Testiranje senzorja za zračni tlak glede na spremembo višine

Seminarska naloga pri predmetu Brezžična senzorska omrežja

Blaž Marolt, 63140158
Rok Šolar, 63140363

12. maj 2019

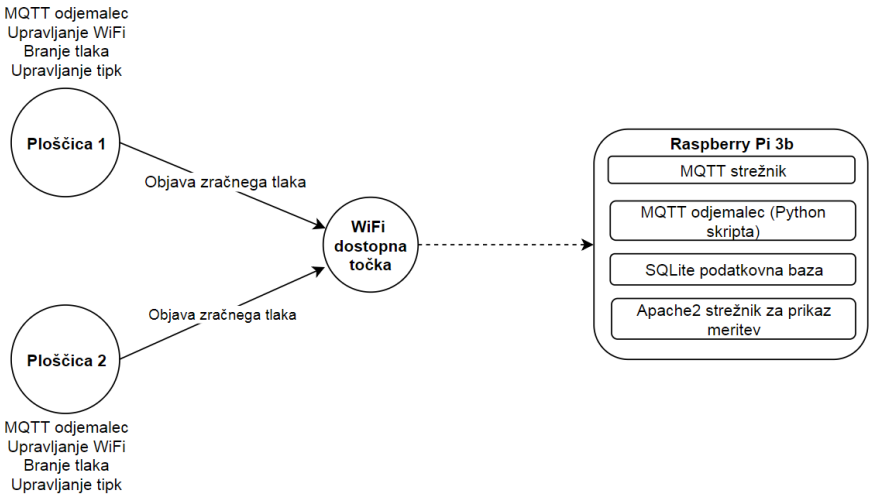
1 Povzetek

Brezžična senzorska omrežja (ang. Wireless Sensor Networks - WSN) se pojavljajo na mnogih področjih vsakodnevnega življenja, kar je očitno še posebej v zadnjih letih, z razvojem paradigme internet stvari (ang. Internet of Things - IoT) [4]. V tem delu bomo testirali senzor za zračni tlak BMP280, vgrajen na senzorskem modulu GY-91, ter ob tem postavili brezžično senzorsko omrežje za prenos pridobljenih meritev na strežnik, z uporabo protokola MQTT. V nadaljevanju bomo podrobneje opisali problem in cilje, opisali arhitekturo omrežja, implementacijo rešitve, eksperimente ter rezultate.

2 Opis problema in cilji

Za implementacijo rešitve smo imeli na voljo razvojno ploščico z modulom WEMOS D1 mini, ki vsebuje mikrokrmilnik ESP-8266EX, anteno za WiFi ter pretvornik USB v serijsko povezavo. Poleg tega ploščica vsebuje razne periferne naprave, med njimi tudi senzorski modul GY-91, ki vsebuje žiroskop, pospeškometer in kompas na modulu MPU-9250 ter termometer in barometer na modulu BMP280. Oba modula sta povezana z mikrokrmilnikom preko vodila I2C, vsak na svojem naslovu.
Cilj naloge je bil testirati senzor za zračni tlak (barometer), vgrajen na modulu BMP280. Definirali smo nekaj eksperimentov, pri katerih smo primerjali delovanje senzorja na dveh identičnih ploščicah.
Določili smo arhitekturo, prikazano na sliki 1. Imamo torej dve ploščici, ki sta z uporabo WiFi antene na modulu povezani z dostopno točko. Na obeh ploščicah je naložena skoraj identična programska oprema, ki skrbi za WiFi, branje tlaka, pošiljanje tlaka na strežnik z uporabo protokola MQTT ter upravljanje tipk, katerih funkcijo bomo natančneje opisali v poglavju 3. Na neki drugi, oddaljeni lokaciji pa teče računalnik Raspberry Pi 3b, katerega primarni namen je omogočati strežnik MQTT, ki sprejema meritve ploščic. Poleg tega teče krati še

Python skripta, ki deluje kot MQTT odjemalec. Torej bere meritve in jih zapisuje v podatkovno bazo SQLite, ki je nameščena na istem računalniku. Ponuja še spletno stran, ki omogoča grafični prikaz meritev za izbrano obdobje v obliki črtnega diagrama.



Slika 1: Arhitektura omrežja.

3 Metodologija

3.1 WEMOS D1 mini

V nadaljevanju bomo predstavili programsko kodo, ki teče na modulih WEMOS D1 mini. Koda je napisana v programskem jeziku C in strukturirana na več funkcij. Pomembnejše funkcije so:

- `i2c_init_bmp()` inicializira I2C vodilo preko katerega komunicira z modulom BMP280. V primeru uspešne inicializacije vrne vrednost true, drugače pa false.
- `init_bmp()` nastavi potrebne parametre in inicializira delovanje modula BMP280. Funkcija vrne true, če se modul uspešno inicializira.
- `get_pressure()` iz modula prebere vrednosti zračnega tlaka, temperature in vlažnosti ter vrne vrednost tlaka.
- `beat_task()` periodično opravilo, ki se izvaja vsako sekundo. Funkcija vsako sekundo prebere vrednost tlaka, naredi povprečje zadnjih 10 meritev ter doda izračunano vrednost v publish_queue čakalno vrsto.

- `mqtt_task()` je funkcija, ki pridobi vrednost iz čakalne vrste ter jo preko protokola MQTT pošlje na strežnik.
- `wifi_task()` je funkcija, ki pregleda prisotna brezžična omrežja ter se poveže na omrežje, ki je definirano v datoteki private_ssid_config.h.
- `button()` je funkcija, ki preverja ali kateri izmed definiranih gumbov pritisnjen. Ob pritisku na gumb 1 se na napravi zamenja način delovanja, kar bomo opisali v nadaljevanju. Ob pritisku na gumb 2 sprožimo ročno meritev.
- `user_init()` je začetna funkcija, ki inicializira vsa opravila.

Pomembne definicije spremenljivk:

- MQTT_HOST MQTT: naslov strežnika na katerega pošiljamo vrednost tlaka.
- MQTT_PORT MQTT: vrata protokola MQTT, nastavljena na 1883.
- MQTT_USER: uporabniško ime MQTT odjemalca.
- MQTT_PASSWORD: geslo MQTT odjemalca.
- WIFI_SSID: SSID brezžičnega omrežja, na katerega se želimo povezati.
- WIFI_PASS: Geslo brezžičnega omrežja.

Opis načinov delovanja:

- Periodični način: pri periodičnem načinu delovanja opravilo `beat_task()` izvede meritev vsako sekundo in pošlje povprečno vrednost meritev vsakih 10 sekund.
- Ročni način: s pritiskom na gumb 1 kličemo funkcijo `vTaskSuspend`, ki ustavi delovanje opravila `beat_task()`. S pritiski na gumb 2 lahko zdaj takoj izvedemo meritev, ki se doda v publish_queue čakalno vrsto in nato pošlje na strežnik. S ponovnim pritiskom na gumb 1 sprožimo funkcijo `vTaskResume`, ki ponovno zažene opravilo `beat_task()`.

Razvojna ploščica takoj po priključitvi na napajanje začne iskati brezžično omrežje določeno v datoteki private_ssid_config.h. Ko se uspešno poveže v omrežje začne privzeto delovati v periodičnem načinu. Ploščice tako ni potrebno ponovno konfigurirati ob naslednjem zagonu ampak le takrat, ko jo prestavimo v drugo brezžično omrežje.

3.2 Raspberry Pi

MQTT strežnik in zapisovanje v podatkovno bazo smo implementirali na mikroračunalniku Raspberry Pi, model 3B. Raspberry se je nahajal na oddaljeni lokaciji zato smo na usmerjevalniku ki skrbi za lokalno omrežje v katerem je Raspberry nastavili posredovanje vrat. Vrata, ki jih uporablja MQTT (v našem primeru 1883) smo preusmerili na lokalni IP naslov, ki ga je dobil Raspberry. Za dostop do mikroračunalnika smo uporabili programsko opremo VNC, ki nam omogoča povezavo z oddaljenim namizjem. V nadaljevanju bomo opisali aplikacije ki tečejo na mikroračunalniku Raspberry.

3.2.1 MQTT strežnik

Kljub temu da že obstaja več brezplačnih MQTT strežnikov, ki nam z uporabo uporabniškega imena in gesla sicer omogočajo osnovno raven varnosti, smo želeli postaviti svoj MQTT strežnik. Uporabili smo Mosquitto, ki nam omogoča enostavno namestitev MQTT strežnika. V nadaljevanju bomo predstavili namestitev strežnika. Mosquitto smo namestili z naslednjimi ukazi:

```
sudo apt-get update
sudo apt-get install mosquitto mosquitto-clients
```

Po namestitvi je strežnik potrebno še pognati z ukazom:

```
mosquitto -d
```

Ukaz zažene Mosquitto strežnik kot demon, tako da program deluje v ozadju.

3.2.2 Uporaba MQTT uporabniškega imena in gesla

Za večjo varnost smo dodali preverjanje gesla in uporabniškega imena pri uporabi protokola MQTT. Na strani strežnika smo na Raspberry Pi ustvarili novo datoteko pass.txt v mapi */etc/mosquitto*. Datoteka pass.txt je oblike ime:geslo. Datoteko je potrebno šifirati, zakar smo uporabili naslednji ukaz.

```
mosquitto_passwd -U pass.txt
```

V datoteki *mosquito.conf* smo spremenili parameter allow anonymous na false ter nastavili pot password_file na ustvarjeno datoteko pass.txt. Nastavitev uporabe gesla na strani odjemalca je bolj preprosta. V kodi, ki teče na razvojnih ploščicah smo spremenili definirani konstatni MQTT_USER in MQTT_PASSWORD, v Python kodi pa smo poklicali metodo client.username_pw_set(ime, geslo).

3.2.3 Python MQTT odjemalec in podatkovna baza

Zaradi nizke kompleksnosti našega sistema ter relativno nizke zmogljivosti računalnika smo se odločili za podatkovno bazo SQLite.
Ukazi z katerimi smo namestili podatkovno bazo SQLite:

```
sudo add-apt-repository ppa:jonathonf/backports
sudo apt-get update && sudo apt-get install sqlite3
```

Shema tabele, ki smo jo uporabili za hranjenje vrednosti tlaka:

```
CREATE TABLE readings(id INTEGER PRIMARY KEY AUTOINCREMENT,
pressure NUMERIC,currentdate DATE,currentime TIME, device TEXT);
```

Python skripta za MQTT odjemalec uporablja knjižnico `paho-mqtt` [2]. Skripta se poveže na strežnik (v našem primeru gre seveda za localhost povezavo, saj strežnik in odjemalec tečeta na istem računalniku), prijavi na ustrezne teme ter čaka na dobljena sporočila. Ob prejetju se poveže na podatkovno bazo SQLite ter zapiše dobljeno vrednost, časovni žig ter ID naprave. Pri tem smo prišli do težave z imenom odjemalca. V kolikor skripto ustavimo in na novo zaženemo ta ne deluje, če ne spremenimo imena odjemalca. Predvidevamo, da se povezava ne zaključi ustrezno in se zato naslednja povezava z istim imenom ne inicializira uspešno, vendar te težave nismo uspeli rešiti.

3.2.4 Spletni strežnik

Spletni strežnik omogoča prikaz spletne strani za grafični prikaz meritev za določeno obdobje. Na Raspberry Pi smo namestili spletni strežnik Apache2 ter podporo za programski jezik PHP (pri tem so nam bila v pomoč navodila [3]), katerega uporabljamo za branje podatkov iz podatkovne baze ter dinamični prikaz. Spletna stran je sestavljena iz enega PHP dokumenta in omogoča izbiri začetnega in končnega datuma in ure. Po ustrezni izbiri prebere podatke iz podatkovne baze ter osveži stran. Za prikaz črtnega diagrama smo uporabili knjižnico `CanvasJS` [1].

3.3 Eksperimenti

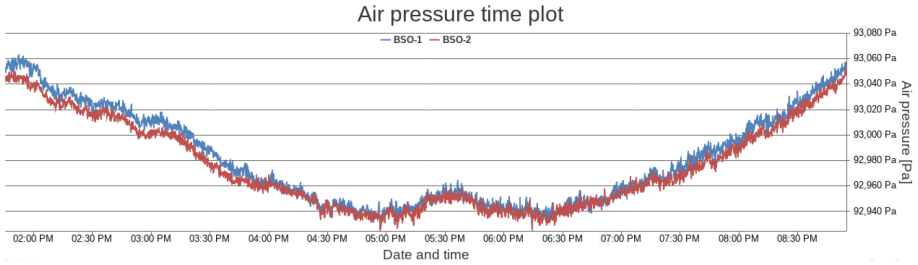
Definirali smo štiri eksperimenti, katere bomo v nadaljevanju opisali ter prikazali rezultate.

3.3.1 Stacionarna meritev, brez povprečenja (nadmorska višina 700m)

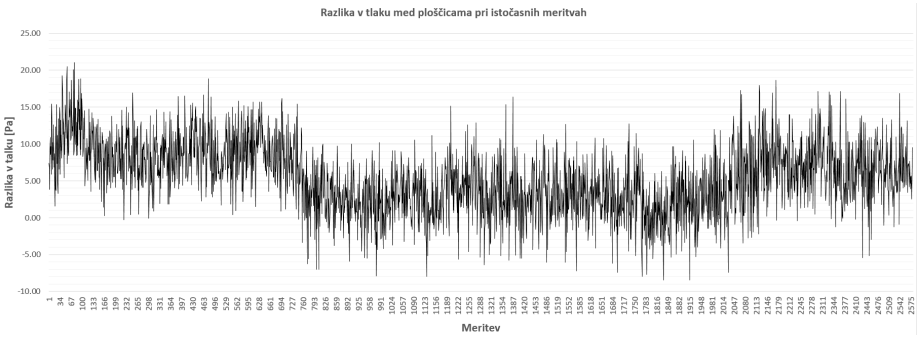
Pri prvem eksperimentu smo ploščici postavili na neko nepremično točko, na nadmorski višini približno 700m. Pri tem so se meritve izvajale vsakih 10 sekund za približno 7 ur. Na MQTT strežnik smo pošiljali surove meritve, torej brez povprečenja večih meritev. Na sliki 2 vidimo pridobljen graf meritev za opisano časovno obdobje, pridobljen iz spletne aplikacije. Na prvi pogled vidimo, da se meritve med ploščicama pri višjem tlaku bolj razlikujejo kot meritve pri nižjem tlaku, kar potrjuje tudi slika 3, kjer so prikazane razlike v meritvah med ploščicama skozi isto časovno obdobje. Tabela 1 prikazuje še statistične podatke meritev, iz katerih lahko razberemo, da v povprečju ena izmed ploščic običajno prebere višji tlak kot druga (statistični podatki so izračunani na absolutnih vrednostih razlik meritev). Opazimo tudi nihanje zračnega tlaka zaradi vremenskih sprememb.

Povprečno odstopanje	12,74
Največje odstopanje	21,09
Standardni odklon	4,01

Tabela 1: Eksperiment 1: statistični podatki



Slika 2: Eksperiment 1: meritve ploščic.



Slika 3: Eksperiment 1: razlike meritev.

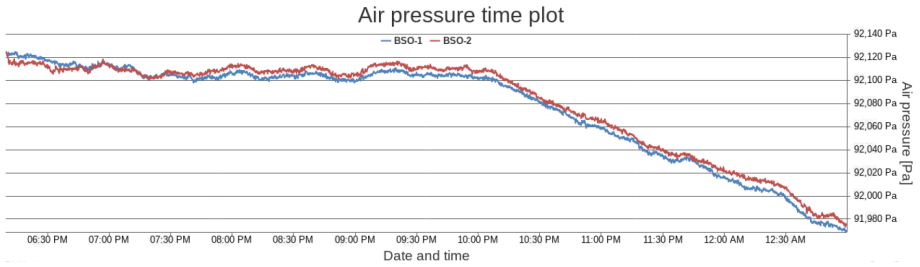
3.3.2 Stacionarna meritev, s povprečenjem (nadmorska višina 700m)

Drug eksperiment je identičen prvemu. Meritve smo še vedno pošiljali na 10 sekund, le da smo v teh desetih sekundah zbrali 10 meritev, jih povprečili in dobljeno vrednost poslali na strežnik. Meritve so prikazane na sliki 4, razlike med meritvami pa na sliki 5. Ugotovitve iz prvega eksperimenta tukaj ne držijo, saj se razlike meritev med ploščicama ne večajo/manjšajo z razliko v zračnem tlaku. Na začetku nekaj meritev prva ploščica vrača višje odčitke, potem se pa zamenjata (prehod iz pozitivnih na negativne razlike) in tako ostaneta do

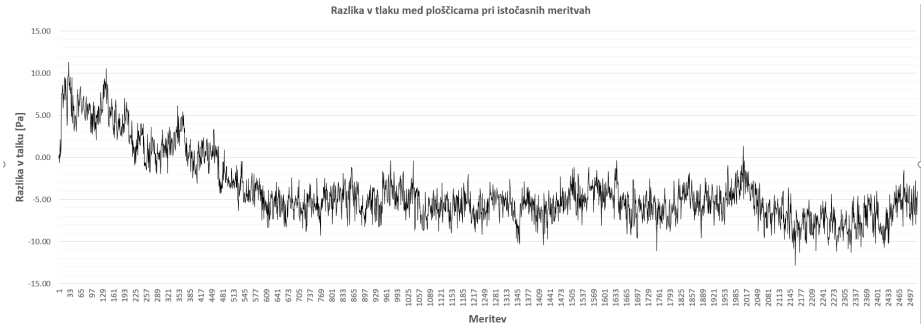
Povprečno odstopanje	5,2
Največje odstopanje	12,84
Standardni odklon	2,23

Tabela 2: Eksperiment 2: statistični podatki

konca eksperimenta, kljub padcu tlaka. Statistični podatki v tabeli 2 prikazujejo manjše vrednosti kot pri prvem eksperimentu, kar pripisujemo povprečenju meritev, saj s tem dobimo točnejše meritve.



Slika 4: Eksperiment 2: meritve ploščic.



Slika 5: Eksperiment 2: razlike meritev.

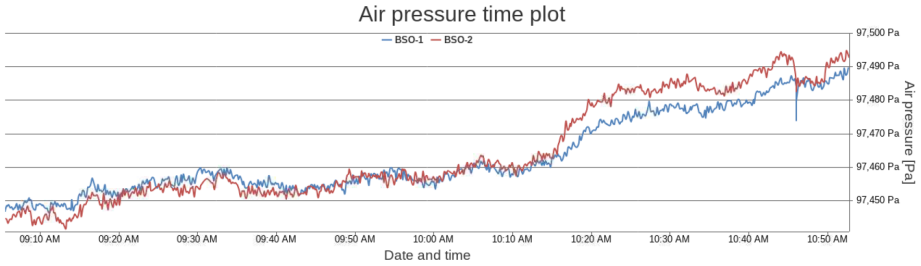
3.3.3 Stacionarna meritev, s povprečenjem (nadmorska višina 340m)

Pri tretjem eksperimentu smo pridobivali stacionarne meritve na občutno nižji nadmorski višini (340m). Zaradi spremembe nadmorske višine seveda opazimo

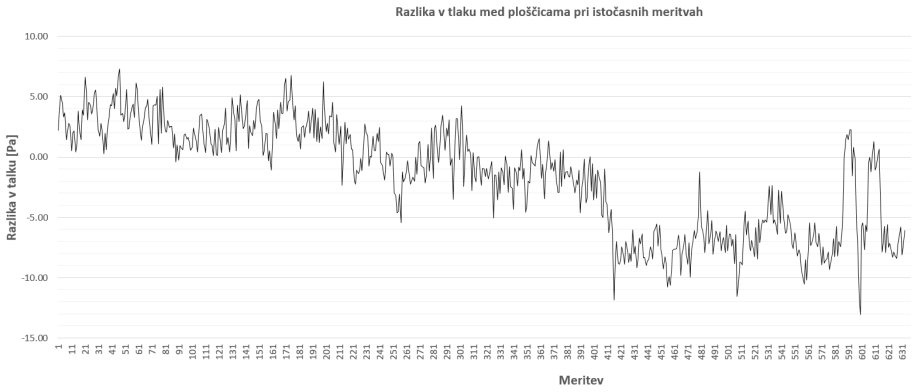
Povprečno odstopanje	3,79
Največje odstopanje	13,07
Standardni odklon	2,82

Tabela 3: Eksperiment 3: statistični podatki

spremembo v zračnem tlaku (nižja nadmorska višina = višji zračni tlak). Iz slik 6 in 7 opazimo ponoven pojav karakteristike, opažene v prvem eksperimentu. Torej povečanje razlike med meritvam ploščic ob povečanju zračnega tlaka. Statistični podatki v tabeli 3 prikazujejo podobna opažanja, kot pri drugem eksperimentu.



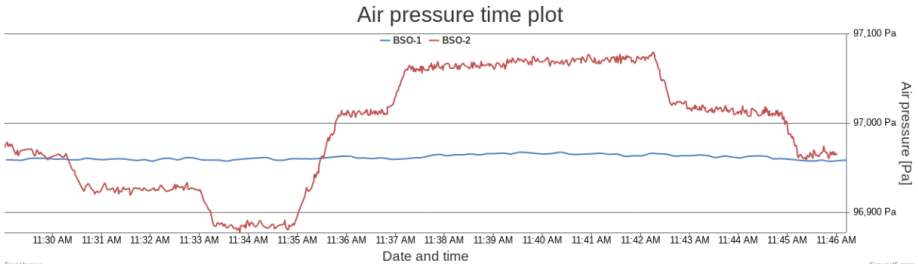
Slika 6: Eksperiment 3: meritve ploščic.



Slika 7: Eksperiment 3: razlike meritev.

3.3.4 Premikajoča meritev, s povprečenjem (nadmorska višina 340m +- 20m)

Pri zadnjem eksperimentu smo poskušali definirati primer bolj praktične uporabe. V stavbi Fakultete za računalništvo in informatiko smo postavili eno izmed ploščic na stacionarno mesto, v drugo nadstropje (ta nam služi kot referenca), z drugo pa smo se premikali po nadstropjih stavbe ter beležili meritve. Brez večjih težav lahko iz slike 8 razberemo premikanje ploščice po nadstropjih. Začeli smo v drugem nadstropju, se premaknili v tretje in četrto (padec tlaka), nato pa neposredno do prvega nadstropja in še pritličja (povišanje tlaka). Na koncu smo se vrnili na mesto prve ploščice.



Slika 8: Sprehod po fakulteti.

3.3.5 Ugotovitve

Prva hipoteza, opažena pri prvem eksperimentu je bila, da je razlika v meritvah med ploščicama večja ob višjem zračnem tlaku. To hipotezo z nadaljnjimi eksperimenti nismo uspeli potrditi, saj so rezultati eksperimentov prikazali nekoliko nasprotujoče se rezultate.
Druga hipoteza, ki smo jo hoteli dokazati je ta, da s povprečenjem večih meritev pridemo do bolj konsistentnih in točnih rezultatov. To hipotezo smo potrdili, saj lahko v tabeli s statističnimi podatki iz eksperimenta 1 (ko nismo povprečili meritev) (tabela 1) opazimo občutno večja odstopanja (povprečno odstopanje, največje odstopanje in standardni odklon), kot pri tabelah 2 in 3, ko smo povprečili 10 meritev preden smo jih poslali na strežnik MQTT.
Pri zadnjem eksperimentu vidimo možnost praktične uporabe, za potrebe določanja višine znotraj stavbe (npr. za določanje pozicije dvigala ipd.). Pri tem potrebujemo dve ploščici, eno za referenco ter drugo za meritev, saj se zračni tlak lahko občutno spremeni zaradi sprememb vremenskih razmer (iz prejšnjih eksperimentov opazimo, da se ob spremembah vremena lahko tlak spremeni za vrednost, ki bi predstavljala dva nadstropja).
Natančnejših karakteristik senzorja za merjenje zračnega tlaka iz izvedenih eksperimentov nismo uspeli ugotoviti.

4 Zaključek

V tem delu smo definirali brezžično senzorsko omrežje, preko katerega smo testirali senzor za zračni tlak BMP280, ki je vgrajen na senzorskem modulu GY-91. Uporabili smo tudi protokol MQTT, za pošiljanje zbranih meritev na strežnik, ki teče na računalniku Raspberry Pi. Na istem računalniku smo postavili še enega odjemalca MQTT v obliki Python skripte, ki meritve zapisuje v lokalno podatkovno bazo SQLite. Postavili smo tudi spletni strežnik, ki ponuja spletno stran za prikaz zbranih meritev.
Definirali smo štiri eksperimente, pri čimer so prvi trije služili za testiranje razlik v meritvah dveh identičnih razvojnih ploščicah. Prikazali smo grafe meritev ter statistične podatke o razlikah meritev med ploščicama. Izkaže se, da sta senzorja na ploščicah zadovoljivo natančna, če ju želimo uporabiti za določanje spremembo nadmorske višine, kakšnih drugih karakteristik senzorjev pa nismo uspeli dokazati. Pri četrtem eksperimentu smo prikazali praktično uporabo merjenja zračnega tlaka v stavbi, za določanje višine oz. nadstropja, kar se je (ob pogoju da uporabljamo dve ploščici) izkazalo za možno rešitev. Nadaljnje delo bi lahko vsebovalo več eksperimentov, preko katerih bi lahko bolj gotovo trdili o raznih karakteristikah senzorjev. Poleg tega bi bilo potrebno izvesti več daljših testiranj zanesljivosti določanja višine v stavbi.

Literatura

[1] Beautiful html5 javascript charts — canvasjs. https://canvasjs.com/. Dostopano: 8. 5. 2019.

[2] paho-mqtt - pypi. https://pypi.org/project/paho-mqtt/. Dostopano: 8. 5. 2019.

[3] Setting up an apache web server on a raspberry pi - raspberry pi documentation. https://www.raspberrypi.org/documentation/remote-access/webserver/apache.md. Dostopano: 8. 5. 2019.

[4] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660, 2013.