Labor zu Sensortechnik (STP) - WS 2023/24

Lastenheft zum Easy-to-use Quadcopter

Prof. Dr.-Ing. Jörg Dahlkemper

1 Zusammenfassung

1.1 Voraussetzungen

Für dieses Modul werden Kenntnisse aus Bereichen Elektrotechnik, Elektronik und Programmieren in C sowie Mikroprozessortechnik vorausgesetzt, sowie die Bereitschaft, sich in neue Themengebiete und Programmiersprachen einzuarbeiten (insbesondere Python).

1.2 Relevanz

Zunehmende Rechenleistung und kostengünstige Hardware sowie der anhaltende Trend zur Miniaturisierung eröffnen der Sensorsignalverarbeitung immer neue Einsatzbereiche und ein stetiges Wachstum. In diesem Projekt werden die Möglichkeiten und Grenzen des Einsatzes von Sensoren zur Erhöhung des Bedienkomforts eines Quadcopters evaluiert und Kenntnisse der Sensorik, Signalverarbeitung, Elektronik und Programmierung vertieft.

1.3 Ziel des Projektes bestehend aus 4 Laborversuchen

Das Ziel des Projektes teilt sich in folgenden vier Inkremente (demonstrierbare Meilensteine) auf. Das Ziel ist, jeweils am Ende des Laborversuchs diese Ergebnisse funktionsfähig demonstrieren zu können:
1. Mikrocontroller-gesteuerte Fernbedienung eines Quadcopters
2. Zustandsmonitoring des Betriebs des Quadcopters über Machine Learning
3. Multisensordatenverarbeitung zur Steuerung des Quadcopters über die Neigung der Bedieneinheit
4. Entwicklung einer Selfie-Drohne, die Fotos von Gesichtern macht, die Smart AirCam

1.4 Inhalte

- Entwicklung eines IoT-Sensornetzwerkes als Testumgebung für Sensoren am Beispiel Raumklima
- Einführung in das maschinelle Lernen zur Erfassung des Betriebszustandes der Fernbedienung
- Entwicklung einer Schnittstelle zwischen Mikrocontroller und Quadcopter
- Sensorische Erfassung von Benutzervorgaben und Sensordatenfusion von Gyro und Accelerometer
- Einstieg in die Bildverarbeitung und Auswertung des Live-Streams des Quadcopters
- Umsetzung einer Gesichtserkennung zur Ansteuerung der Drohne

1.5 Material

Die zur Verfügung gestellten Komponenten werden in späteren Veranstaltungen wiederverwendet. Daher ist ein festes Verbinden beispielsweise durch Löten oder Crimpen nicht zulässig. Dies betrifft Quadrocopter und Zubehör, Mikrocontrollerboard, Sensormodul und Joystick.

Materialliste STP – Easy-to-use Quadcopter

Anzahl	Bezeichnung	Geprüft
1	Quadcopter Tello EDU mit kleinem Ladekabel	
2	Akkus	
1	Ladehub für Tello Akkus (für Standard-Micro-USB-Ladegerät)	
1	Schutzbrille	
1	Paar Sicherheitshandschuhe (Gr. 8 oder 9)	
2	Steckbretter (Breadboards)	
1	Satz Steckkabel m-m (Jumper wire, ca. 65 Stück, unterschiedliche Längen)	
1	Satz Steckkabel f-m (ca. 10 Stück)	
1	USB-Kabel kurz (blau)	
1	USB-Kabel 1 m	
1	ESP32 Node MCU Module	
1	Temperatur- und Luftdrucksensor BMP280	
1	Beschleunigungs- und Drehratensensor GY-521 MPU6050	
1	Joystickmodul KY-023	
1	Taster rot	

2 Projektablauf

Die Projektdurchführung erfolgt in unabhängig voneinander arbeitenden Gruppen. Es stehen für jedes Team (im Falle von Online-Veranstaltungen: für jedes Teammitglied) ein Miniatur-Quadcopter sowie die zugehörige Hardware zur Verfügung.

Projektraum für Präsenzveranstaltung ist 865 (Maker Space) + ein zusätzlicher Laborraum 804,
Im Falle eines online Labors findet dies in MS Teams im Team MIK/MES-ST statt.

Meilensteine sind am Ende jeden Laborversuchs nachzuweisen,
in Ausnahmefällen bei unvorhersehbaren technischen Problemen eine Woche später.

Labor	Inhalt	Deliverable
1	IoT-Sensornetzwerk	Demonstration der vollständigen Funktion eines Sensorsystems bestehend aus: Auslesen von zwei Joystick-Achsen und Joystick-Taste Auslesen von Notaus-Taste Anzeige der Raumtemperatur Anzeige des Luftdruckverlaufs Theorie und Experiment zur Genauigkeit der Höhenerfassung mittels Luftdrucksensor Dokumentation: - Blockschaltbild und Schaltplan - Softwaredokumentation mit UML-Darstellung (z.B. Ablaufdiagramm)
2 + 3	Betriebszustandsmonitoring	Vorführung der Erkennung der Zustände Ruhe – Fernsteuerung – Transport auf dem Bildschirm des PCs über 1 min fehlerfrei bei wechselnden Zuständen. Dokumentation der folgenden zwei Varianten: a) Klassifizierung auf Basis der Rohdaten b) Klassifizierung auf Basis von vorverarbeiteten Daten (z.B. statistische Merkmale) Zu jeder Variante soll dargestellt werden: - welche Sensoren/Merkmale werden genutzt - welche Accuracy wird auf Testdatensatz erreicht - Programmlisting (graphische Darstellung optional

4	Lagegesteuerter Quadrocopter unter Einsatz von Sensordatenfusion	Präsentation der Lagesteuerung über µC: - Starten und Landen über Joy-Stick-Tastendruck - Ansteuerung Rollen/Nicken über Neigung der Fernbedienung - Ansteuerung Schub/Gieren über Joystick - Demonstration Not-Aus aber: kein erkennbarer Einfluss bei horizontaler Verschiebung der Fernbedienung auf Tischplatte in horizontaler Position Dokumentation: - Darstellung der Sensordatenfusion in Theorie und Umsetzung (mit Blockschaltbild, formelmäßiger Darstellung, gewählte Parameter, Ablaufdiagramm) - Nachweis der Driftfreiheit - Schaltplan - Softwaredokumentation mit UML-Darstellung (z.B. Ablaufdiagramm mit expliziter Darstellung der Vorverarbeitung (Kalibration, Filterung) - Foto des Aufbaus sowie des Quadcopters im Flug
5 + 6	Smart AirCam	Präsentation der Foto-Drohne mit PC-basierter Steuerung: - Automatisches Einnehmen der Flughöhe von 1,50 m - Live-Anzeige des Videostreams des Quadcopters - Automatisiertes Erkennen und Markieren des zuerst gefundenen Gesichtes im Live-Bild (optional: eines zuvor eingelernten Gesichtes) - Rotation des Quadcopters auf der Stelle, bis das erste Gesicht mittig im Bild aufgenommen wird - Aufnahme von 2 Bildern im Abstand von 2 Sekunden - Anschließendes Landen des Quadcopters Dokumentation als Sofort-Protokoll: - Software-Dokumentation mit UML-Darstellung - Abschlussfoto des Teams zur eigenen Verwendung, gerne auch in Emil hochladen

3 Bewertungskriterien der Prüfungsvorleistung

Für die erfolgreiche Vergabe der Prüfungsvorleistung sind die in Abschnitt 2 dargestellten Deliverables grundsätzlich am Ende eines Praxisblocks bestehend aus 1 oder 2 Laborterminen nachzuweisen.

Es sind hierbei die in Kapitel 5 dargestellten Anforderungen zu berücksichtigen.

4 Hardwareaufbau

Die Hardware wird auf Breadboards aufgebaut und mit den Jumper-Kabeln ohne Löten verbunden. Hierbei ist unbedingt die interne Verdrahtung eines Breadboards zu beachten und zu verstehen:

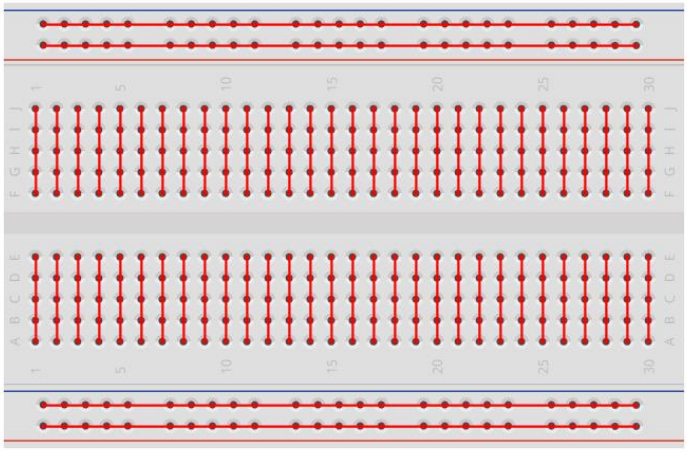
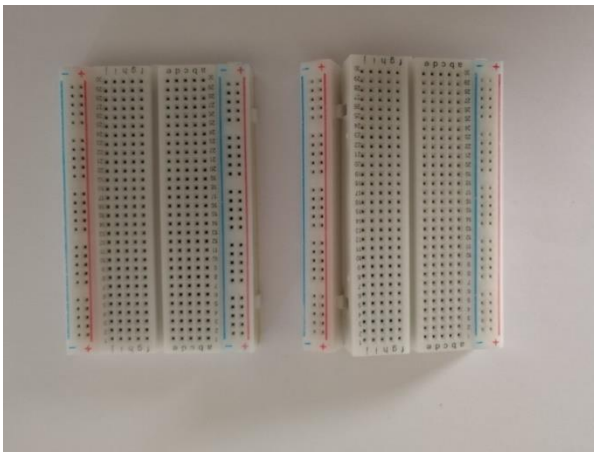


Abbildung 1: Aufbau eines Breadboard [https://www.datenreise.de/raspberry-pi-wie-verwendet-man-ein-breadboard-steckplatine-anleitung/]

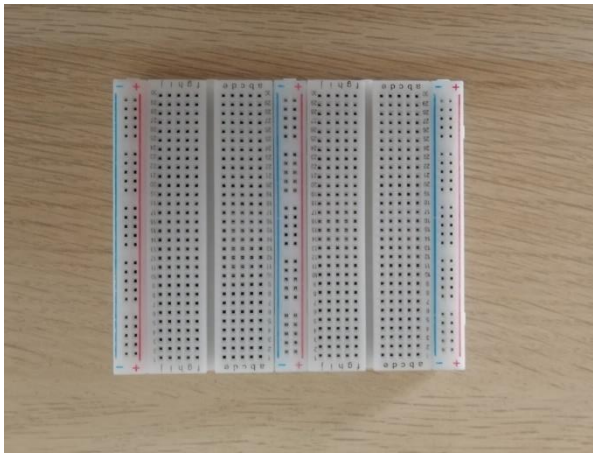
Das verwendete Microcontroller-Board ist ein OpenSource-Entwicklungsboard mit der Bezeichnung ESP32 NodeMCU. Der Prozessor arbeitet mit einer Taktrate von bis zu 240 MHz, 4 MB Programmspeicher und 520 kB RAM (zum Vergleich: ein Arduino Uno Rev 3 hat eine Taktrate von 16 MHz, 32 kB Programmspeicher, 2 kB RAM). Das Board ist ein leistungsfähiger Nachfolger des ESP8266. Es eignet sich insbesondere für IoT-Anwendungen, da es über integriertes WLAN und Bluetooth verfügt.

Das Mikrocontrollerboard ist für die Montage auf einem Breadboard zu breit, daher ist es erforderlich, zwei Breadboards zu kombinieren und eine der Spannungsschienen zu entfernen. Die Anpassung ist dank des modularen Aufbaus der Breadboards einfach umsetzbar und auf folgender Seite in Abbildung 2 dargestellt.

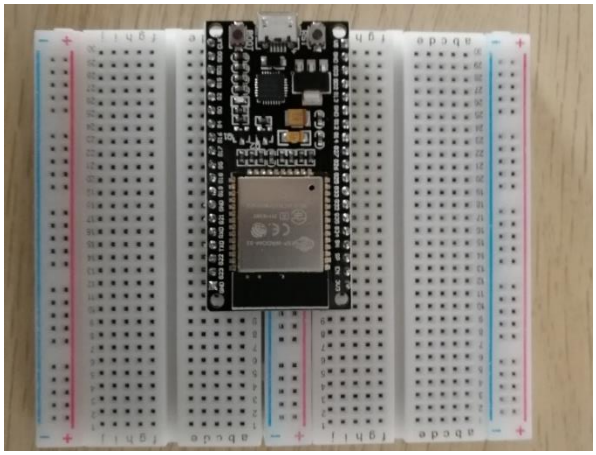


a) eine Spannungsschiene herausschieben

b) auf Gegenstück Klebeband kürzen



c) zusammenschieben und Klebeband andrücken



d) Seite mit 3,3 V erhält eine Reihe mehr

Abbildung 2: Kombination von zwei Breadboards nach Entfernen einer Spannungsschiene

5 Anforderungen

Die nachfolgend dargestellten Anforderungen sind umzusetzen und stellen sicher, dass der verfolgte Lösungsweg zu einem funktionsfähigen System führen kann. Diese Anforderungen sind nicht als Empfehlung zu verstehen, sondern als harte Forderung der Umsetzung und müssen entsprechend gelesen und beachtet werden.

Anforderungen an einen Laborbericht

Der Bericht muss systematisch gegliedert sein und auf die zu jedem Inkrement geforderten Ergebnisse eingehen.

Bei der Erstellung eines Berichts ist insbesondere auch auf eine saubere äußere Form sowie auf Rechtschreibung und Kommasetzung zu achten.

Der Laborbericht soll je Inkrement 10 Seiten nicht überschreiten.

5.1 Inkrement 1: IoT-Sensornetzwerk

Als Basis für die darauf aufbauenden Versuche soll ein Multisensornetzwerk aufgebaut werden und die korrekte Funktion mehrerer Sensoren einschließlich des Joysticks über einen Internetbrowser graphisch visualisiert werden. Auf dieser Grundlage soll ermittelt werden, wie genau eine Höhenmessung über den Luftdruck ist.

5.1.1 Vorbereitung

Erstmalige Softwareinstallation ist durchaus zeitaufwändig, daher bitte vorab erledigen und testen.
Bei Nutzung eines Laborrechners entfallen die ausgegraut dargestellten Installationen.

- Arduino IDE (z.B. über MS Store oder https://www.arduino.cc/en/Main/Software)
- Schnittstellentreiber für das ESP32-Board:
  bei Windows wird mit Verbinden des Boards automatisch ein veralteter Treiber installiert, daher ist unabhängig vom Betriebssystem ein aktueller Treiber zur Kommunikation zwischen PC und Board von https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers zu installieren.
  Zu Win10: Bei erfolgreicher Installation des Treibers wird nach Verbinden des ESP32 Boards mit dem PC dieses im Gerätemanager unter „Anschlüsse“ als „Silicon Labs CP210x USB to UART Bridge“ angezeigt. Hier kann man den Port ablesen, der dem Board zugewiesen wurde.
- ESP32 Board in Arduino IDE einbinden
  in Arduino IDE: Datei > Voreinstellungen, dort in „Zusätzliche Boardverwalter-URLs“ folgenden Eintrag ergänzen: https://dl.espressif.com/dl/package_esp32_index.json
  Dann in Arduino IDE: Werkzeuge > Board > Boardverwalter … auswählen und „esp32 by Espressif Systems“ installieren und als Board „ESP32 Dev Module“ auswählen. Anschließend unter Werkzeuge die Upload-Speed auf 115200 reduzieren und den korrekten Port auswählen.
- Funktionstest des ESP32 NodeMCU Boards:
  in Arduino IDE: Datei > Beispiele > ESP32 > ChipID > GetChipID laden
  Zum Hochladen müssen Sie auf die Schaltfläche Hochladen von der Arduino IDE klicken und evtl. die Taste „BOOT“ auf dem NodeMCU ESP32 gedrückt halten. Das Hochladen ist abgeschlossen, bis das Beschreiben 100% erreicht hat und Sie aufgefordert werden, einen Neustart durchzuführen (Hardresetting via RTS pin...) mit der Taste „RST“. Üblicherweise erfolgt dies ohne Nutzereingriff. Die Ausgabe des Beispielprogramms können Sie im seriellen Monitor mit der Baudrate 115200 unter Werkzeuge aufrufen.
- Installation der Bibliothek „Adafruit BM**P**280 Library“ und „PubSubClient von Nick O’Leary“ über die Arduino IDE > Werkzeuge > Bibliotheken verwalten

- Installation des MQTT-Brokers Mosquitto von http://mosquitto.org/download und die Option „Service“ abwählen, damit dies nicht standardmäßig im Hintergrund ausgeführt wird.
- Node.js als Voraussetzung für Node-RED von https://nodejs.org/en/
  Dabei die automatische Installation zu zusätzlichen Komponenten (Chocolatey) deaktivieren.
- Node-RED in Konsole über `npm install -g --unsafe-perm node-red`

Für spätere Versuche:

- Die aktuelle Python-Version über Anaconda installieren:
  https://www.anaconda.com/products/individual
  Bei der Installation bitte die empfohlenen Installationsoptionen auswählen.
  Zur Python-Programmierung empfiehlt sich folgendes Vorgehen:
  Im Windows-Startmenu `Anaconda > Anaconda Prompt` aufrufen. In der dann erscheinenden Eingabekonsole per `<X>:` auf das gewünschte Laufwerk X und dann per `cd` das Verzeichnis wählen. Dann beispielsweise die Entwicklungsumgebung Jupyter notebook durch Eingabe des Befehls `jupyter notebook` aufrufen oder eine Entwicklungsumgebung wie VS Code nutzen.
- Eine Python-Umgebung für die Veranstaltung ST einrichten. Dazu in der Eingabeaufforderung des Anaconda prompts die aktive Entwicklungsumgebung (base) verlassen, indem der Befehl
  conda deactivate
  aufgerufen wird und anschließend eine neue User-spezifische Entwicklungsumgebung mit dem Namen ST durch folgenden Befehl einrichten:
  `conda create --name ST anaconda`
  Diese ist anschließend zu aktivieren, indem in der Eingabeaufforderung
  `conda activate ST`
  aufgerufen wird. Im Prompt erscheint vorangestellt die Bezeichnung der aktivierten Umgebung. Die erstellte Entwicklungsumgebung ist jedes Mal beim Neustart des Anaconda Prompts durchzuführen.
- Dort in der aktivierten Umgebung ST per
  `pip install pyserial opencv-python`
  die Bibliothek zur seriellen Kommunikation mit dem ESP und zur Bildverarbeitung einbinden.

5.1.2 Microcontroller-Board

Es ist ein Mikrocontroller-Entwicklungs-Board des Typs ESP32 NODE-MCU zu verwenden, der über eine integrierte WLAN-Schnittstelle verfügt und über eine Arduino IDE programmiert werden kann. Dieser Microcontroller soll die Daten eines Joysticks einlesen und eines Sensormoduls für die Luftdruck- und Temperaturmessung.

Hinweise:
Das Mikrocontroller-Board wird über den USB-Port mit einer Spannung von 5 V versorgt, arbeitet intern aber mit 3,3 V und diese Spannung von 3,3 V darf bei Anschaltung von Peripherie nicht überschritten werden.

5.1.3 Sensoren

Es sind die folgenden Sensoren an den Mikrocontroller anzubinden und auszulesen:
Sensor für Temperatur und Luftdruckmessung, 2 Joystick-Achsen, Taster des Joysticks, Notaus-Taster

Hinweise zum Temperatur- und Luftdrucksensor:

Für die Erfassung der Temperatur und des Luftdrucks ist ein Sensorboard auf Basis des BMP280 zu verwenden.

https://www.az-delivery.de/products/azdelivery-bmp280-barometrischer-sensor-luftdruck-modul-fur-arduino-und-raspberry-pi

Das Sensorboard ist über den I2C-Bus mit dem Mikrocontroller in Hardware und Software zu verbinden. Das verwendete ESP32-Board nutzt G21 als SDA und G22 als SCL für den Standard-I2C-Bus. Dieser kann damit wie bei einem Arduino über die Bibliothek wire.h genutzt werden:

Tutorial zu I2C für ESP: https://randomnerdtutorials.com/esp32-i2c-communication-arduino-ide/
Beispieldatei zum Scannen des I2C-Busses: I2Cscan.ino
Schließen Sie den Sensor dazu so an, dass er auf der I2C-Adresse 0x77 arbeitet und überprüfen Sie dies über das Programm I2Cscan.ino. Siehe dazu das Bosch-Datenblatt, https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bmp280-ds001.pdf , S. 28 und 29 oben.

Anwendungsbeispiele mit BMP280:

Minimalversion: Es ist die Bibliothek „Adafruit BMP280 Library“ über die Bibliotheksverwaltung der Arduino Entwicklungsumgebung einzubinden und wie mit dem im Git-Repo der Vorlesung mit der Datei BMP280.ino dargestellt, zu nutzen. Diese Library erwartet den Sensor BMP280 an der I2C-Adresse 0x77.

Hinweise zum Joystick:

Die Tastfunktion des Joysticks verbindet den Ausgang SW (switch) bei Druck mit GND. Der gewählte Pin sollte also über einen internen Pull-up Widerstand verfügen, wie G25 und in der Betriebsart INPUT_PULLUP konfiguriert werden.

Zum Einlesen von Analogwerten bieten sich die Eingänge ADC1_CH4 an G32 und ADC1_CH5 an G33 an. Anders als bei einem Arduino ist bei dem ESP32 die Bibliothek adc.h einzubinden und es ist die Auflösung auf 10 Bit einzustellen (1024 Werte) und dieser ist so zu konfigurieren, dass der Wertebereich von 0 bis 3,3 V reicht. Siehe hierzu das eigene Beispiel analog_input.
Der Joystick ist sehr kostengünstig. Finden Sie heraus, wie sich dies auf die Kennlinie des Joysticks auswirkt

Hinweise zum Notaus-Taster:
Auch für den Taster sollte ein I/O-Pin mit internem Pull-up Widerstand verwendet und entsprechend konfiguriert werden.

5.1.4 Sensornetzwerk

Die Sensordaten sollten unter Nutzung des Protokolls MQTT zur Verfügung gestellt werden. Als MQTT-Server ist mosquitto (http://mosquitto.org) empfohlen.

Hinweise zur Konfiguration des MQTT-Brokers mosquitto:

Im Labor: MQTT-Server wird nach Verbinden mit dem WLAN Muecke ein MQTT Server unter der IPAdresse von Muecke (z.B. 192.168.42.1) zur Verfügung gestellt. Zur Kommunikation mit dem MQTTServer ist entsprechend diese IP anstelle der 127.0.0.1 des eigenen Rechners zu verwenden.

Bei eigener MQTT-Installation:

Das Ausführen durch einen Doppelklick startet den MQTT-Broker nur im Lokalen Modus. Verbindungen zum Broker sind nur innerhalb des Computers möglich, aber nicht von außerhalb. Für die Konfiguration muss unter Windows die Datei C:\Program Files\mosquitto\mosquitto.conf um folgendes ergänzt werden. Dabei ist die IP Adresse des Computers einzusetzen und es darf keine # (Kommentarzeichen) am Anfang der folgenden Zeilen stehen:

listener 1883 192.168.xxx.xxx
allow_anonymous true

Für die Bearbeitung ist es notwendig, einen Editor mit Administratorrechten zu starten. Im Startmenü "Editor" suchen und mit Rechtsklick "Als Administrator ausführen" öffnen.

Datei > Öffnen auswählen
Zu C:\Program Files\mosquitto navigieren
Unten neben dem Öffnen Button statt "Textdatein (*.txt)" "Alle Datein (*.*) auswählen
mosquitto.conf öffnen und bearbeiten

Zum Start mit der geänderten Konfiguration des MQTT-Brokers:

Eingabeaufforderung öffnen
cd C:\Program Files\mosquitto
mosquitto -v -c mosquitto.conf

5.1.5 Visualisierung

Zur Visualisierung der Sensordaten soll die Software Node-RED genutzt werden, eine graphische Programmierumgebung, die häufig bei Home Automation-Projekten Verwendung findet.

Dazu ist in der Eingabeaufforderung
`node-red`
einzugeben. Wenn der Befehl nicht gefunden werden kann, ist zunächst sicherzustellen, dass die Kapitel 5.1.1 dargestellten Schritte ausgeführt wurden. Wenn der Befehl node-red aufgrund eines fehlenden Pfad Eintrags nicht gefunden wird, direkt in das Verzeichnis mit node-red wechseln. Dies ermittelt man per
`npm list -g`
Nach Aufruf von node-red wird in den erscheinenden Status-Meldungen die korrekte Verbindung mit dem MQTT-Broker auf Port 1883 angezeigt und die Adresse des Node-RED-Servers
`http://127.0.0.1:1880/` angezeigt werden.

Node-RED ist eine Browser-gestützte Entwicklungsumgebung und wird über einen Browser durch Eingabe der Adresse http://127.0.0.1:1880/ aufgerufen. Da es sich nicht um eine geschützte https-Verbindung handelt, muss der Zugriff auf die potentiell unsichere Seite im Browser in der Regel explizit über ein erweitertes Menu bewusst zugelassen werden.

Zur Visualisierung der Prozessdaten, ist zunächst eine Library für ein Dashboard über den Menu-Button rechts oben neben dem Deploy-Button zu laden. Hierzu ist im Menu `Manage palette` auszuwählen und im Tab Install `node-red-dashboard` einzugeben und die Installation auszuführen.

Machen Sie sich mit der graphischen Programmierung von Node-RED in einschlägigen Tutorials vertraut und präsentieren Sie die folgenden Sensordaten ansprechend und sinnvoll aufgeteilt.

- Stellung jeder Joystickachse per Gauge (Zeigerinstrument)
- Drücken des Joy-Stick-Tasters
- Drücken des Notaus-Tasters über grün / rot
- Temperatur und Luftdruck jeweils als Diagramm und als Textfeld

5.1.6 Genauigkeitsuntersuchung Höhenmessung

Ermitteln Sie unter Nutzung der Datenblattangaben des Luftdrucksensors, wie genau Sie eine Höhenänderung mittels Luftdrucksensor messen können und überprüfen Sie Ihr Ergebnis unter Nutzung von Mittelwert und Standardabweichung experimentell über Analyse von Mittelwert und Standardabweichung in 2 um 50 cm voneinander abweichenden Höhen.

Beachten Sie, dass der BMP280 gemäß Datenblatt die Sensormessung nur alle 500 ms aktualisiert. Wenn dieser häufiger abgefragt wird, ergibt sich eine Mehrfachabtastung desselben Wertes, was die Standardabweichung fälschlicherweise reduziert. Daher ist die Messrate auf 2 Hz zu reduzieren. Die Messreihe sollte 30 s nicht überschreiten, da andernfalls veränderte Umgebungsbedingungen wie der Einfluss der Raumbelüftung die Standardabweichung ungünstig beeinflussen.

5.1.7 Sicherheit

Die vorgestellte Implementierung weist erhebliche sicherheitstechnische Schwächen auf, da beispielsweise das WLAN-Kennwort im ESP32-Code im Klartext hinterlegt und auf Ihrem Rechner gespeichert wird. Weiterhin erfolgt die Übertragung der Sensordaten per http unverschlüsselt. Der MQTT-Broker ist ohne Kennwortschutz konfiguriert.

Damit eignet sich dieser Versuch ausschließlich zum kurzzeitigen Demonstrationsbetrieb und es ist während der Versuche ein temporäres WLAN-Kennwort zu verwenden oder anschließend das Kennwort zu ändern.

Aktuelle Browser lassen einen Zugriff auf nicht https-geschützte Seiten nur über das explizite Freigeben der unsicheren Verbindung zu.

Ferner ist zu beachten, dass der externe Zugriff auf den Mosquitto-Server durch den ESP32 in der Regel über die Standard-Firewall eines Rechners geschützt wird und daher erst der Port explizit in der Firewall freigeschaltet werden muss. Auch dies stellt ein potentielles Sicherheitsrisiko dar und sollte bei nicht durch die Firewall des WLAN-Routers geschützten Rechnern nur temporär für die Dauer des Versuchs zugelassen werden. Unter Windows erfolgt dies über Einstellungen > Windows-Sicherheit > Firewall & Netzwerkschutz über „Kommunikation von Apps durch die Windows-Defender Firewall zulassen“ für das Programm mosquitto.exe.

5.2 Inkrement 2: Betriebszustandsmonitoring

Zu Demonstrationszwecken ist eine Überwachung des Betriebszustandes auf Basis der Motion Processing Unit MPU 6050 zu realisieren, die erkennt, ob die im 3. Inkrement zu entwickelnde Fernbedienung ruht, transportiert wird oder ob ein aktiver Steuerbetrieb stattfindet.

5.2.1 Sensor-Messwerte

Es ist das Mikrocontroller-Board ESP32 zu verwenden.

Das Mikrocontroller-Board wird über den USB-Port mit einer Spannung von 5 V versorgt, arbeitet intern aber mit 3,3 V und diese Spannung von 3,3 V darf bei Anschaltung von Peripherie nicht überschritten werden.

Empfohlene Anleitung zur Inbetriebnahme:

http://anleitung.joy-it.net/wp-content/uploads/2019/02/SBC-NodeMCU-ESP32-Anleitung_V1.4.pdf

Für die Erfassung der Neigung der Bedieneinheit ist ein Sensorboard auf Basis der Motion Processing Unit MPU 6050 zu verwenden.

https://www.az-delivery.de/products/gy-521-6-achsen-gyroskop-und-beschleunigungssensor

Das Sensorboard ist über den I2C-Bus mit dem Mikrocontroller in Hardware und Software zu verbinden. Das verwendete ESP32-Board nutzt G21 als SDA und G22 als SCL für den Standard-I2C-Bus. Dieser kann damit wie bei einem Arduino über die Bibliothek wire.h genutzt werden:

Tutorial zu I2C für ESP: https://randomnerdtutorials.com/esp32-i2c-communication-arduino-ide/
Beispieldatei zum Scannen des I2C-Busses: I2Cscan.ino

Anwendungsbeispiele mit MPU6050:

Minimalversion: siehe IMU_I2C.ino (Man beachte die Kombination von zwei einzelnen Bytes, die jeweils einen Teil des im Zweierkomplement kodierten Datenworts darstellen.)
https://howtomechatronics.com/tutorials/arduino/how-i2c-communication-works-and-how-to-use-it-with-arduino/

https://www.electronicwings.com/arduino/mpu6050-interfacing-with-arduino-uno

Es bietet sich an, die Library MPU6050_light über die Bibliotheksverwaltung der Arduino Entwicklungsumgebung einzubinden und zu nutzen, so wie mit dem im Git-Repo der Vorlesung mit der Datei MPU6050.ino dargestellt.

5.2.2 Rohdaten

Es ist vor dem Hintergrund der Eignung für die Erfassung der Zustände Ruhe – Transport – Betrieb eine automatische Zustandserkennung auf der Basis der Rohdaten der Sensoren durchzuführen.

5.2.3 Sensorsignalverarbeitung zur Merkmalsextraktion

Es ist der zeitliche Zusammenhang als zusätzliche Information für die Generierung von weiteren Merkmalen (z.B. statistische Merkmale) zu nutzen und bei der Klassifizierung zu berücksichtigen.

5.2.4 Trainingsphase

Sofern nicht bereits im Vorfeld ein Testdatensatz erstellt wurde, ist ein solcher Datensatz zu erstellen und es sind etwa 10.000 Datenpunkte jeweils für einen der Zustände aufzunehmen und als gelabelte Daten zu speichern.

Auf der Grundlage a) dieser Rohdaten und b) der zusätzlichen Merkmale ist ein k-Nearest Neighbour-Klassifikator zu trainieren und die anhand der Accuracy zu bewerten. Zusätzlich ist eine Konfusionsmatrix zu erstellen.

Hinweis:

Zur Auswertung der Daten bietet sich die Nutzung der Bibliothek `pandas` und zur Klassifikation die Nutzung der Bibliothek `sklearn` an.

Bei etwaiger Verwendung von Trainingsdaten anderer Gruppen ist zu berücksichtigen, dass sowohl die Ausrichtung des Sensors als auch die Null-Lage voneinander abweichen und die Daten nicht ohne Normierungsschritte übertragbar sind.

5.2.5 Klassifikationsphase

Es ist ein k-Nearest Neighbour-Klassifikator auf dem PC zu implementieren, der den jeweiligen Betriebszustand der Fernbedienung erfasst und anzeigt.

Die Funktionsfähigkeit ist durch eine fehlerfreie Demonstration aller Betriebszustände über 1 Minute nachzuweisen.

5.3 Inkrement 3: Mikrocontroller-gesteuerte Fernbedienung des Quadcopters

Der Quadcopter soll über einen Joystick gestartet und gelandet und es sollen die Achsen Roll und Pitch über denselben Joystick gesteuert werden können. Anschließend soll eine verbesserte Multisensorsignalverarbeitung implementiert werden.

5.3.1 Ansteuerung des Quadcopters über SDK

Es ist ein Quadcopter vom Typ dji Ryze Tello EDU mit dem SDK 2.0 zu verwenden. Das Software-Development-Kit SDK 2.0 stellt die notwendige Schnittstelle zur Kommunikation mit dem Quadcopter bereit.

Um sich mit der Funktion des Quadcopters vertraut zu machen, empfiehlt sich die Installation der App Tello (nicht TELLO EDU) der Firma RYZE Tech. Dazu ist das Smartphone mit dem WLAN der TELLO-Drohne zu verbinden.

Die Schnittstelle zur Steuerung des Quadcopters wird im Tello SDK 2.0 User Guide beschrieben:
https://dl-cdn.ryzerobotics.com/downloads/Tello/Tello%20SDK%202.0%20User%20Guide.pdf

Zur Einarbeitung in die Schnittstellenfunktion empfiehlt sich die Nutzung des über das HAW git bereitgestellten Python-Skripts Tello3.py:

Voraussetzungen:

- Verbindung des Rechners über WLAN mit Drohne
- Aktivierung der Anaconda Umgebung ST im Anaconda Prompt per `conda activate ST`
- Wechsel in das Verzeichnis mit der Datei Tello3.py per cd
- Start des Skripts per `python Tello3.py`
- Zur Aktivierung der Steuerbefehle ist in der Konsole der Befehl „command“ einzugeben. Bei korrekter Verbindung wird dies durch die grüne LED der Drohne quittiert. Aus Sicherheitsgründen ist der Befehl „command“ nach 15 Sekunden erneut auszuführen.

Die Steuerung eines Quadcopters erfolgt über die vier Achsen:

- Throttle – Schub – Höhe
- Yaw – Gieren – Drehung
- Pitch – Nicken – vor- oder rückwärts
- Roll – Rollen – seitwärts

Hierzu ist sich insbesondere mit der Funktion der folgenden Befehle vertraut zu machen:

- `command`
- `takeoff`
- `land`
- `emergency`
- `rc a b c d` (Hinweis: empfehlenswert sind Werte von a, b, c und d im Bereich von 30. Zu kleine Werte werden ignoriert.)

Das Programm kann per <CTRL>+C beendet werden.

5.3.2 Microcontroller-Board

Es ist das im vorigen Inkrement verwendete Mikrocontroller-Entwicklungs-Board des Typs ESP32 NODE-MCU zu verwenden. Dieser Microcontroller soll die Daten eines Joysticks einlesen und per Tastendruck den Quadcopter starten und landen sowie die Achsen Roll- und Pitch-Steuern.

5.3.3 Remote Control Unit (Fernbedienung) – Phase 1

Es ist ein mechanisch möglichst robuster und optisch ansprechender Aufbau einer Remote Control Unit (Fernbedienung) bestehend aus Microcontroller und Joystick zu realisieren.

Für die RCU ist eine möglichst professionelle Realisierung auf einem Breadboard (Steckbrett) ausreichend.

Die Remote Control Unit soll die folgenden Funktionen bereitstellen:

- Starten des Quadcopters bei Druck auf Joystick
- Landen des Quadcopters bei Druck auf Joystick
- Ansteuerung der Achsen Roll und Pitch über den Joystick unter Nutzung des Befehls `rc` des SDK 2.0
- Sofortiges Stoppen aller Motoren, wenn der rote Notaus-Taster gedrückt wird

Hinweise:

Zur Herstellung einer Verbindung zwischen Microcontroller und Quadcopter bietet sich als Ausgangsbasis das folgende Git-Repo an:

```
git clone https://github.com/akshayvernekar/telloArduino.git
```

Die SDK 2.0-Funktionen `rc` und `land` geben nicht zuverlässig ein OK als Quittung zurück, wodurch der Programmierversuch eines echten Handshakes regelmäßig zu einem Deadlock führt.

Sicherheit

Die Funktionsfähigkeit der Not-Aus-Funktion bei Drücken roten Tasters mit sofortigem Stopp der Motoren ist nachzuweisen.

Um ein unbeabsichtigtes Abheben zu verhindern und das Verletzungsrisiko zu minimieren ist der Quadcopter bei Tests stets mit Gurten o.ä. am Tisch oder Boden gegen Abheben zu sichern. Hierdurch soll insbesondere das Risiko des reflexartigen Ergreifens des Quadcopters mit möglichen schweren Schnittverletzungen vermieden werden.

5.3.4 Remote Control Unit – Stufe 2 – Lagesteuerung mit Multisensordatenverarbeitung

Die Roll- und Nickachse des Quadcopters soll über die Neigung der Fernbedienung gesteuert werden, um eine intuitive Bedienung zu ermöglichen. Schub und Gieren sollen jetzt über die zwei Joystick-Achsen eingestellt werden. Zur Einbindung der Sensoren siehe Inkrement 2, Kapitel 5.2.1.

5.3.4.1 Sensorik

Für die Erfassung der Neigung der Bedieneinheit ist ein Sensorboard auf Basis der Motion Processing Unit MPU 6050 aus Inkrement 2 zu verwenden. Siehe dazu auch die Hinweise in 5.2.1.

Es ist eine Ansteuerung der Längs- und Querneigung des Quadcopters (Roll und Pitch) über die Neigung der Remote Control Unit zu implementieren. Zusätzlich sollen Schub und Drehung (Throttle und Yaw) über den Joystick gesteuert werden.

Eine Neigung der RCU von 45 Grad soll jeweils dem Vollausschlag des Knüppels entsprechen.

Es ist eine möglichst einfache Implementierung zu wählen, in der nur ein Sensortyp für die Erfassung der Lage der RCU verwendet wird. Bei der Auswahl ist zu berücksichtigen, dass bei einer dauerhaft horizontalen Position der RCU die Ruhestellung der Roll- und Pitch-Achse sowie die Höhe des Quadcopters auch dauerhaft beibehalten wird.

5.3.4.2 Signalverarbeitung

Durch die Nutzung von mindestens zwei Messprinzipien je Achse soll die Lage der Remote Control Unit robust erfasst werden können. Insbesondere sollen Beschleunigungen in horizontaler Richtung ohne Veränderung der Lage nicht zu einer Bewegung des Quadcopters führen.

Es ist sicherzustellen, dass bei in horizontaler Lage ruhender Remote Control Unit über 30 Sekunden keine driftbedingte Lageveränderung von mehr als 5° gemessen werden. Dies ist durch entsprechende Messreihen auf dem PC zu visualisieren und zu dokumentieren.

Im Sinne des ingenieurmäßigen Vorgehens sollen vor dem Hintergrund dieser Anforderungen zwei unterschiedliche Ansätze zur Datenfusion realisiert und verglichen werden, nämlich ein komplementäres Filter sowie ein Kalman-Filter unter Nutzung des constant velocity Modells.

Hierzu ist eine Diagnosemöglichkeit zu schaffen, die für einen Zeitraum von 3 Sekunden sowohl die gefilterten als auch die ungefilterten Daten einer beliebigen auszuwählenden Sensorachse als Diagramm visualisiert. Die Zykluszeit des Steuerbefehls an den Quadcopter ist an die Geschwindigkeit des SDK 2.0 anzupassen und zu dokumentieren.

Für den Vergleich zwischen komplementärem Filter und Kalman-Filter sind beide Filter auf dem µC unter Nutzung verfügbarer Bibliotheken zu implementieren. Dabei sind insbesondere das Verhalten bei einer alleinigen Änderung des Nickwinkels der Fernbedienung von 0 auf 45° und bei einer alleinigen Beschleunigung in x-Richtung ohne Winkeländerung zu optimieren und anhand von Diagrammen zu vergleichen.

Hinweis:
Machen Sie sich zur Visualisierung von Daten unter Python mit der Matlab-ähnlichen Bibliothek matplotlib vertraut.

Sicherheit

Die Funktionsfähigkeit der Not-Aus-Funktion bei Drücken roten Tasters mit sofortigem Stopp der Motoren ist nachzuweisen.

Um ein unbeabsichtigtes Abheben zu verhindern und das Verletzungsrisiko zu minimieren ist der Quadcopter bei Tests stets mit Gurten o.ä. am Tisch oder Boden gegen Abheben zu sichern. Hierdurch soll insbesondere das Risiko des reflexartigen Ergreifens des Quadcopters mit möglichen schweren Schnittverletzungen vermieden werden.

5.4 Inkrement 4: Smart AirCam

Das Ziel des 4. Inkrements ist die Programmierung eines Quadrocopters als Foto-Drohne, die automatisch startet und dann auf der Stelle solange dreht, bis diese ein Gesicht erkannt hat und dann ein Foto des Gesichtes macht.

5.4.1 Hardware

Da die Bildverarbeitung rechenintensiv ist, erfolgt diese in einem PC und der Quadcopter wird anders als zuvor ausschließlich durch den PC gesteuert.

Zur Ansteuerung der Drohne muss der PC in das WLAN des Quadcopters eingewählt sein. Anschließend kann dieser zu Testzwecken mit dem Programm Tello3.py aus dem git-Repo angesteuert werden, wie in Kapitel 5.3.1 erläutert.

Als Sensorik wird die eingebaute Kamera des Quadcopters genutzt, so dass keine weitere Hardware neben dem Quadcopter sowie ggfs. einem WLAN-Stick erforderlich ist.

5.4.2 Bilderfassung

Der Quadrocopter kann durch den Befehl `streamon` bzw. `streamoff` einen Videostream der Kamerabilder über den UDP-Port 11111 starten oder stoppen.

Analysieren Sie das Testprogramm `opencv_tello.py` zeilenweise und machen Sie sich den Ablauf und die Funktion jedes einzelnen Befehls klar und testen Sie die Videoübertragung.

Messen Sie den zeitlichen Verzug zwischen Bildaufnahme und Anzeige auf dem Rechner und versuchen Sie, diesen zu minimieren.

5.4.3 Bildverarbeitung

Entwickeln Sie ein Programm unter Nutzung von Haar-Cascades zur Detektion von Gesichtern. Hierzu sei das Tutorial https://pyimagesearch.com/2021/04/12/opencv-haar-cascades/ empfohlen.

Entwickeln Sie eine Ablaufsteuerung, die

1. den Quadcopter startet,
2. auf eine Höhe von 1,50 m steigen lässt,
3. den Videostream anzeigt,
4. Gesichter detektiert und diese durch eine Bounding Box im Live-Stream anzeigt,
5. den Quadcopter so lange auf der Stelle drehen lässt, bis das erste Gesicht gefunden wurde,
6. so lange dreht, bis genau dieses Gesicht mittig positioniert ist und dort 3 Sekunden verharrt,
7. im Abstand von jeweils 2 Sekunden zwei Fotos ohne Bounding Box aufnimmt und speichert
8. und anschließend den Quadcopter landet.

Der Quadcopter soll dabei in jedem Zustand durch Drücken der Tastenkombination <CTRL>+C einen Emergency Stop ausführen. Machen Sie sich dazu mit dem Exception-Handling und speziell dem Konstrukt try-except KeyboardInterrupt vertraut.

5.4.4 Dokumentation

Das 4. Inkrement ist durch eine sehr gut kommentierte graphische Darstellung des Programmablaufs und die erfolgreiche Aufnahme des Teams zu dokumentieren.

5.5 Materialrückgabe

Die Prüfungsvorleistung ist erst dann erfüllt, wenn das ausgegebene Material nach erfolgreichem Test im letzten Laborversuch wieder vollständig zurückgegeben wurde.

Dazu bitte die folgenden Hinweise beachten, damit die Teams nach Ihnen problemlos starten können:

- Akkus in Ladezustand 60 … 80 % bringen
- Akkus aus dem Quadcopter entnehmen
- Quadcopter, Akkus, Ladegerät, Ersatzpropeller sorgfältig in Karton verpacken (SO DASS ES PASST!)
- Kabel durch Gummiband oder Kabelbinder zusammenbinden
- Bauelemente auf ESD-Schaumstoff (falls fehlt, auf das Breadboard) stecken
- Material einschließlich USB-Kabel, Sicherheitsbrille, … auf Vollständigkeit mit Materialliste S. 2 prüfen
- Zettel mit Ihrem Namen erstellen
- Material in HAW-Beutel mit dem Namenszettel abgeben

Vielen Dank für eine sorgfältig durchgeführte Rückgabe.