HOCHSCHULE OSNABRÜCK

UNIVERSITY OF APPLIED SCIENCES

Praktikum im Fach

Internet of Things (IoT) / Industrie 4.0

Versuch 3: Road Condition Monitoring

Autoren: Marco Schaarschmidt (m.schaarschmidt@hs-osnabrueck.de)
Nicolas Lampe (n.lampe@hs-osnabrueck.de)

Version: 2.1

Wichtige Hinweise:

- Zu dieser Aufgabenstellung gehört ein Dokument mit Basisinformation.
- Bei Fragen oder Rückmeldungen zum Praktikumsversuch stellen Sie diese bitte primär zu den Praktikumszeiten (dienstags 14:30 bis 16:00). Bei dringenden Fällen schreiben Sie eine E-Mail an oder Nicolas Lampe (n.lampe@hs-osnabrueck.de) oder Sebastian Böhm (sebastian.boehm@hs-osnabrueck.de).

1 Einleitung

Für die Steigerung des Komforts und der Fahrsicherheit ist vor allem in der Automobilindustrie ein aktuelles Forschungsthema das Road Condition Monitoring. Die Ziele sind das Erkennen von Straßenanomalien wie beispielsweise Schlaglöcher und das Ermitteln verschiedenen Straßenoberflächen [GEE16]. Durch die Kenntnis von potenziellen Gefahrenstellen können diese ausgebessert werden. Dazu gibt es Messfahrzeuge wie den Road Friction Tester [BL17]. Messfahrten mit diesem Fahrzeug sind hingegen immer mit einem hohen Kostenaufwand verbunden. Ziel der Forschung ist es dementsprechend eine kostengünstige Alternative zu den bestehenden Messfahrzeugen zu schaffen. Ein Ansatz ist die Verwendung der im Smartphone enthaltenen Beschleunigungssensorik und in Kombination mit den GPS (Global Positioning Systems)-Daten, sodass cloudbasiert die Positionsdaten mit einem zugehörigen Kennwert über die Straße gesammelt werden können [MBJ18]. Weiterhin gibt es Ansätze bei denen anhand der gesammelten Daten Echtzeitkarten erstellt werden, in denen die ermittelten Straßenanomalien gekennzeichnet sind [GEE16].

In dem folgenden Versuch wird ebenfalls das Fahrrad mit in die Ermittlung von Straßenanomalien einbezogen. Dazu soll der Beschleunigungssensor des M5Stacks in Kombination mit dem GPS-Modul verwendet werden. Für diesen Versuch werden aus dem IoT/I4.0-Box die folgenden Hardwarekomponenten benötigt:

- M5Stack Core
- NEO-M8N GPS Modul

Machen Sie sich vor Beginn des Praktikums mit der Funktionsweise des GPS Moduls sowie die im Praktikum eingesetzten Arduino-Libraries vertraut. Generell sollten Sie vor Beginn dieses Versuches die Basisinformationen gelesen und ebenfalls verstanden haben, damit Sie die Grundlagen zur Bearbeitung des Versuches besitzen.

2 Versuchsdurchführung

Im Folgenden soll vorerst das GPS-Modul und im Anschluss der GPS-Sensor in Betrieb genommen werden. Die Daten der jeweiligen Sensoren sollen auf dem Bildschirm angezeigt und ebenfalls an einen InfluxDB-Server geschickt werden, sodass die Daten gespeichert, angezeigt und ausgewertet werden können. Die Analyse der Daten ist nicht Teil des Versuches und soll nur exemplarisch betrachtet werden. Als Versuchsprotokoll sind die nachfolgenden Fragen in schriftlicher Form zu beantworten und mit dem entwickelten Programmcode in OSCA vor dem Testat zu hinterlegen. Zusätzlich muss Ihr Versuchsprotokoll eine Beschreibung sowie Screenshots der Graphen und den von Ihnen genutzten Einstellungen aus der InfluxDB beinhalten.

- Erklären Sie kurz die Funktionsweise eines GPS-Sensors. Welche Werte kann dieser ermitteln?
- Hier ist ein beispielhafter Rohdatensatz eines GPS-Sensors dargestellt:
  GPGGA,110113.4,3536.055,N,13872.722,E,1,12,0.85,03170,M,051,M,,*57<CR><LF
  Welche Bedeutung haben die einzelnen Bestandteile? An welchem Ort und in welchem Land wurde dieses GPS Signal aufgenommen?
- Was gibt der HDOP-Wert an und was sind Gründe für einen hohen HDOP?
- Welche Größen misst der MPU9250 und welche Einheiten haben diese?
- Wie kommuniziert der MPU9250 mit dem ESP32 des M5Stack Core Moduls? Beschreiben Sie kurz die Funktionsweise der Schnittstelle.
- Um was für einen Datenbanktyp handelt es sich bei der InfluxDB?
- Wofür stehen die Bezeichnungen *bucket* und *measurement* im Kontext der InfluxDB?
- Auf welche Schwierigkeiten sind Sie bei der Versuchsdurchführung gestoßen?

2.1 Inbetriebnahme des GPS-Moduls

In einem ersten Schritt soll das GPS-Modul angeschlossen werden. Dazu muss der M5Stack mittels eines Inbusschlüssels aufgeschraubt werden. Das GPS Modul kann nun auf das M5Core Modul passend aufgesetzt werden. Die beiden Schrauben sollten zur Befestigung leicht festgezogen werden. Anschließend kann die GPS Antenne angeschlossen werden. Damit das GPS-Modul die GPS Antenne verwendet, muss das schwarze Kabel auf der Unterseite des Moduls angeschlossen werden anstatt das graue (dies ist für die interne Antenne). Lassen Sie sich

- die Anzahl der Satelliten,
- den HDOP-Wert,
- den Breiten- (Latitude) und Längengrad (Longitude),
- sowie das Datum und die Uhrzeit

auf dem Bildschirm des M5Stacks anzeigen.

2.2 Inbetriebnahme des Beschleunigungssensors

Erweitern Sie den Sketch aus Abschnitt 2.1 durch die Verwendung des Beschleunigungssensors. Lesen Sie die Beschleunigungsdaten in x-, y- und z-Richtung aus und zeigen Sie diese – ebenfalls wie die GPS-Daten – auf dem Bildschirm an.

2.3 Erweiterung der Anzeige

Im Folgenden soll die Anzeige auf dem Bildschirm des M5Stacks erweitert werden. Um zur aktuellen Position genauere Werte und erste Auswertungen zu liefern, soll auf dem Bildschirm die Distanz zu einem Startpunkt und die Zeit seitdem Festlegen des Startpunkts angezeigt werden. Binden Sie Button A dazu so ein, dass beim Drücken von Button A die aktuelle Position als Startposition und die aktuelle Zeit als Startzeit gespeichert wird. Berechnen Sie die Distanz der aktuellen Position zu der Startposition, sowie die vergangene Zeit zur Startzeit und geben Sie diese ebenfalls auf dem Display aus. Die Anzeige auf dem Bildschirm könnte beispielsweise aussehen wie in Abbildung 1 dargestellt.

Abbildung 1: Mögliches Ergebnis für die Anzeige auf dem M5Stack

2.4 InfluxDB

Für die Interaktion mit der InfluxDB muss der Arduino IDE zunächst eine weitere Bibliothek hinzugefügt werden. Unter „Bibliothek verwalten“ und unter der Verwendung des Suchbegriffes „influxdb“ sollte Ihnen wahlweise die Bibliothek *ESP8266 Influxdb* oder *InfluxDBClient*

for Arduino vorgeschlagen werden, die Sie in der Version 3.8.0[1] installieren. Für die Nutzung der InfluxDB werden die folgenden Parameter benötigt:

- URL: `https://eu-central-1-1.aws.cloud2.influxdata.com`
- Organisationseinheit: `hsos`
- Bucket: `iot-bucket`

Zur Authentifizierung mit der Datenbank wird zudem ein Token benötigt, der wie folgt lautet:

```
FESIyzQ8punR1imo7K6NvofJQ8q5Gy6fDOWzC8UThTNYCoGHCHKxkGNplFT1vXgAF-
HAK9a1-9f5_7A23rhnO4g==
```

Wenn Sie Datenwerte in die Datenbank schreiben möchten, muss von Ihnen im Vorfeld ein `measurement` definiert werden. Achten Sie bei der Benennung darauf, einen eindeutigen Namen zu verwenden. Dieser kann wahlweise Ihre Gruppennummer oder Abkürzungen der Nachnamen der einzelnen Gruppenteilnehmer beinhalten. Für die Anzeige und Auswertung Ihrer aufgezeichneten Datenpunkte rufen Sie die oben genannte URL mit Ihrem Browser auf. Es sollte der in Abbildung 2 dargestellte Inhalt angezeigt werden.

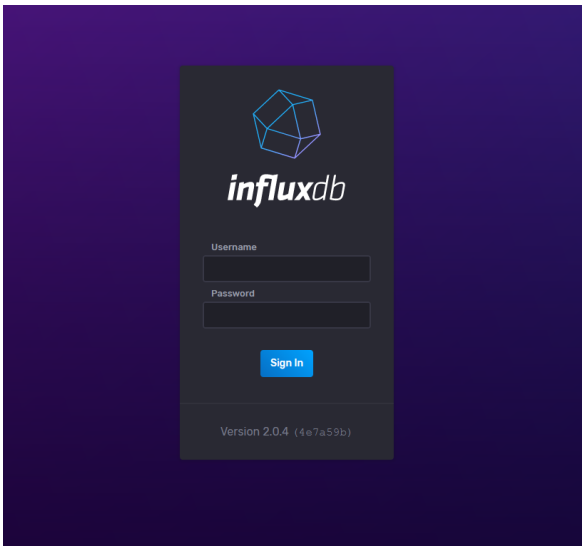

Abbildung 2: Anmeldefenster der InfluxDB

Für die Anmeldung in der InfluxDB wurde bereits ein Account erstellt. Mit der E-Mail Adresse *PraktikumDummy@gmail.com* und dem Passwort *FJG-GKCb!GqKzn3z#5LK8e~m* können Sie sich anmelden. Anschließend klicken Sie auf *Explore* in der linkseitigen Menüleiste. Sie sollten nun den *Data Explorer* wie in Abbildung 3 sehen. Nutzen Sie bitte nicht die weiteren Reiter, da Sie sonst versehentlich unerwünschte Änderungen vornehmen können.

[1] Befehlsübersicht unter: https://github.com/tobiasschuerg/InfluxDB-Client-for-Arduino

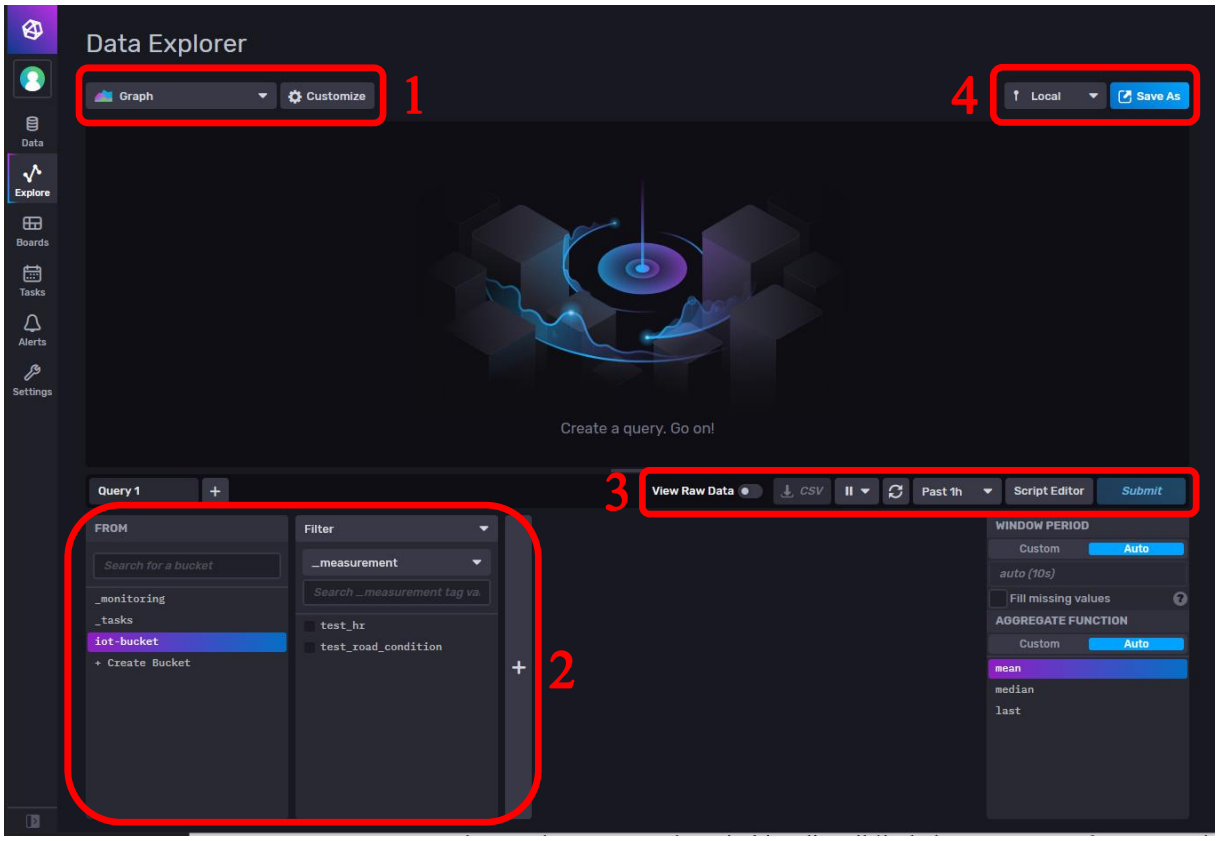

Abbildung 3: Data Explorer der InfluxDB

Zunächst können Sie die Art des Graphen (1 in Abbildung 3) auswählen. Um die Datenquelle zu spezifizieren, müssen Sie im angegebenen *Bucket* ihr zuvor definiertes *Measurement* auswählen (2). Was der Graph anzeigen soll, können Sie unter (3) genauer spezifizieren, in dem Sie entsprechende Filter setzen. Sollte Ihr *Measurement* nicht vorhanden sein, ist das Programm oder die Kommunikation des M5Stack mit der InfluxDB fehlerhaft. Haben Sie ihr *Measurement* entsprechend ausgewählt, sollten bereits erste Daten angezeigt werden. Den zu berücksichtigen Zeitraum sowie eine Auto-Update-Funktion können Sie unter (3) konfigurieren. Um die erstellte Auswertung zu einem späteren Zeitpunkt wieder aufrufen zu können, kann der Graph unter (4) gespeichert werden. Berücksichtigen Sie, dass Ihre aufgezeichneten Daten nach Ablauf einer eingestellten Frist (aktuell 7 Tage) gelöscht werden. Erstellen Sie also direkt die Screenshots für die Auswertung.

Hinweise zu der Aufgabe:

- Die Baudrate des GPS Sensors beträgt 9600 und muss bei der Initialisierung der Verbindung mit angegeben werden. Dies gilt ebenfalls für die Pins der UART-Schnittstelle.
- Für die Auswertung der Rohdaten des GPS-Sensors können verschiedene Bibliotheken verwendet werden. Verwenden Sie hier die Bibliothek *TinyGPS++.h*. Diese stellen wir Ihnen im OSCA bereit. Hier ist ein Auszug der zugehörigen Befehle dargestellt:

`gps.encode(…)`	Enkodieren eines Roh-GPS-Signals
`gps.location.lat()`	Längengrad in Grad
`gps.location.lng()`	Breitengrad in Grad
`gps.location.isvalid()`	Ausgabe über eine gültige GPS-Position
`gps.location.rawLat().deg`	Rohlängengrad in Grad
`gps.location.rawLng().deg`	Rohbreitengrad in Grad
`gps.date.value()`	Rohdaten des Datums im Format DDMMYY (u32)
`gps.date.year()`	Jahr (2000+) (u8)
`gps.date.month()`	Monat (1-12) (u8)
`gps.date.day()`	Tag (1-31) (u8)
`gps.time.value()`	Rohdaten der Zeit im Format HHMMSSCC (u32)
`gps.time.hour()`	Stunden (0-23) (u8)
`gps.time.minute()`	Minuten (0-59) (u8)
`gps.time.second()`	Sekunden (0-59) (u8)
`gps.time.centisecond()`	Centisekunden (0-99) (u8)
`gps.speed.value()`	Rohdaten der Geschwindigkeit (i32)
`gps.speed.knots()`	Geschwindigkeit in Knoten (double)
`gps.speed.mph()`	Geschwindigkeit in Meilen pro Stunde (double)
`gps.speed.mps()`	Geschwindigkeit in Metern pro Sekunde (double)
`gps.speed.kmph()`	Geschwindigkeit in Kilometern pro Stunde (double)
`gps.course.value()`	Rohdaten des Kurses in 100.-tel eines Grads (i32)
`gps.course.deg()`	Kurs in Grad (double)
`gps.altitude.value()`	Rohdaten der Höhe in Centimetern (i32)
`gps.altitude.meters()`	Höhe in Metern (double)
`gps.altitude.miles()`	Höhe in Meilen (double)
`gps.altitude.kilometers()`	Höhe in Kilometern (double)
`gps.altitude.feet()`	Höhe in Fuß (double)
`gps.satellites.value()`	Anzahl der Satelliten (u32)
`gps.hdop.value()`	Angabe des HDOP-Wertes (i32)

- Mit folgender Funktion kann die Distanz zwischen zwei GPS-Positionen bestimmt werden:

```
(unsigned long)TinyGPSPlus::distanceBetween(
      Längengrad Position A,
      Breitengrad Position A,
      Längengrad Position B,
      Breitengrad Position B);
```

- Nutzen Sie für die Verwendung des Beschleunigungssensors die *MPU9250.h* Bibliothek. Diese ist bereits im bereitgestellten zip-Ordner *M5Stack.zip* unter *src/utility* integriert.
- Für die Nutzung des MPU9250 müssen die Pins der I2C-Verbindung (SDA, SCL) korrekt definiert werden. Diese können aus den Basisinformationen entnommen werden.
- Für die Verwendung der InfluxDB binden Sie bitte die *InfluxDbCloud.h* Bibliothek ein.

3 Quellenverzeichnis

Alle Internetquellen wurden zuletzt am 08.03.2022 abgerufen.

BL17 Braun, Volker; Lellmann, Christian: *Verfahren und Vorrichtung zum Detektieren von Anomalien in Signaldaten für eine Reibwertschätzung für ein Fahrzeug.* Robert Bosch GmbH, Deutschland. 29.05.2019. Veröffentlichungsnr. DE102017221050A1

GEE16 Gawad, Shahd Mohamed Abdel (Hrsg.); El Mougy, Amr (Hrsg.); El-Meligy, Menna Ahmed (Hrsg.): Dynamic Mapping of Road Conditions Using Smartphone Sensors and Machine Learning Techniques, 2016

MBJ18 Mohamed Akram Ameddah (Hrsg.); Bhaskar Das (Hrsg.); Jalal Almhana (Hrsg.): Cloud-Assisted Real-Time Road Condition Monitoring System for Vehicles. Piscataway, NJ: IEEE, 2018