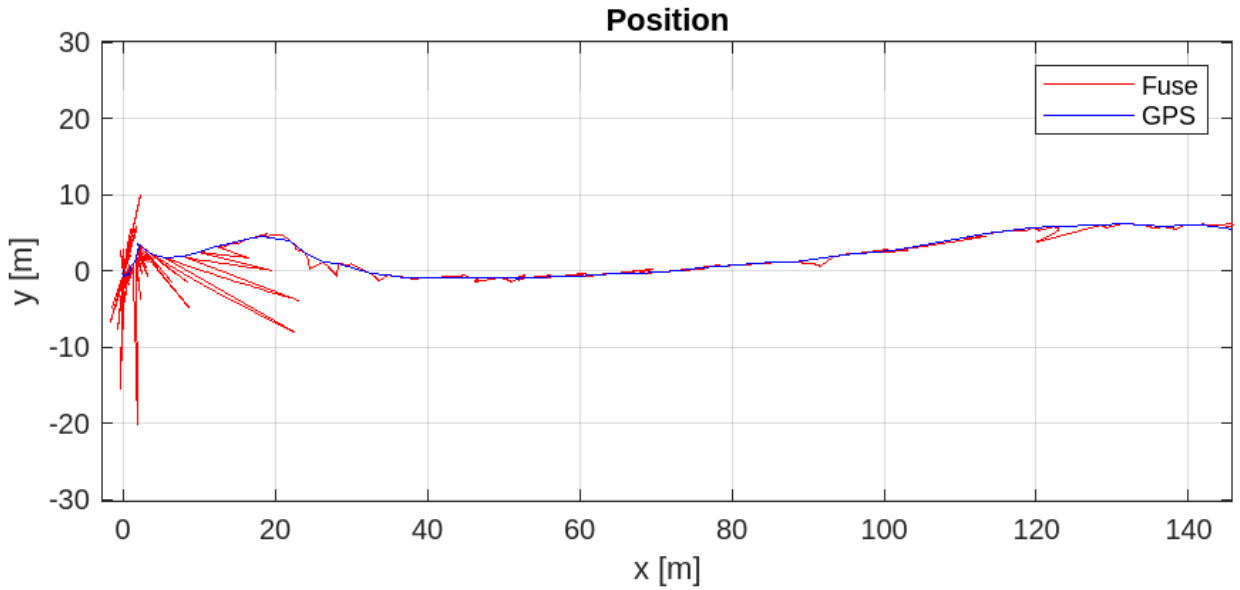
- Verschiedene Zustandsvektoren
  - Position (2D, 3D)
  - Geschwindigkeit
  - Beschleunigung
  - Orientierung/ Heading
  - Winkelbeschleunigung

- In MATLAB und C



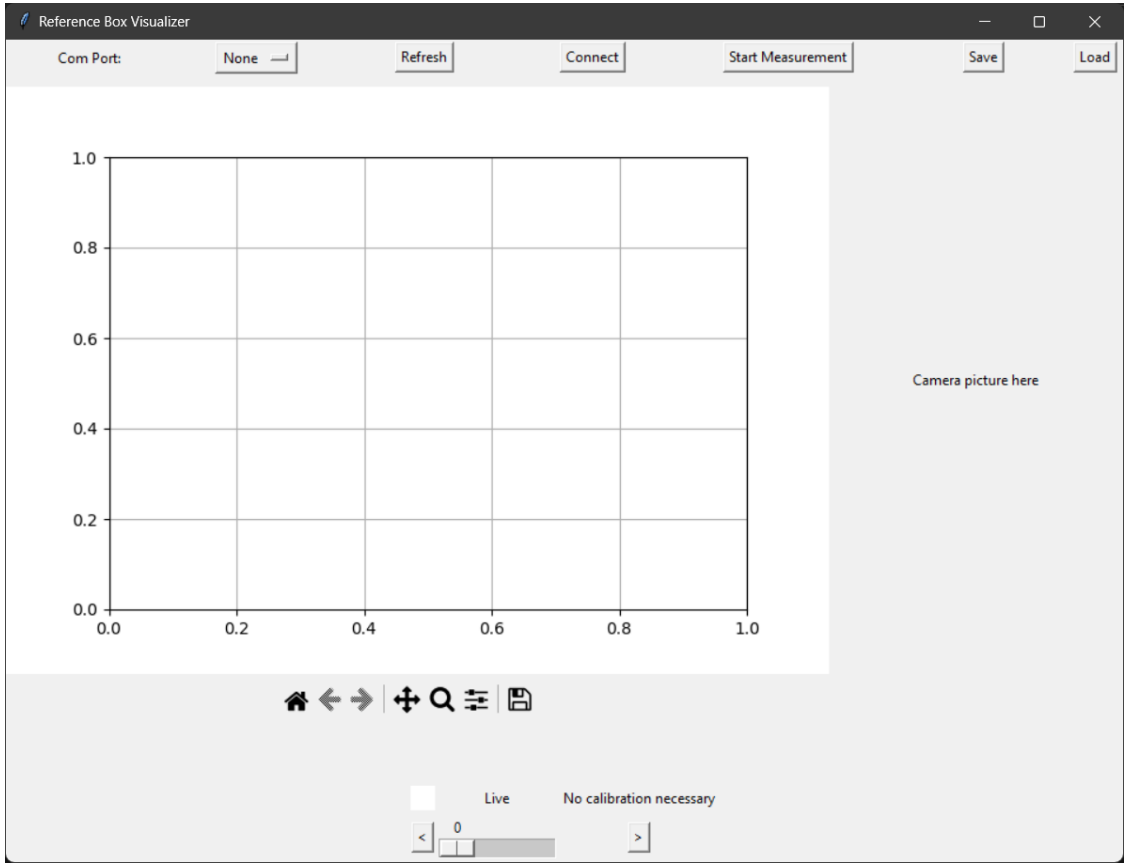
Fusion der Positionsdaten: Kalman-Filter





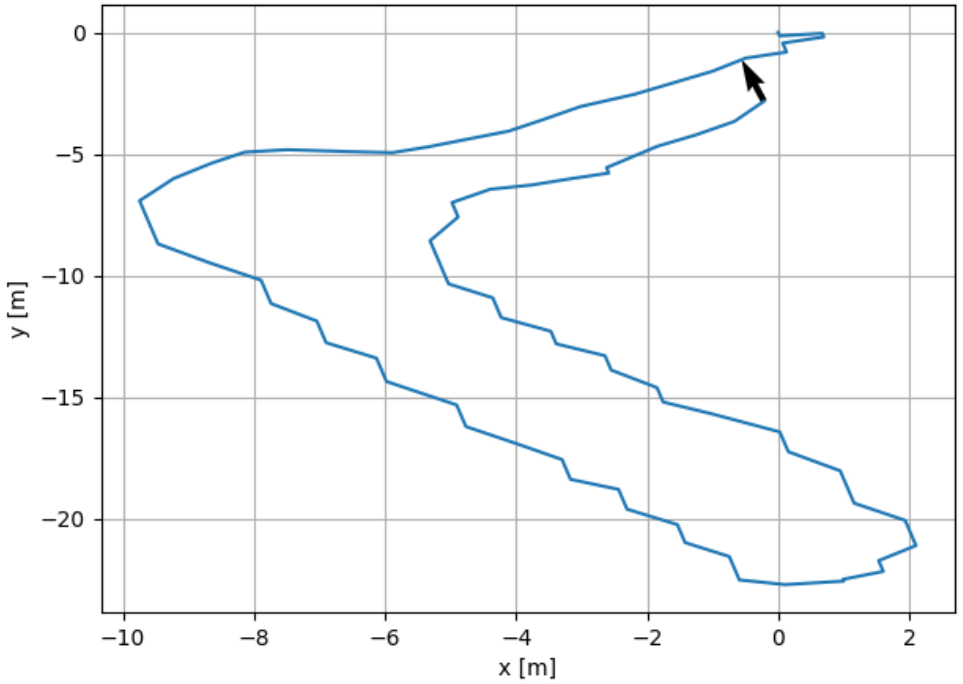
Visualisierung

- Schnittstelle zur Radar Scanner Reference Box
  - Anzeigen des Kalibrationsstatus
- Programmierung mit Threads und Tkinter
- Starten von Messungen
  - Auslesen und Parsen des Datenstreams
  - Abspeichern der Messung in CSV-Datei



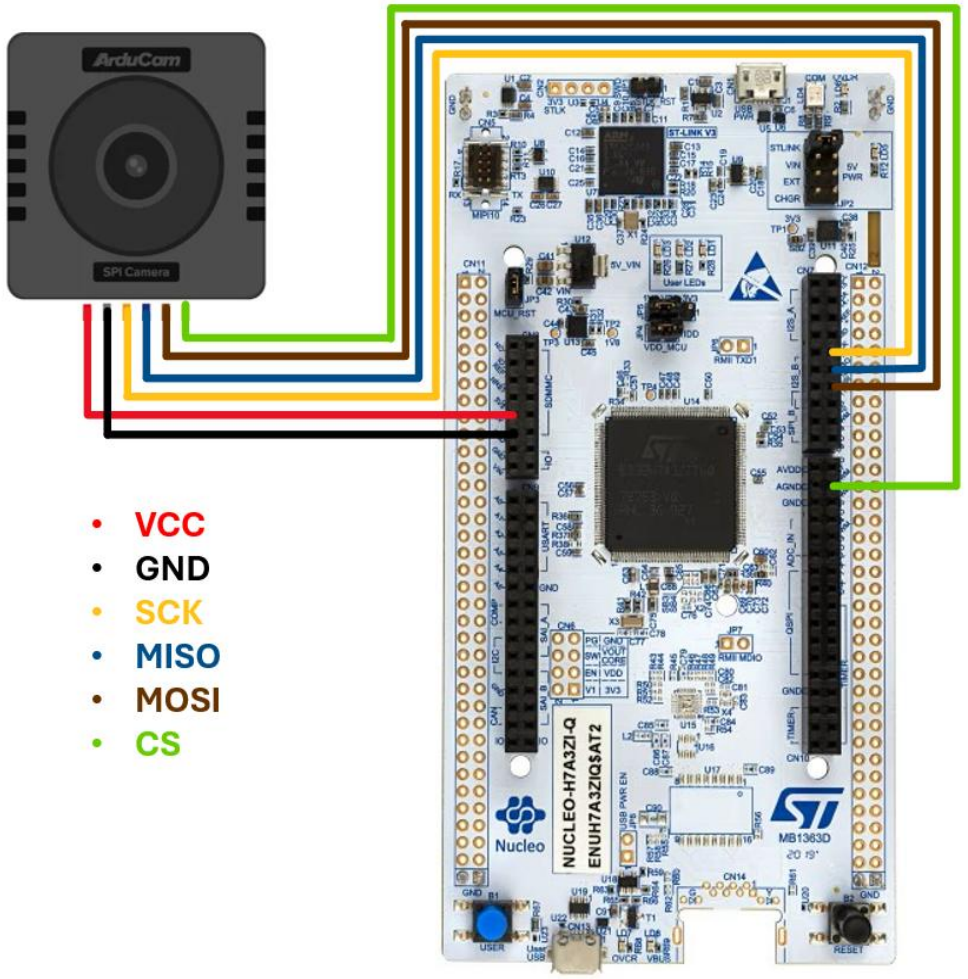
Darstellung Messung

- Darstellung einer gespeicherten Messung
  - Kompletter 2D Verlauf
  - Anpassung von Longitude/Latitude zu X/Y
  - Relativ zum ersten Messpunkt
  - Orientierung für einzelne Frames



Kamera Datenerfassung

- **Protokoll: SPI**
- **Ausgelesene Daten:**
  - Kamera Modus: JPEG, RGB565, YUV
  - Kamera Bild in HEX Werten
- **Daten-Beispiele:**
  - Anfang Bild Nachricht: 0xFF, 0xAA, 0x01
  - Bildlänge: 0xE4 0xD2 0x00 0x00
  - Bildauflösung: 0x41
  - Start JPEG Bild: 0xFF
  - Ende: 0xFF 0xBB



Kamera Datenerfassung

- **Probleme:**
  - Spi-Kommunikation
  - Kamera Library
  - Datenübertragung

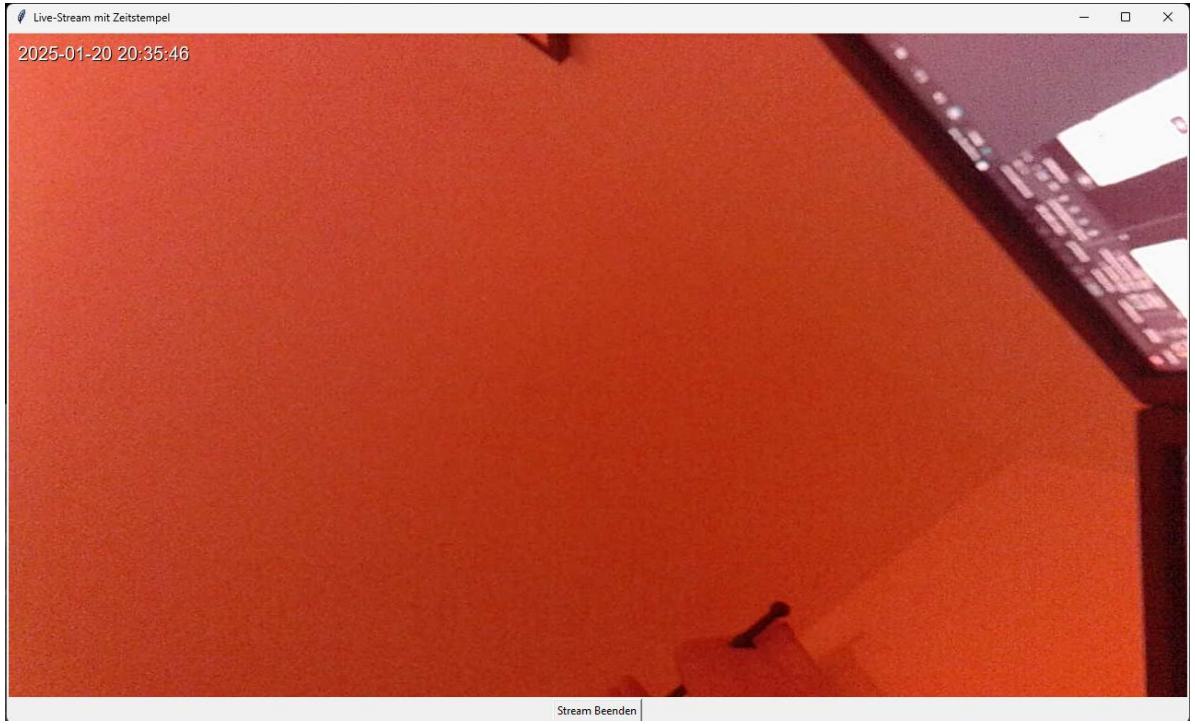
- **Lösung:**
  - Anpassung des Kommunikation Aufbau
  - Vergleich Arduino mit STM-Kommunikation
    - Aufteilung der Receive und Transfer Funktion
    - Anpassung Spi Initialisierung
    - Spi-Takt auf 8 MHz setzten
  - Verwendung der USB OTG Schnittstelle

```c
uint8_t spiReadWriteByte(uint8_t TxData)
{
    uint8_t retry = 0;
    while (SPI_I2S_GetFlagStatus(SPI1, SPI_I2S_FLAG_TXE) == RESET)
    {
        retry++;
        if (retry > 200)
            return 0;
    }
    SPI_I2S_SendData(SPI1, TxData);
    retry = 0;

    while (SPI_I2S_GetFlagStatus(SPI1, SPI_I2S_FLAG_RXNE) == RESET)
    {
        retry++;
        if (retry > 200)
            return 0;
    }
    return SPI_I2S_ReceiveData(SPI1);
}
```

```c
void arducamSpiTransfer(uint8_t data) {
    //uint8_t receivedData = 0;

        HAL_SPI_Transmit(&hspi1, &data, 1, HAL_MAX_DELAY);

        //return receivedData;
}

uint8_t arducamSpiReceive() {
    uint8_t receivedData = 0;

    HAL_SPI_Receive(&hspi1, &receivedData, 1, HAL_MAX_DELAY);
    //HAL_Delay(1);
    return receivedData;
}
```

Kamera Datenverarbeitung

- **Kommunikation-Aufbau**
  - Kamera zurücksetzen: 0x55 0xFF 0xAA
  - Kamera Daten auslesen: 0x55 0x0F 0xAA
- **Qualität einstellen**
  - 0x55 0x01 0x14 0xAA für 1280x720
- **Bild Empfangen**
  - 0x55 0x10 0xAA
- **Datenverarbeitung PC**
  - Qualität einstellen, Bild Empfangen
  - Hex Datei verarbeiten und Darstellen
  - Wiederholen für Stream



Retrospektive

Offene Punkte

- Kamera und GPS/IMU code zusammenführen
- Fusionsalgorithmus:
  - Läuft bisher nicht auf STM
  - Weiteres Feintuning für bessere Ergebnisse
- Anpassungen an der Visualisierung
  - Einbauen des Kamerabildes
  - Möglichen Live-Modus

Herausforderungen

- Komplikationen mit Kameraimplementierung
- Visualisierung sehr abhängig von anderem Entwicklungsstand
- Teils wenige Updates über Entwicklungsstand

- Arbeit sehr entkoppelt voneinander

  -> nicht viele gemeinsame Treffen

  -> Aufgabenpakete oft alleine bearbeitet
- Schwere Koordination des Teams durch unterschiedliche Wohnorte, Vorlesungen, Arbeit und Teamgröße
- Geringe Dokumentation der erledigten Schritte

Ausblick

- Ideen für Weiterentwicklung

  - Verwendung von genauerem GPS-Sensor (RTK)
  - Bei Bedarf: IMU-Modul zur genaueren Bestimmung der Orientierung
  - Verwendung einer hochauflösenden USB-Kamera

  - Separates Auslesen der Daten von GPS, IMU und Kamera
  - ➢ Verarbeitung/Speicherung am PC

Hochschule Konstanz
Fakultät Elektrotechnik
und Informationstechnik

Vielen Dank für Ihre Aufmerksamkeit!