



SVEUČILIŠTE U ZAGREBU – GEODETSKI FAKULTET
UNIVERSITY OF ZAGREB – FACULTY OF GEODESY

Zavod za geomatiku; Katedra za geoinformatiku

Institute of Geomatics; Chair of Geoinformatics

Kačićeva 26; HR-10000 Zagreb, CROATIA

Web: <http://www.geof.hr>; Tel.: (+385 1) 46 39 222; Fax.: (+385 1) 48 28 081

Usmjerenje: geoinformatika

Diplomski rad

**INTEGRACIJA PRISTUPAČNIH I ŠIROKO DOSTUPNIH SENZORA U
SVRHU PRIKUPLJANJA PROSTORNIH PODATAKA**

Izradio:

Dalibor Sruk

Čugovečki put 7

Zagreb

dsruk@geof.hr

mentor: prof. dr. sc. Damir Medak, dipl. ing. geod.

voditelj: Dražen Odošić, dipl. ing. geod.

Zagreb, rujan 2013.

I. Autor

Ime i prezime:	Dalibor Sruk
Datum i mjesto rođenja:	27. 04. 1987., Zagreb

II. Diplomski rad

Naslov:	Integracija pristupačnih i široko dostupnih senzora u svrhu prikupljanja prostornih podataka
Mentor:	prof. dr. sc. Damir Medak
Voditelj:	Dražen Odošić, dipl. ing.

III. Ocjena i obrana

Datum zadavanja zadatka:	22. 01. 2013.
Datum obrane:	20. 09. 2013.
Sastav povjerenstva pred kojim je branjen diplomski rad:	prof. dr. sc. Damir Medak prof. dr. sc. Drago Špoljarić dr. sc. Ivan Medved

Zahvala: Zahvaljujem mentoru prof. dr. sc. Damiru Medaku i asistentima Mariu Mileru i Draženu Odošiću koji su mi svojim znanstvenim i stručnim savjetima pomogli u izradi ovoga diplomskog rada. Zahvaljujem sestri Ivani Sruk na pomoći pri obavljanju mjerenja, prijatelju Petru Trsliću na posuđenom Arduino modulu i Zavodu za prometno planiranje Fakulteta prometnih znanosti u Zagrebu na posuđenoj IMU mjernoj jedinici.

INTEGRACIJA PRISTUPAČNIH I ŠIROKO DOSTUPNIH SENZORA U SVRHU PRIKUPLJANJA PROSTORNIH PODATAKA

Sažetak: Ovaj rad opisuje postupak integracije dvaju senzora za potrebe mjerenja uzdužnog profila ceste. Integrirani su IMU mjerna jedinica i odometar. Odometar je zapravo računalni miš podešen da udovoljava potrebnim uvjetima mjerenja duljine prijeđenog puta. Podaci mjerenja senzora su spremeni u memoriju prijenosnog računala u realnom vremenu žičanom vezom. Korišten je Python programski jezik za izradu potrebnih skripta pomoću kojih su prikupljani, obrađivani i prikazani mjereni podaci. Vizualizacija dobivenih rezultata realizirana je u sučelju Python IDLE. Rezultati dobiveni integracijom senzora uspoređeni su sa rezultatima dobivenih geometrijskim nivelmanom milimetarske točnosti.

Ključne riječi: integracija senzora, IMU mjerna jedinica, odometar, uzdužni profil ceste, Python programski jezik

INTEGRATING ACCESSIBLE AND WIDELY AVAILABLE SENSORS FOR GEODATA COLLECTION

Abstract: This paper describes two sensor integration in order to measure lengthwise profile of the road. IMU and odometer were integrated. Odometer was presented by a computer mouse which was modified to measure traveled distance. Sensor data was stored real time in laptop's memory using wired connection. Python programming language was used to create scripts for data collection, data processing and result presentation. Results were visualized in Python IDLE. Sensor integration results were compared to results from millimeter accurate geometric leveling.

Keywords: sensor integration, IMU, odometer, road lengthwise profile, Python programming language

Sadržaj

1. Uvod	1
2. Prijašnji projekti slične teme	2
2. 1. IMU odometry	2
2. 2. DREAMBOT Project	3
3. Integracija senzora.....	4
4. Metode određivanja visinskih razlika.....	5
4. 1. Geometrijski nivelman	5
4. 2. Trigonometrijski nivelman	5
4. 3. Hidrostatski nivelman.....	6
4. 4. Barometrijski nivelman	7
4. 5. "Niveliranje" uz pomoć GNSS-a i geoida.....	7
5. Korištene tehnologije	8
5. 1. Inercijalna mjerna jedinica	8
5. 1. 1. Realizacije inercijalne mjerne jedinice	10
5. 2. Aplikacija 3DM-GX1	11
5. 3. Računalni miš	11
5. 3. 1. Općenito o računalnim miševima	11
5. 4. Arduino modul.....	13
5. 5. Python.....	14
5. 6. Nivelir i nivelmanska letva.....	15
6. Praktični rad	16
6. 1. Testiranje prenamjene računalnog miša u odometar	16
6. 2. Mjerenje udaljenosti pomoću Arduino modula	21
6. 3. Integracija mjerenja inercijalne mjerne jedinice	23
6. 4. Integracija senzora	24
6. 5. Programiranje u Python-u.....	29
6. 6. Montiranje senzora na bicikl i obavljanje mjerenja.....	30
6. 6. 1. Montiranje senzora.....	30
6. 6. 2. Mjerenje pomoću integriranih senzora	31
7. Diskusija.....	33
7. 1. Prenamjena računalnog miša u odometar	33
7. 2. Rezultati mjerenja udaljenosti pomoću Arduino modula	33

7. 3. Rezultati mjerenja integracijom senzora	34
8. Zaključak	39
Popis kratica i stranih riječi	40
Literatura	41
Popis URL-ova	41
Popis grafikona.....	43
Popis slika	43
Popis tablica	44
Prilog 1 Izvorni kod skripte za prikupljanje podataka mjernog sustava sa objašnjenjima.....	45
Prilog 2 Izvorni kod skripte za obradu podataka mjernog sustava i prikaz rezultata sa objašnjenjima	47
Životopis.....	49

1. Uvod

Geodezija kao znanost je usko vezana uz razvoj tehnologije jer koristi mjerne instrumente u fazi prikupljanja podataka tj. u fazi mjerenja.

Prvi korišteni instrumenti bili su veliki, teški, skupi, a često i neprecizni te netočni. Najpoznatiji stari instrumenti su teodolit i optički nivelir. Teodolit se koristi za mjerenje smjerova iz kojih računamo kuteve, a optički nivelir se koristi za mjerenje visine iznad horizontalne osi iz kojih računamo razlike visina točaka.

Prvotni mjerni instrumenti su bili analogni, dok danas prevladavaju digitalni instrumenti. U fazi mjerenja princip izmjere smjerova odnosno kuteva se proširuje s novim instrumentima, kao što je inercijalna mjerna jedinica (eng. IMU). IMU omogućuje mjerenje tri kuta rotacije: ljuľljanja, posrtanja i skretanja (eng. *roll*, *pitch*, *yaw*). IMU se uglavnom koristi kao dopuna GNSS mjerenjima zbog veće rate davanja mjerenja (>50 Hz), te se time omogućuje kontinuirano mjerenje prilikom kretanja po zadanoj ruti.

Glavne geodetske zadaće su određivanje položaja točke (E, N ili φ , λ) i određivanje visinskih razlika. Načini određivanja visinskih razlika (prijenosa visina) koji se danas koriste u geodetskoj struci su: geometrijski nivelman, trigonometrijski nivelman, hidrostatski nivelman, barometrijski nivelman i "niveliranje" uz pomoć GNSS-a i geoida. Danas je moguća integracija različitih senzora u jedinstveni mjerni sustav, kao i zajednička obrada njihovih informacija što omogućuje pridobivanje boljih, točnijih i pouzdanijih, ili posve novih informacija. Cilj integracije senzora pritom je, da se tražene informacije zahtijevane kvalitete odrede maksimalno efikasno uz minimalne troškove (Bačić, Ž., ISuG_P6, 2012/2013).

Imajući to na pameti, motivacija za diplomski rad je temeljena na integraciji novih senzora koji se sve više koriste u geomatici. Cilj je napraviti mjerni sustav koji će koristeći poznata pravila i formule iz trigonometrijskog nivelmana mjeriti uzdužni visinski profil rute kojom se sustav kreće. Korištenjem inercijalne mjerne jedinice eliminira se odstupanje mjerenja uzrokovano refrakcijom vizurne osi vezano uz optički nivelir. Druga prednost ovakvog sustava je mogućnost izrade puno gušćeg profila (manja međusobna udaljenost točaka profila) od profila koji se može izraditi optičkim nivelirrom.

2. Prijašnji projekti slične teme

U ovom podpoglavlju nalaze se projekti koji su napravljeni na sličnu temu integriranja IMU i odometra. Uglavnom se autori bave integracijom za potrebe položajne navigacije, odnosno određivanja položaja u ravnini (X, Y koordinate). Ovi projekti se ne bave visinskom koordinatom, ali su dobro poslužili za razumijevanje nekih faza integracije senzora kojom se ovaj rad bavi.

2. 1. IMU odometry

Autor David P. Anderson se u ovom projektu bavi integracijom IMU mjerne jedinice i odometra za potrebe određivanja položajnih koordinata (X, Y) robota i izbjegavanje prevrtanja robota. Na web stranici je objavio detalje projekta kao što su kodovi aplikacije u programskom jeziku C koju robot koristi.

Dio koda aplikacije koji računa položaj robota:

```
X_pos += (float)(avg_inches * sin(theta));
Y_pos += (float)(avg_inches * cos(theta));,
```

gdje su : X_{pos} i Y_{pos} položajne koordinate, avg_inches prosječna udaljenost i θ kut skretanja (yaw).

Ovaj projekt je pomogao kod koncepcije izrade matematičkog modela mjernog sustava jer objašnjava način integracije IMU jedinice i odometra. Detalji se mogu pronaći na internetskoj stranici projekta (URL 5).



Slika 1. Robot IMU odometry (URL 5)

2. 2. Dreambot Project

U ovom projektu, autor Carlos Agell se bavi konceptom korištenja računalnog kugličastog miša kao brzinomjera/odometra za potrebe upravljanja robotom.



Slika 2. Robot Dreambot Project (URL 6)

Ovaj projekt je poslužio kao referenca za shvaćanje koncepta prenamjene računalnog miša u odometar. Detalji se mogu pronaći na internetskoj stranici projekta (URL 6).

3. Integracija senzora

Danas je moguća integracija različitih senzora u jedinstveni mjerni sustav, kao i zajednička obrada njihovih mjerenja što omogućuje pridobivanje boljih, točnijih i pouzdanijih ili posve novih informacija. Cilj integracije senzora pritom je da se tražene informacije zahtijevane kvalitete odrede maksimalno efikasno uz minimalne troškove. (Bačić, Ž., ISuG_P6, 2012/2013).

U ovisnosti od zadatka, razina integracije senzora može biti različita. U pojedinim slučajevima neki senzor može poslužiti samo kao dopuna (ili kontrola) drugog senzora (GNSS + odometar), dok je u drugim slučajevima tek integracijom dva ili više senzora moguće dobiti željene informacije (GNSS + INS¹ + kamere).

Generalno je potrebno definirati koordinatne sustave u kojima pojedini senzori rade i transformacije između njih, riješiti tehničke probleme integracije različitih uređaja i problem modeliranja raznovrsnih mjerenja. Modeliranje se dijeli na dva glavna modela:

- dinamičko i
- kinematičko modeliranje.

Dinamičko modeliranje se bavi objašnjavanjem gibanja definiranjem sila ili drugih zakona interakcije u cilju opisivanja djelovanja jednog fizikalnog sustava na drugi.

Opisuje se izrazom: $m \frac{d^2 r}{dt^2} = F(r, v, t)$,

gdje su: m – masa, r - prijeđeni put, t – vrijeme.

Kinematičko modeliranje bavi se opisivanjem gibanja bez obaziranja na uvjete ili interakcije potrebne da se pojedino gibanje izvrši.

Opisuje se izrazom: $mr(t) = fp_i(t_j)$,

gdje su: p_i – diskretno mjerenje i t - vrijeme.

Kod integracije senzora važno je kinematičko modeliranje.

¹ INS – inercijani navigacijski sustav

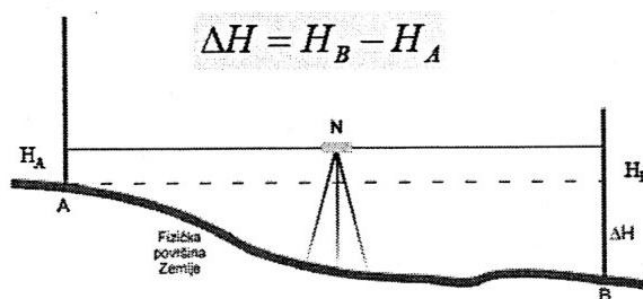
4. Metode određivanja visinskih razlika

Načini određivanja visinskih razlika (prijenosa visina) koji se danas koriste u geodetskoj struci su: geometrijski nivelman, trigonometrijski nivelman, hidrostatski nivelman, barometrijski nivelman i "niveliranje" uz pomoć GNSS-a i geoida.

4. 1. Geometrijski nivelman

Geometrijski nivelman je metoda određivanja visinskih razlika pomoću horizontalne vizure, a instrument koji se koristi za mjerenje zove se nivelir. Nivelir se sastoji od durbina koje se može okretati oko svoje osi, time opisujući horizontalnu ravninu kojom se sijeku letve postavljene na točkama A i B, čija se visinska razlika određuje. Ako na točkama A i B podjeljenje počinje od terena, te ako se na njima očitaju mjesta H_A i H_B na kojima ih presijeca horizontalna vizura nivelira N, visinska razlika ΔH između točaka A i B se računa:

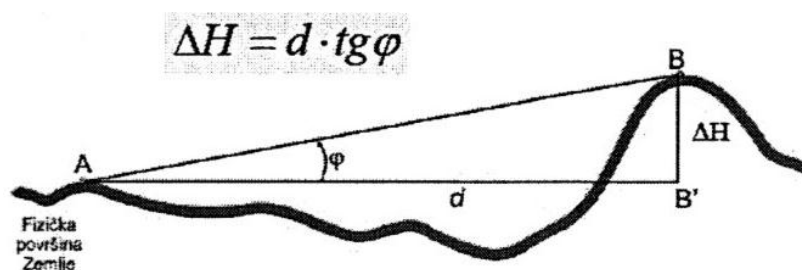
$$\Delta H = H_B - H_A \text{ (Bašić, T., DI 09, 2010/2011).}$$



Slika 3. Geometrijski nivelman (Bašić, T. DI 09, 2010/2011)

4. 2. Trigonometrijski nivelman

Kada je poznat vertikalni kut i horizontalna ili kosa duljina između dvije točke može se pomoću fundamentalne trigonometrije izračunati razlika u visinama između te dvije točke. Takva metoda posrednog određivanja visinske razlike je vrlo korisna pri određivanju visina na neravnom, vrlo strmom terenu, gdje se upotreba direktnih metoda mjerenja čini nepraktična i dugotrajna. Uz pretpostavku da su dvije točke A i B tako blizu da se nivo ploha AB' između njih može smatrati ravninom, najjednostavnije će se visinska razlika ΔH dviju točaka dobiti uz pomoć njihove međusobne udaljenosti d i vertikalnog kuta φ pod kojim se one dogledaju (Bašić, T., DI 09, 2010/2011).



Slika 4. Trigonometrijski nivelman (Bašić, T., DI 09, 2010/2011)

Za obavljanje mjerenja se koristi mjerni instrument totalna mjerna stanica. Ona koristi dalekozor (durbin) za određivanje zenitne udaljenosti, odnosno vertikalnog kuta. Poznato je da refrakcija utječe na optičku os, pa je stoga optička os u prostoru krivulja, a ne prema pretpostavci ravna linija.

Na točnost određivanja visinske razlike utječe i zakrivljenost Zemlje. Formula za visinsku razliku kod koje je uzet u obzir utjecaj zakrivljenosti Zemlje glasi:

$$\delta h = d \tan \varphi + c_1.$$

Ako Zemlju aproksimiramo kuglom sa polumjerom $R=6377000$ m, c_1 će imati vrijednosti ovisno o udaljenosti d :

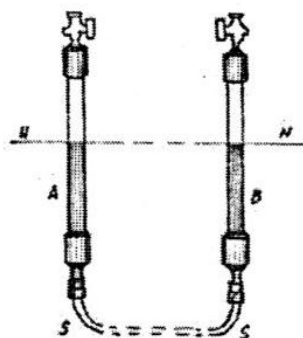
Tablica 1. Vrijednosti člana c_1 ovisno o udaljenosti d (Izmjera zemljišta, Džapo, M., 2008)

d (m)	100	200	300	400	500	1000	2000
c_1 (m)	0.001	0.003	0.007	0.01	0.02	0.08	0.31

Ako mjerimo vertikalni kut na udaljenostima manjim od 100 m, utjecaj zakrivljenosti Zemlje je zanemariv, što je i jest slučaj kod obavljenih mjerenja u ovom radu (udaljenost 0.049 m).

4. 3. Hidrostatski nivelman

Kod mjerenja visinskih razlika hidrostatskim nivelmanom koristi se gumeno crijevo napunjeno vodom, na čijim krajevima se nalaze posude s vodom (Slika 5). Po zakonu spojenih posuda nivo vode u jednoj i drugoj posudi je isti, a pored vizualnog očitavanja moguće je očitavati i posebnim mjernim uređajima za precizno očitavanje (Bašić, T., DI 09, 2010/2011).



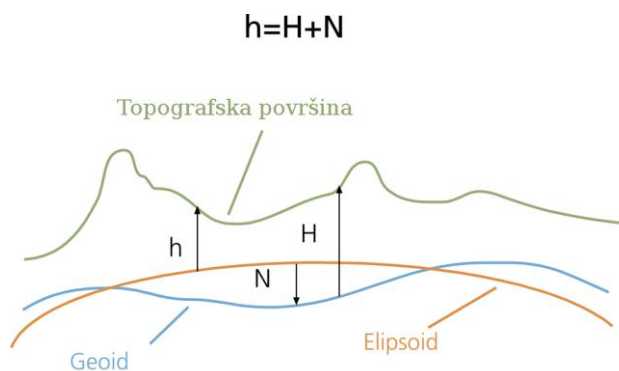
Slika 5. Prikaz crijeva napunjenog vodom (Bašić, T. DI 09, 2010/2011)

4. 4. Barometrijski nivelman

Kako pritisak zraka pada s povećanjem visine, moguće je mjeriti visinske razlike pomoću mjerenja pritiska zraka i primjene tzv. barometrijske formule. Mjerenje barometrijskim nivelmanom počinje i završava na točki poznate nadmorske visine, na taj način se kalibrira mjerna skala barometrijskog nivelmana (Bašić, T., DI 09). Danas se MEMS² barometri mogu naći u nekim pametnim telefonima kao što su Samsung Galaxy Nexus, Samsung Galaxy S3, Sony Xperia GO, LG Nexus 4.

4. 5. "Niveliranje" uz pomoć GNSS-a i geoida

Zahvaljujući pouzdanosti GNSS tehnologije kod definiranja elipsoidnih visina točaka, kao i sve točnijim rješenjima za plohu geoida na globalnom i/ili lokalnom nivou, danas se sve više u geodeziji koristi prijenos visina uz pomoć ove dvije veličine (Bašić, T., DI 09, 2010/2011). Formula glasi: $H = h - N$, gdje je H ortometrijska visina, h elipsoidna visina i N undulacija geoida.



Slika 6. Veza između elipsoidne i ortometrijske visine (URL 1)

² MEMS- Microelectromechanical Systems, u prijevodu vrlo maleni elektromehanički sustavi

5. Korištene tehnologije

Za određivanje visinske razlike trigonometrijskim nivelmanom potrebno je izmjeriti zenitnu udaljenost ili vertikalni kut i udaljenost (horizontalna ili kosa duljina). Mjerni sustav korišten u tu svrhu realiziran je integracijom inercijalne mjerne jedinice (IMU) i računalnog miša. Računalni miš mjeri tj. daje informaciju o udaljenosti između dvije točke, a inercijalna mjerna jedinica daje kuteve orijentacije sustava odnosno potrebni vertikalni kut. Programiranje potrebnih aplikacija i skripta vršeno je u Python programskom jeziku. Ispitana je upotreba Arduino modula za mjerenje udaljenosti pomoću dijelova ciklokompjutera.

5. 1. Inercijalna mjerna jedinica

U ovom radu korištena je inercijalna mjerna jedinica 3DM-GX1 proizvođača LORD Corporation, MicroStrain® Sensing Systems iz Vermont-a, SAD. Ova jedinica se sastoji od:

- 3 žiroskopa,
- 3 akcelerometra,
- 3 magnetometra,
- multiplexer-a,
- 16 bit A/D konvertera,
- mikrokontrolera.

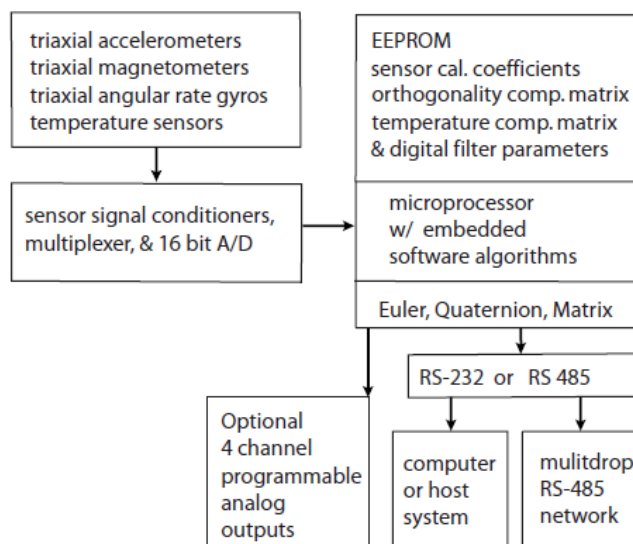
Specifikacije jedinice su dane u tablici 2.

Tablica 2. Specifikacije inercijalne mjerne jedinice (URL 7)

Orijentacijski raspon (posrtanje, ljuljanje, skretanje)	360° sve osi ± 90°, ± 180°, ± 180° (Euler kutevi)
Raspon senzora	žiroskop: ± 300 °/s FS akcelerometar: ± 5 g FS magnetometar: ± 1.2 Gauss FS
A/D rezolucija	16 bit
Nelinearnost akcelerometar	0.2 %
Bias stabilnost akcelerometar	0.010 g
Nelinearnost žiroskop	0.2 %
Bias stabilnost žiroskop	0.7 °/s
Nelinearnost magnetometar	0.4 %
Bias stabilnost magnetometar	0.010 Gauss
Rezolucija orijentacije	< 0.1° minimum
Ponavljanje	0.20°

Točnost	$\pm 0.5^\circ$ tipično za statične uvjete $\pm 2.0^\circ$ tipično za dinamičke (cikličke) uvjete i proizvoljne orijentacijske kuteve
Izlazni načini	matrica, quaternion, Euler kutevi i devet skalirani senzori sa temperaturom
Digitalni izlazi	serijski RS-232 i RS-485 opcionalni sa softwareskim programiranjem
Analogni izlaz	4 kanalni, 0-5 V potpuno programibilni analogni izlazi
Rate digitalnog izlaza	100 Hz za Euler, Matrix, Quaternion 350 Hz za devet ortogonalne senzore
Serijska rata podataka	19.2/38.4/115.2 kbaud, programibilno
Napon napajanja	5.2 V DC minimalno, 12 V DC maksimalno
Struja napajanja	65 mA
Konektori	jednoključni LEMO, dva za opciju RS-485
Radna temperatura	-40 do +70°C sa kućištem -40 do +85°C bez kućišta
Kućište	64 mm x 90 mm x 25 mm
Težina	75 grama sa kućištem, 30 grama bez kućišta
Šok limit	1000 g (ne uključen), 500 g (uključen)

Prikaz principa rada IMU jedinice:



Slika 7. Princip rada jedinice (URL 7)

Ova jedinica podržava dva načina rada: kontinuirani i na zahtjev. Kad je jedinica u kontinuiranom načinu, neprekidno šalje mjerenja, a kad je u načinu na zahtjev mjerenja se

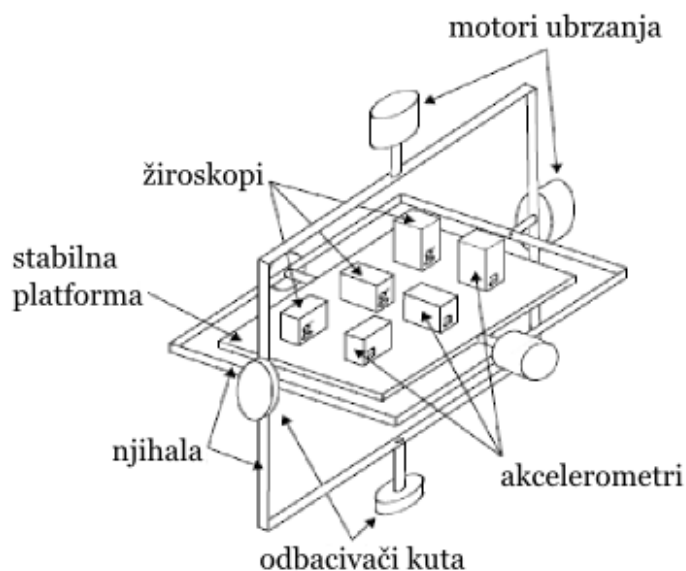
dobivaju slanjem odgovarajuće naredbe mikrokontroleru koji onda proslijeđuje podatke mjerenja. Rata slanja podataka mjerenja IMU jedinice u načinu na zahtjev je 76 Hz tj. 76 mjerenja u sekundi prema specifikaciji proizvođača.

5. 1. 1. Realizacije inercijalne mjerne jedinice

IMU se danas realizira u dva osnovna oblika:

- kao stabilna platforma i
- kao *strapdown* sustav.

Sustavi na stabilnoj platformi (Slika 8) osiguravaju stabilan referentni okvir da bi izolirali žiroskope i akcelerometre od utjecaja gibanja tijela (vozila, aviona, broda, ..). Riječ je o polu-analitičkom sustavu kojeg opisujemo kao u prostoru stabilizirani inercijali navigacijski sustav (INS) ili platforma u lokalnom referentnom sustavu. Kod realizacije takvih sustava lokalna platforma je uvijek tako orijentirana da zadržava z-os paralelno s lokalnom normalnom na referentni elipsoid, a da su horizontalne osi usmjerene ka sjeveru i istoku.

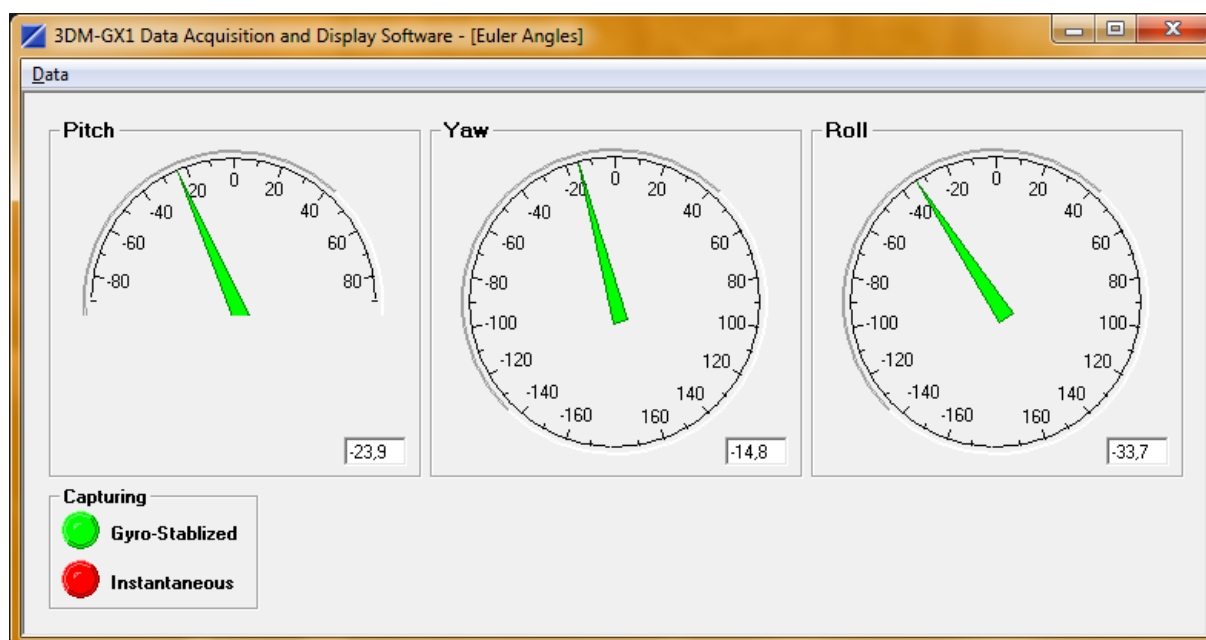


Slika 8. Stabilna platforma

Strapdown IMU je analitički inercijalni sustav, u kojem senzori nisu izolirani od tijela, ali koji koristi izlazne vrijednosti žiroskopa da analitički sračuna relativnu orijentaciju sustava u nekom trenutku u odnosu na početno stanje sustava. *Strapdown* sustavi bolje odgovaraju GNSS sustavu i zato se češće koriste pri integracijama. U ovom radu korišten je *strapdown* sustav.

5. 2. Aplikacija 3DM-GX1

Kako bi se bolje shvatile koordinatne osi IMU jedinice i provjerila kalibriranost iste, korištena je aplikacija koja dolazi s prije spomenutom jedinicom. Aplikacija se može preuzeti na internetu (URL 7). U aplikaciji se može lokalno spremati sve podatke koje jedinica mjeri, ali i vidjeti njihove vrijednosti u realnom vremenu. Unutar aplikacije je moguće podešavanje učestalosti prikupljanja podataka IMU jedinice, kao i način rada kontinuirani ili na zahtjev.



Slika 9. Sučelje aplikacije 3DM-GX1

5. 3. Računalni miš

Potreba za preciznim određivanjem udaljenosti (kose duljine) rezultirala je prenamjenom računalnog miša u precizni odometar. Računalni miš je ujedno i praktično rješenje jer je namijenjen spajanju na računalo. Operativni sustav posjeduje sve potrebne protokole za razmjenu podataka s mišem, pa nema potrebe za dodatnim komponentama koje će podatke odometra nekim protokolom slati u računalo.

5. 3. 1. Općenito o računalnim miševima

Danas postoje kugličasti, optički i laserski računalni miševi. Svaka vrsta ima različite principe rada. Ti principi definiraju ograničenja njihovoj prenamjeni u odometar. Pošto je korišten kugličasti miš, objašnjen je samo njegov princip rada, kasnije u tekstu.

Konstrukcija je uglavnom ista kod svih korištenih računalnih miševa, a isti dijelovi su:

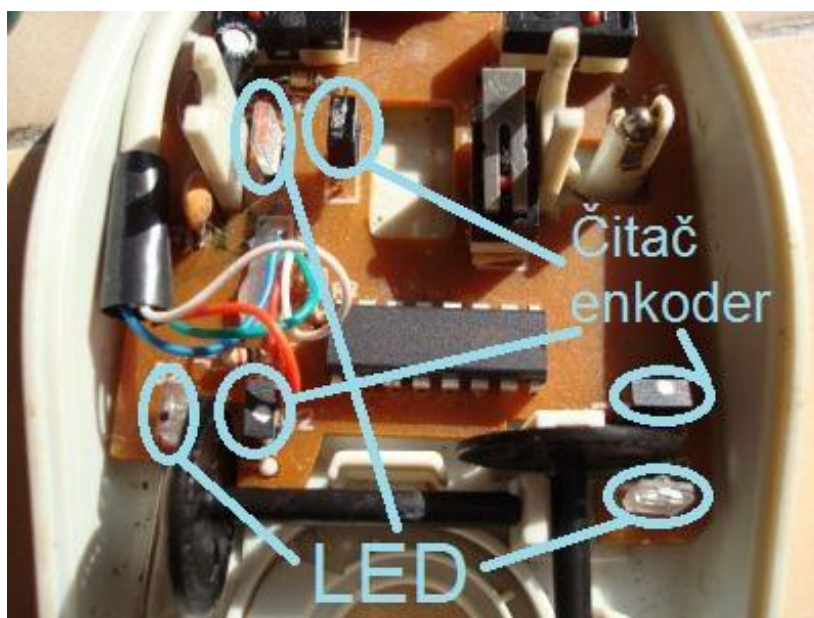
- tri tipke: lijeva, desna i *scroll*,
- *scroll* kotač,
- mehanizam za određivanje pomaka miša.

Glavno ograničenje kod korištenja računalnog miša za potrebe mjerenja duljine je rata osvježavanja podataka miša. Moguće rate su: 125 Hz, 250 Hz, 500 Hz, 1000 Hz. Rata označava maksimalan broj mjerenja tj. maksimalan broj prepoznavanja pomaka miša u jednoj sekundi.

Princip detektiranja pomicanja kugličastog miša je slijedeći:

Kao što se vidi na slici 10, LED emitira svjetlo koje sa druge strane očitava čitač (*encoder*). LED dioda ima dva mjesta emitiranja svjetla, gornje i donje koje onda ovisno o prolasku svjetlosti kroz kotač određuju smjer i vrijednost pomaka miša. Detaljnije objašnjenje moguće je naći na stranicama (URL 2 i URL 3).

Miš osim dvije tipke (lijeve i desne) sadrži i tzv. *scroll* kotač za pomicanje klizača prozora gore/dolje. Princip rada je isti kao i kod kotača miša za pomicanje kursora miša lijevo/desno i gore/dolje po ekranu.



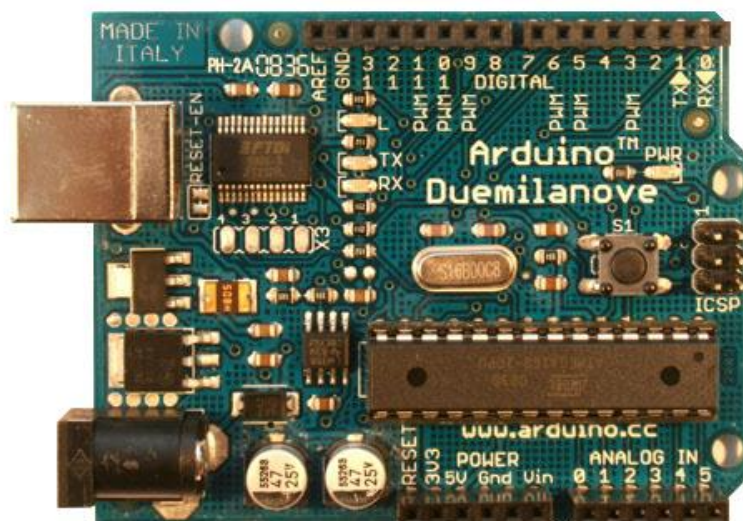
Slika 10. Prikaz enkodera miša

5. 4. Arduino modul

Arduino modul (Slika 11) je pločica s mikrokontrolerom ATmega168. Ima 14 digitalnih ulaza/izlaza (od kojih su 6 korišteni kao PWM izlazi), 6 analognih ulaza, kristalni oscilator od 16 MHz, USB konekciju, nastavak za napajanje i ICSP naglavnik i gumb za resetiranje. Jednostavno se spoji na računalo preko USB kabela koji ujedno i služi kao napajanje. Moguće je i vanjsko napajanje preko AC-DC adaptera ili pomoću baterije.

Tablica 3. Specifikacije Arduina (URL 22):

Mikrokontroler	ATmega168
Operativni napon	5 V
Ulazni napon (preporučeno)	7-12 V
Ulazni napon (ograničenja)	6-20 V
Digitalni ulazni/izlazni pinovi	14 (od kojih 6 nude PWM izlaz)
Analogni ulazni pinovi	6
DC struja po ulaz/izlaz pinu	40 mA
DC struja za 3.3 V pin	50 mA
Flash memorija	16 KB od kojih 2 KB korištenih od bootladera
SRAM	1 KB
EEPROM	512 bajta
Brzina sata	16 MHz



Slika 11. Arduino (URL 22)

5. 5. Python

Python je programski jezik koji je 1990. godine prvi razvio Guido van Rossum. To je objektno orijentirani, interaktivni i interpreterski jezik kojeg su od 2000. godine prihvatile i ugledne institucije kao što su MIT, NASA, IBM, Google. Koristeći Python nećete naići na nove i revolucionarne ideje i rješenja u programiranju već ćete na jednome mjestu imati na optimalan način ujedinjene ideje i načela rada mnogih drugih programskih jezika. On je snažan i jednostavan istodobno jer je spoj tradicionalnih skriptnih jezika i sistematskih jezika, pa na taj način omogućuje programeru lakšu orijentiranost na problem s obzirom da zahtjeva manje razmišljanja o samom jeziku. Potrebno je još napomenuti kako je Python slobodan, softver otvorenog koda s izuzetno dobrom podrškom, literaturom i dokumentacijom tako da nije problem nabaviti isti i naučiti ga koristiti.

Za veću funkcionalnost Pythona i njegovo lakše korištenje potrebno je dodatno instalirati pojedine biblioteke (eng. *library*). Instalacijom istih povećana je funkcionalnost programskog jezika jer su se u programiranju mogle koristiti već unaprijed definirane funkcije i operacije pa je samim time program lakše izraditi, a i program je pregledniji i kvalitetnije strukturiran.

U izrađenim skriptama korišteni su slijedeći Python moduli i biblioteke:

- pythoncom – biblioteka koji služi za primanje svih poruka koje vraća sistem (URL 8),
- pyHook – biblioteka koji služi za nadzor unosa tipkovnice i miša (URL 9),
- serial – modul za sve potrebne operacije vezane uz serijski *port* (URL 10),
- struct – biblioteka koja služi za manipuliranje podacima koji su oblika C programskog jezika (URL 11),
- datetime – modul koji se koristi za očitavanje sistemskog vremena i datuma (URL 12),
- math – modul koji služi za matematičke operacije (npr. kvadrat, korijen, tan, sin itd.) (URL 13),
- win32api – modul koji služi za pozivanje Win32 APIa (URL 15),
- win32con – modul koji služi za korištenje konstanti Win32 (URL 14),
- msvcrt – modul koji služi za korištenje microsoft visual biblioteke (URL 16),
- Arduino – modul koji služi za korištenje Arduino modula u Python jeziku (URL 17),

- shapely – modul koji se koristi za rad s geometrijom (točka, linija, poligon...) (URL 18),
- matplotlib – modul koji se koristi za iscrtavanje geometrije (URL 19).

5. 6. Nivelir i nivelmanska letva

U ovom radu korišten je digitalni nivelir Sokkia SLD 30 (Slika 12) i nivelmanska letva Sokkia BGS 27 (Slika 13). Nivelir je visoke kvalitete i nudi dodatke kao što su: prepoznavanje nagnutosti letve i prepoznavanje okrenute letve naopačke. Letva je visokoprecizna, napravljena od fiberglasa i nudi kodiranu podjelu za digitalne nivelire i brojčanu podjelu za rad sa optičkim nivelirom.



Slika 12. Nivelir



Slika 13. Nivelmanska letva

Tablica 4. Specifikacije digitalnog nivelira (URL 21)

Točnost mjerenja visine	1 mm (Sa fiberglas RAB-kod letvom)
Točnost mjerenja dužine	[<10 m = +/-10 mm] [10 m do 50 m = +/-0.1 %xD] [>50 m = 0.2 %xD]
Veličina ekrana	0.0001 / 0.001 m
Domet snimanja	1.6 do 100 m
Povećanje	32x
Kompenzator	kompenzator na njihalu sa magnetskim prigušivanjem
Veličina memorije	2000 točaka
Komunikacija	RS-232C
Zaštita od prašine/vode	IPX4
Radna temperatura	-20 do +50°C (-4 do +122°F)
Radno vrijeme	više od 8.5 sati
Veličina	D 158 x Š 257 x V 182 mm
Težina	2.4 kg

6. Praktični rad

U ovom poglavlju je dan pregled postupaka koji su rezultirali izradom mjernog sustava i popratnih aplikacija.

6. 1. Testiranje prenamjene računalnog miša u odometar

Za potrebe rada, istestirana su tri miša, za svaku vrstu po jedan. Konstrukcija je uglavnom ista kod svih korištenih računalnih miševa, a isti dijelovi su:

- tri tipke: lijeva, desna i *scroll*,
- *scroll* kotač,
- mehanizam za određivanje pomaka miša.

Prvi miš koji je korišten bio je Logitech RX 300 optički miš sa 800 dpi. U operativnom sustavu Ubuntu izvršena je kreirana Python skripta kako bi se ispisali sirovi podaci miša (interne x i y koordinate miša). Korišten je Linux OS jer se dobivanje sirovih podataka miša u Python programskom jeziku pokazalo jednostavnije i lakše razumljivo od istog postupka na Windows 7 OS (nema potrebe za Python modulima itd.).

Za vrijednost varijable imena *mouse* se uzima vrijednost sirovih podataka koju Linux dobiva od računalnog miša (vrijednost sadrži podatke tipa C koji opisuju parametre miša kao što su pomak po x osi, pomak po y osi, pomak *scroll* kotača, pritisak lijeve i desne tipke itd.) i iz te vrijednosti se uzimaju pomaci po x i y osi miša, te se za x os zbraja ukupni pomak i prikazuje na ekranu u beskonačnoj petlji.

Izvorni kod Python skripte za Linux OS:

```
mouse = file('/dev/input/mouse0')
suma = 0
while True:
    status, dx, dy = tuple(ord(c) for c in mouse.read(3))
    def to_signed(n):
        return n - ((0x80 & n) << 1)
    dx = to_signed(dx)
    dy = to_signed(dy)
    suma = suma + dx
    print suma
```

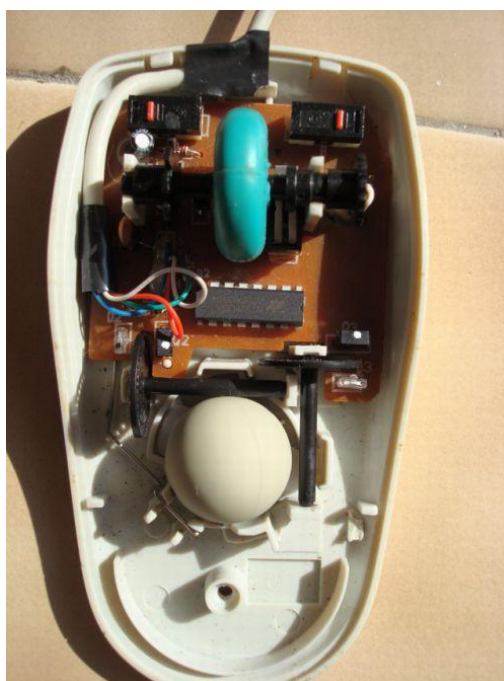
To testiranje je pokazalo da ovaj miš u udaljenosti od 1 centimetra "putuje" od vrijednosti 0 do 2032 po x osi miša. To znači da miš u svakom centimetru ima 2032 očitavanja koji predstavljaju osjetljivost miša od 800 točaka po inču. To znači da je ograničenje brzine

kretanja miša pri rati od maksimalno mogućih 1000 Hz oko 0.5 cm/s, ako ne želimo izgubiti podatke mjerenja miša. To znači da bi se u ovom slučaju bicikl mogao kretati najvećom brzinom od oko 0.02 km/h, što je neprihvatljivo i nema smisla raditi mjerni sustav koje se može kretati tako malom brzinom.

Drugi miš koje je testiran preko iste Python skripte u Linux OS je laserski Logitech RX 1000, s 1000 dpi-a. Ovaj miš u jednom centimetru ima oko 2540 očitavanja po x osi. To znači da je za ovaj miš ograničenje brzine 0.01 km/h što je još lošije od prethodno testiranog optičkog miša.

Miš osim dvije tipke (lijeve i desne) sadrži i tzv. *scroll* za pomicanje klizača prozora gore/dolje.

Iako sve tri vrste računalnog miša imaju isti princip određivanja pomaka tzv. *scroll* kotača miša, najjednostavnije je bilo prilagoditi stari kugličasti miš da mjeri udaljenost. S toga je u ovom radu korišten stari kugličaski miš SM-18, proizvođača Microsoft. Na slici 14. prikazana je konstrukcija korištenog miša.



Slika 14. Prikaz konstrukcije korištenog miša

Scroll kotač je korišten jer je napravljen od gume, pa zbog toga osigurava bolje prijanjanje uz obruč kotača bicikla od prijanjanja plastičnog kotača koji se pokreće kuglicom miša. Druga prednost je veličina, jer je *scroll* kotač veći od kotača koji služi za mjerenje pomaka miša i

zbog toga omogućuje kretanje pri većim brzinama od manjeg kotača. Što je kotač veći, to je broj očitavanja pomaka kotača manji.

Standardna rata USB porta kompjutera je 125 Hz, pa je pretpostavljeno da je rata čitanja *encodera* (svih triju) miševa 125 Hz. Empirijskim testiranjima i pomoću aplikacije *dimr* (URL 4) došao sam do zaključka da *scroll* kotač PS/2 miša ima ratu od 62 Hz neovisno o postavkama rate miša (125, 250, 500 i 1000 Hz). *Scroll* kotačić optičkog i laserskog miša ima ograničenje od 125 Hz neovisno o postavkama rate miša (125, 250, 500 i 1000 Hz).

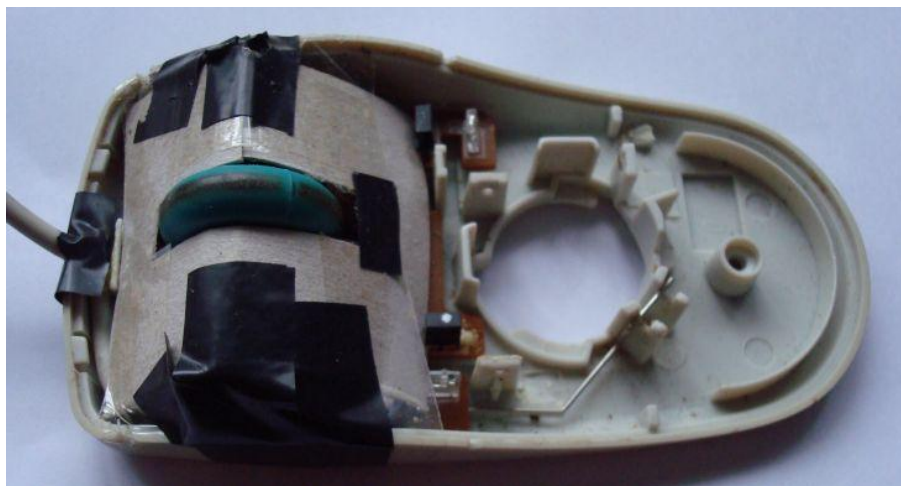
Pošto je rata IMU jedinice u načinu rada na zahtjev 76 Hz, tj. veća od rate *scroll* kotača miša, glavno ograničenje brzine kretanja bicikla je rata *scroll* kotača miša i direktno je vezana uz tu vrijednost.

Zbog specifičnosti određivanja smjera okretanja *scroll* kotača miša, najmanji broj očitavanja koji okretanje kotača za jedan puni krug uzrokuje je 2. Pošto originalna konstrukcija miša po jednom okretaju *scroll* kotača očitava 9 pomaka, izrađena je maskica od kartona koja je zamijenila kotačić sa prorezima koji originalno dolazi u mišu.



Slika 15. Izrađena maskica i postojeći kotač

Pošto se očitavanje vrši pomoću detektiranja emitiranoga svjetla iz LED diode, potrebno je osigurati da sunčeva svjetlost ne remeti taj postupak. Zbog toga je dio koji očitava pomak *scroll* kotača zatvoren kartonom (Slika 16).



Slika 16. Zatvoren dio za očitavanje okretanja *scroll* kotača miša

Zbog činjenice da je miš postavljen na okvir kotača (obruč), a ne na vrh kotača, kotač prijeđe veću udaljenost od one koju se dobije iz mjerenja miša. Slika 17 najbolje objašnjava problematiku.



Slika 17. Razlika polumjera obruča i kotača

Zbog razlike polumjera cijelog kotača i okvira kotača, potrebno je podesiti očitavanje *scroll* kotača miša na „pravu“ udaljenost koju kotač prijeđe sa udaljenosti koju zapravo on prijeđe i koja se dobije iz sirovih podataka. „Pravi“ put koji mjerni sustav prijeđe između dva očitavanja miša je izračunat pomoću omjera polumjera cijelog kotača i polumjera okvira kotača:

$$"pravi"put = (\frac{r_{kotač}}{r_{okvir}} * O_{scrollkotač})/2 = (\frac{210cm}{165cm} * 7.5cm)/2 = 4.9 cm$$

gdje su :

$O_{scrollkotač}$ – opseg *scroll* kotača miša,

$r_{kotač}$ –polumjer cijelog kotača,

r_{okvir} – polumjer okvira kotača.

Ovaj račun je izrađen za dobivanje približnih vrijednosti jer je praktički nemoguće točno u milimetar mjernom vrpcom izmjeriti polumjer kotača i polumjer okvira kotača. Izračun maksimalne brzine kretanja bicikla kada se koristi *scroll* kotač PS/2 miša slijedi u nastavku.

Pošto se jednim okretajem kotača dobiju dva mjerenja, a opseg kotača je 7.54 cm, to znači da u našem slučaju u jednoj sekundi pri brzini od 10 km/h (278 cm/s) imamo 278/4.9, tj. otprilike 57 mjerenja kada se u obzir uzme razlika polumjera kotača i okvira. To znači, ako je ograničenje miša 57 mjerenja u sekundi bicikl se može kretati najbrže 9.8 km/h, ako se želi spremiti sva mjerenja, tj. ako se ne želi imati izgubljena mjerenja zbog ograničenja rate očitavanja kretanja *scroll* kotača. Iz ovoga računa dobivene su opet približne vrijednosti jer je rezultat direktno vezan za „pravi“ prijeđeni put.

Detaljni račun za određivanje maksimalne brzine bicikla:

$$O = 2 * r * \pi = 2.4cm * \pi = 7.54 cm$$

$$v = \frac{mjer}{s} * \frac{O}{2} * \frac{r_{kotač}}{r_{okvir}} = 57 * 3.77 * 1.27 = 273cm/s = 9.8 km/h$$

gdje su:

v – brzina,

O – opseg *scroll* kotača miša,

$mjer/s$ – broj mjerenja u sekundi,

$r_{kotač}$ – polumjer cijelog kotača,

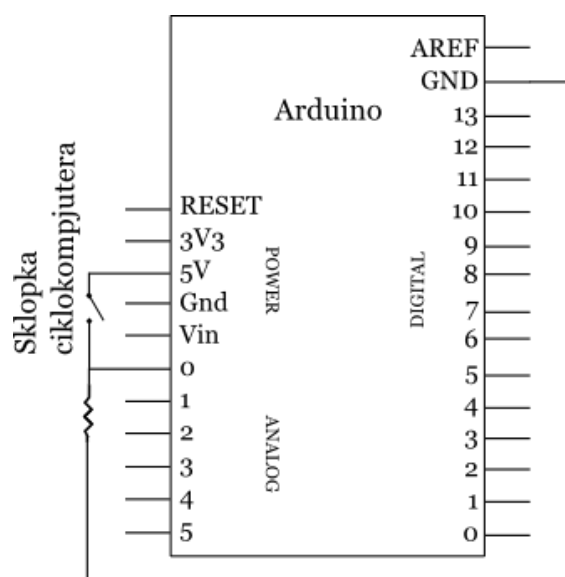
r_{okvir} – polumjer okvira kotača.

Iz izraza $mjer/s = \frac{v}{0/2}$, vidljivo je da veći opseg kotača omogućuje manji broj očitavanja mjerenja u sekundi tj. brže kretanje bicikla koje je uvjetovano ratom očitavanja miša (maksimalno 57 mjerenja u sekundi, uzevši u obzir razliku kotača i okvira).

Ideja je prisloniti *scroll* kotač miša na okvir kotača bicikla kako bi se odredilo kretanje bicikla tj. udaljenost između dvije točke.

6. 2. Mjerenje udaljenosti pomoću Arduino modula

Arduino modul je korišten za detektiranje vrtnje prednjeg kotača bicikla. To je izvedeno tako da se dio ciklokomputera koji se sastoji od sklopke, žica i magneta iskoristi umjesto postavljanja nove sklopke i magneta. Princip detektiranja vrtnje kotača (pomaka bicikla) je slijedeći: na žbicama kotača se nalaze 2 magneta koji prilikom prelaska zatvaraju sklopku koja je spojena žicama na Arduino modul. Sklopka je spojena tako da je jedna žica spojena na izvor napajanja 5V, a druga na analogni ulaz broj 0. Tako zapravo sklopka kad je zatvorena pušta napon od 5V na analogni ulaz. Analogni ulaz može poprimiti vrijednosti od 0 za 0 V do 1023 za 5 V. Da bi se dobile vrijednosti 0 (kad je sklopka otvorena) ili 1023 (kad je sklopka zatvorena) na analognom ulazu spojen je otpornik od 10 K Ω na uzemljenje i analogni ulaz broj 0. Shema je prikazana na slici 18.



Slika 18. Shema

Znači princip je jednostavan, neprestano se očitava vrijednost analognog ulaza i kada se vrijednost promjeni sa 0 na 1023 znači da se kotač zavrtio za vrijednost koja je definirana postavljanjem magneta na žbicu kotača (puni krug ili pola kruga). Testiranjem ovakvog

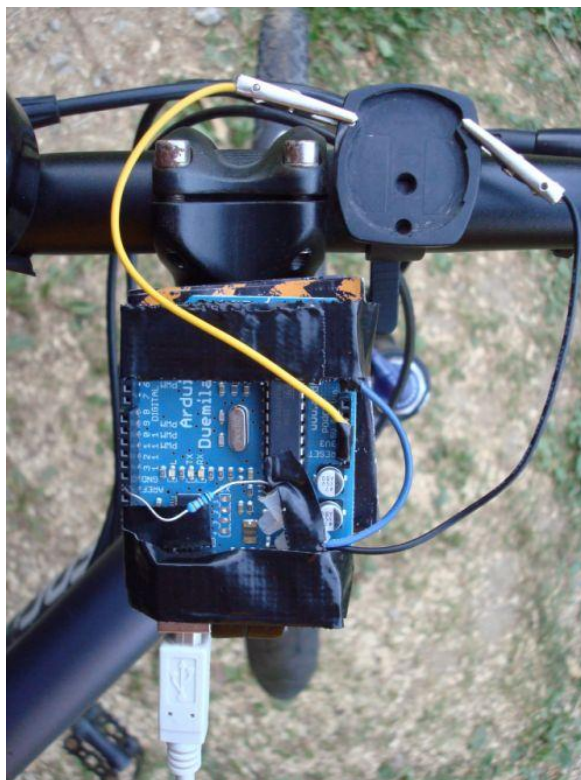
načina mjerenja udaljenosti otkriveno je da se na analognom ulazu prilikom prolaska magneta pored sklopke (zatvaranja sklopke) mogu očitati vrijednosti od 1021 do 1023. Ako se kotač vrti sporije, očitavanja mogu biti zbunjujuća jer zbog činjenice da je magnet duže vrijeme pored sklopke, sklopka duže vrijeme zatvorena pa zbog toga nekoliko uzastopnih vrijednost analognog ulaza može biti 1023 jer je sklopka zatvorena neko vrijeme (Slika 19).

Kako bi se iz dobivenih očitavanja dobila točna udaljenost (točan broj prelazaka magneta pored sklopke) stavljen je uvjet da je jedan prolaz kada se vrijednost analognog ulaza promijeni za vrijednost veću od 10.

U ovom radu su korištena 2 magneta na suprotnim stranama kotača, a pošto je opseg kotača bicikla približno 2m, to znači da je prilikom jednog zatvaranja sklopke bicikl prošao približno 1m. Ovaj način mjerenja udaljenosti daje najmanju gustoću točaka od 1m bez interpolacije između dviju točaka uzdužnog profila.

0, 18:0:8:375	0, 18:0:8:678
0, 18:0:8:390	0, 18:0:8:694
0, 18:0:8:406	0, 18:0:8:709
0, 18:0:8:422	1023, 18:0:8:726
0, 18:0:8:438	1023, 18:0:8:742
0, 18:0:8:454	0, 18:0:8:758
0, 18:0:8:470	0, 18:0:8:773
1021, 18:0:8:486	0, 18:0:8:790
1023, 18:0:8:502	1023, 18:0:8:806
1022, 18:0:8:518	0, 18:0:8:822
0, 18:0:8:534	0, 18:0:8:838
0, 18:0:8:550	1023, 18:0:8:854
0, 18:0:8:566	0, 18:0:8:870
0, 18:0:8:582	0, 18:0:8:886
0, 18:0:8:598	1023, 18:0:8:902
0, 18:0:8:614	0, 18:0:8:918
1021, 18:0:8:629	0, 18:0:8:934
1022, 18:0:8:646	1022, 18:0:8:949
0, 18:0:8:662	0, 18:0:8:965

Slika 19. Primjer vrijednosti analognog ulaza (vrijednost, vrijeme)



Slika 20. Spajanje Arduino modula na postolje za ciklokompjuter

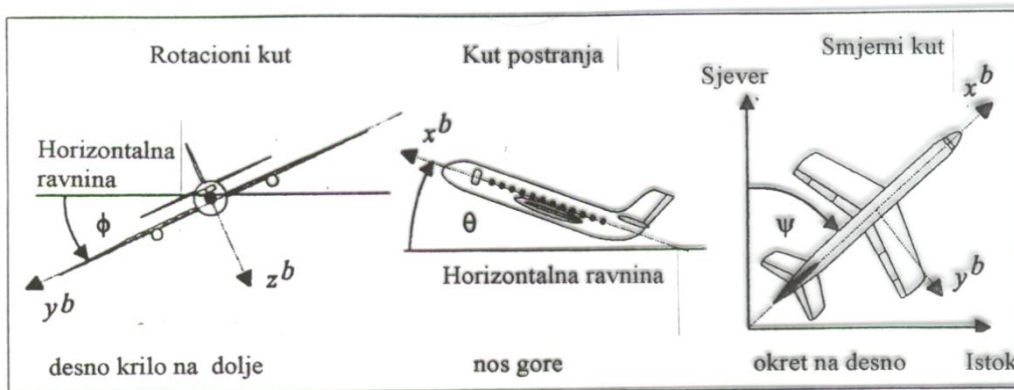
6. 3. Integracija mjerenja inercijalne mjerne jedinice

Inercijalna mjerna jedinica je korištena u integraciji radi mjerenja vertikalnog kuta koji se kasnije u računu koristi za određivanje visinske razlike između dvije točke. Među ostalim mjerenjima, jedinica daje set tzv. Eulerovih kuteva koji su kutevi fiksni uz tijelo Zemlje.

Prema specifikaciji proizvođača, Eulerovi kutevi su računati prema „ZYX“ ili „zrakoplovnom“ koordinatnom sustavu. Zemaljsko fiksni koordinatni sustav je definiran koordinatnim osima:

X os u smjeru sjevera,
Y os u smjeru istoka,
Z os u smjeru dolje.

Slika 21 prikazuje tzv. zrakoplovni koordinatni sustav i kuteve u njemu.



Slika 21. Objašnjenje koordinatnog sustava i kuteva (Bačić, Ž., ISuG_P7 2012/2013)

IMU jedinica daje mogućnost dobivanja filtriranih Euler kuteva, koji su i korišteni u ovom radu. Princip je korištenja algoritma komplementarnog filtera koji filtrira odstupanja zbog prijelaznog linearnog ubrzanja i zbog prijelaznih magnetskih smetnji. Proizvođač takve kuteve naziva žiro-stabilizirani Euler kutevi.

Kutevi ljuljanja i skretanja dobivaju se u intervalu od -32768 do +32767 koji predstavljaju interval od -180 do +180 stupnjeva. Kut posrtanja je u intervalu -16384 do +16383 koji predstavlja interval od -90 do +90 stupnjeva. Da bi se izlazni podatak pretvorio u stupnjeve potrebno ga je pomnožiti s faktorom 360/65536.

Kod mjerenja kuta korišten je način na zahtjev IMU jedinice radi smanjenja količine podataka. Sustav je postavljen tako da mjeri vertikalni kut svaki put kada bude mjerena udaljenost (svakih 0.049 m).

6. 4. Integracija senzora

Integracija senzora se sastoji od osam faza u kojima se realizira mjerni sustav korak po korak.

1.) Definirati (identificirati) referentni sustav i okvir senzora

Referentni sustav IMU jedinice:

- Zemaljsko fiksni koordinatni sustav

Referentni sustav odometra:

- Koordinatni sustav u ravnini

2.) Definirati referentni sustav i okvir mjernog sustava

Referentni sustav mjernog sustava:

- Zemaljsko fiksni koordinatni sustav (bilo koji sustav visina)

3.) Definirati (identificirati) referentnu točku (ili projekcijsko središte) senzora

Referentna točka IMU jedinice:

- Središte lokalnog koordinatnog sustava (centar koordinatnih osi X,Y, Z)

Referentna točka odometra:

- Središte *scroll* kotača

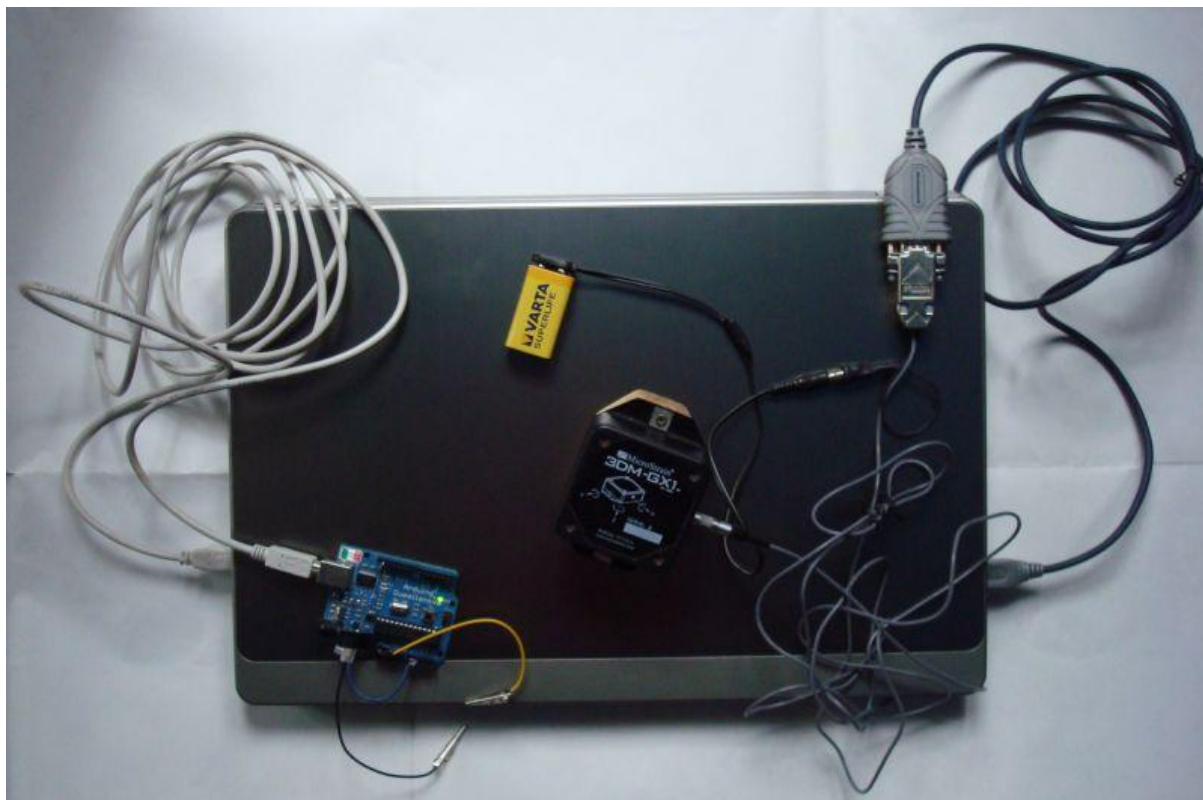
4.) Definirati referentnu točku (ili projekcijsko središte) mjernog sustava

Referentna točka mjernog sustava:

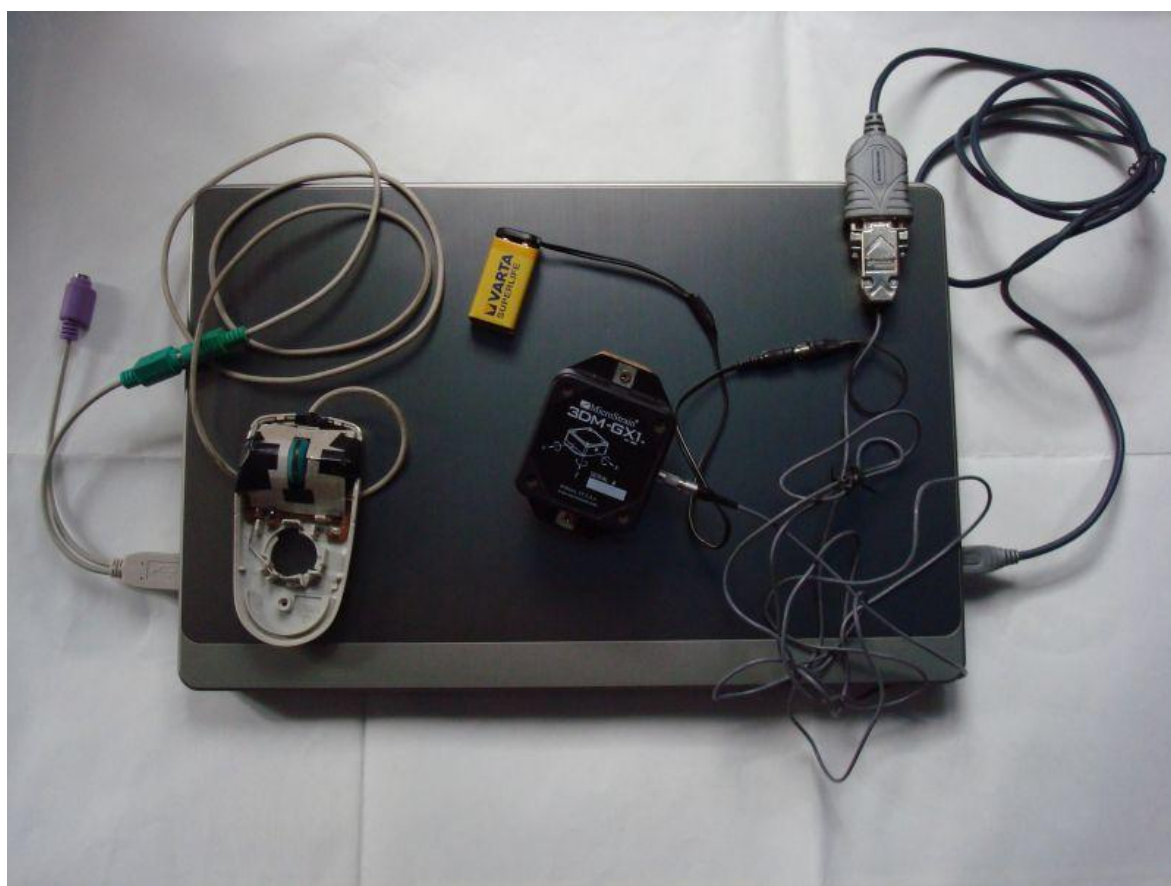
- Središte lokalnog koordinatnog sustava (centar koordinatnih osi X,Y, Z) IMU jedinice

5.) Povezivanje svih senzora u mrežu i osiguranje napajanja

Senzori su spojeni USB kabelom na prijenosno računalo (Slika 22 i 23) koje se napaja vlastitom baterijom. USB kabel šalje podatke iz senzora u prijenosno računalo i obrnuto, a ujedno i napaja senzore. IMU jedinca ima posebno napajanje preko 9 V baterije tipa E block, 6F22.



Slika 22. Prikaz spajanja senzora na prijenosno računalo (Arduino lijevo, IMU desno)



Slika 23. Prikaz spajanja senzora na prijenosno računalo (PS/2 miš lijevo, IMU desno)

6.) Definirati i sinkronizirati mjerenja (po sadržaju i u vremenu)

Za mjerenje razlike visina trigonometrijskim nivelmanom potreban nam je vertikalni kut i dužina. Vertikalni kut je dobiven mjerenjem kuta posrtanja IMU jedinice (eng. *pitch*), dok je udaljenost dobivena očitajima okretaja *scroll* kotača miša. Mjerenja su sinkronizirana pomoću sistemskog vremena koje se sprema prilikom spremanja podataka IMU jedinice i računalnog miša.

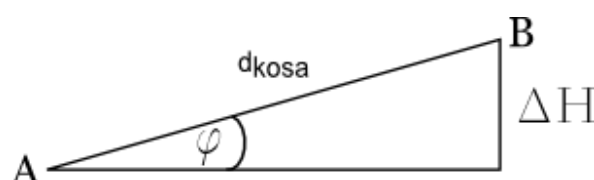
7.) Definirati matematički model mjernog sustava (algoritam, aplikacije)

Princip određivanja razlike visina iz mjerenog vertikalnog kuta i kose udaljenosti svodi se na matematičku formulu:

$$\Delta H = \sin \varphi * d_{\text{kosa}}, \text{ gdje su:}$$

φ - vertikalni kut i

d_{kosa} – kosa duljina



Slika 24. Princip određivanja visinske razlike

Izvorni kod aplikacije tj. skripta dan je u prilogima 1 i 2 Izvorni kod skripte za prikupljanje podataka mjernog sustava sa objašnjenjima i Izvorni kod skripte za obradu podataka mjernog sustava i prikaz rezultata sa objašnjenjima.

8.) Osiguranje adekvatnog računala i računalno povezivanje cijelog mjernog sustava

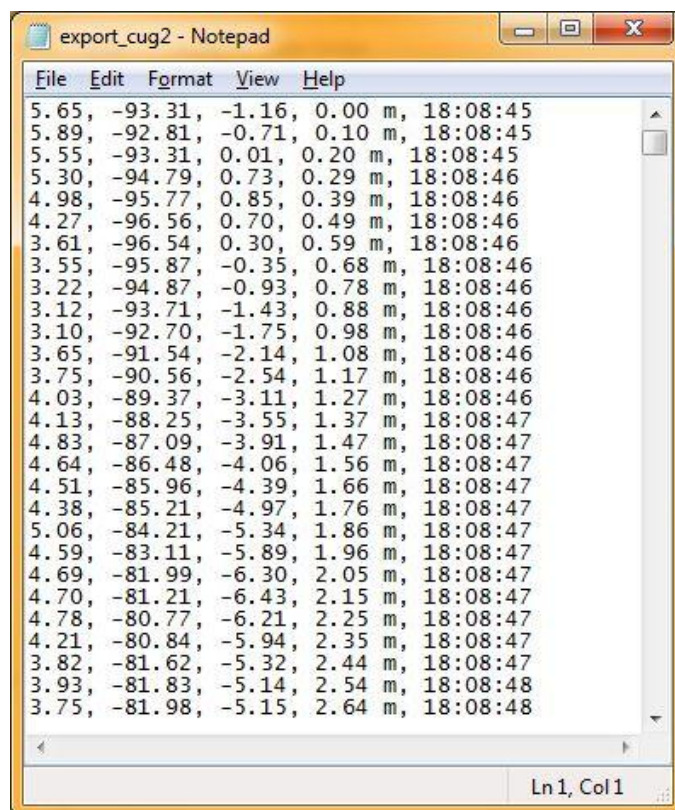
Svi senzori (računalni miš, IMU jedinica i Arduino modul) su spojeni na prijenosno računalo koje ima svoje napajanje preko baterije (vidi slike 22, 23). Prijenosno računalo je postavljeno u ruksak radi praktičnosti i kako bi se vozač mogao voziti na biciklu prilikom obavljanja mjerenja.

9.) Izrada aplikacije za prikupljanje mjerenja i obradu podataka

Predzadnji korak integracije je definicija matematičkog modela mjernog sustava (algoritam, aplikacije). Stoga je izađena aplikacija za prikupljanje i obradu podataka. Aplikacija se sastoji

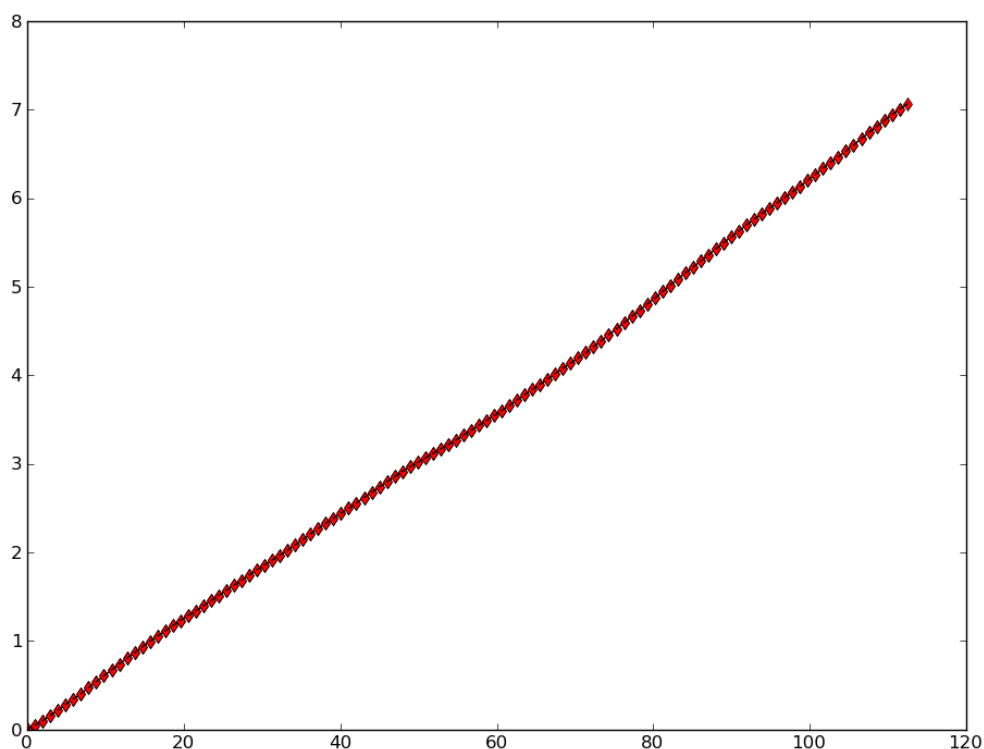
od dvije skripte u Python programskom jeziku. Prva skripta prikuplja mjerenja IMU jedinice i podatke miša, a druga služi za obradu mjerenja i prikaz rezultata.

Podaci prikupljeni sa senzora su spremeni u tekstualnu datoteku na lokalnom disku. Na slici 25 vidi se struktura spremljenih podataka, a redoslijed je slijedeći: rotacijski kut, kut posrtanja, smjerni kut, udaljenost, sistemsko vrijeme.



Slika 25. Spremljeni podaci mjerenja

Druga skripta obrađuje prikupljena mjerenja i iz njih računa i prikazuje uzdužni profil prijedene trase. Slika 26 prikazuje rezultat koji prikazuje Python skripta za drugu lokaciju. Na grafu horizontalna os prikazuje udaljenost, a vertikalna os prikazuje visinu (početna visina je postavljena na 0 m).



Slika 26. Prikaz rezultata Python skripte

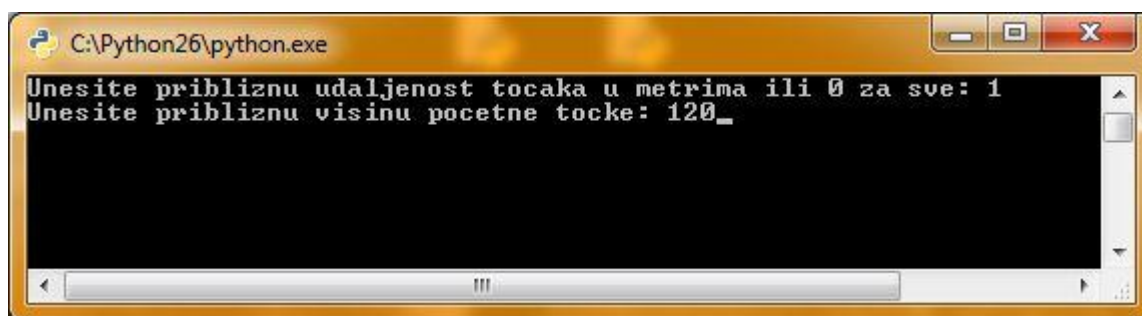
Pošto se skripta koja prikuplja podatke Arduino modula pokazala nekorisna izostavljena je iz ovog rada. Izvorni kod korištenih Python skripti dan je u prilogima 1 i 2.

6. 5. Programiranje u Python-u

Python je odabran iz razloga što sam već upoznat sa programiranjem u tom jeziku i jednostavniji je od programskog jezika C u kojem su dani primjeri aplikacija proizvođača IMU jedinice (URL 7). Ti primjeri su korišteni za "prevođenje" funkcija dobivanja podataka IMU jedinice iz C jezika u Python. U Python jeziku se za programiranje koriste tzv. moduli koji su zapravo datoteke sa spremljenim definicijama i izvorom funkcija. Za korištenje matematičkih i drugih operacija, koriste se tzv. biblioteke.

Za potrebe spremanja i obrade mjerenja senzora izrađene su dvije skripte. Prva prikuplja podatke i sprema ih na lokalni disk u tekstualnu datoteku. Ta skripta zapravo očitava pomak miša i za svaki takav događaj šalje naredbu mikrokontroleru IMU-a da pošalje kuteve orijentacije, koje kasnije zajedno sa vremenom i zbrajanom udaljenosti od početne točke sprema u unaprijed kreiranu tekstualnu datoteku. U drugoj se skripti spremljeni podaci

učitavaju i obrađuju, te se prikazuje rezultat tj. uzdužni profil prijeđene rute. Na početku izvršenja skripte moguće je odabrati željenu gustoću profila koja se želi prikazati na ekranu i visina početne točke od koje se počelo mjeriti (Slika 27).



Slika 27. Odabir prikaza rezultata

Izvorni kodovi sa objašnjenjima izrađenih Python skripta se nalaze u prilogima 1 i 2.

6. 6. Montiranje senzora na bicikl i obavljanje mjerenja

Mjerni sustav je realiziran montiranjem senzora na brdski bicikl, veličine kotača 26". IMU jedinica je montirana na volan, a računalni miš na vilicu bicikla.

6. 6. 1. Montiranje senzora

Kako bi se IMU jedinica mogla montirati na volan bicikla izrađena je drvena platforma koja je pričvršćena na nosač zadnjeg svjetla za bicikl. To omogućuje olakšano precizno horizontaliranje IMU jedinice.



Slika 28. Montirana IMU jedinica

Računalni miš je montiran na vijak koji je montiran na nosač vilice za prednju kočnicu. Kako bi se osiguralo neprestano prisanjanje *scroll* kotača za okvir kotača bicikla, postavljena je gumica koja pritiskuje *scroll* kotač na okvir.



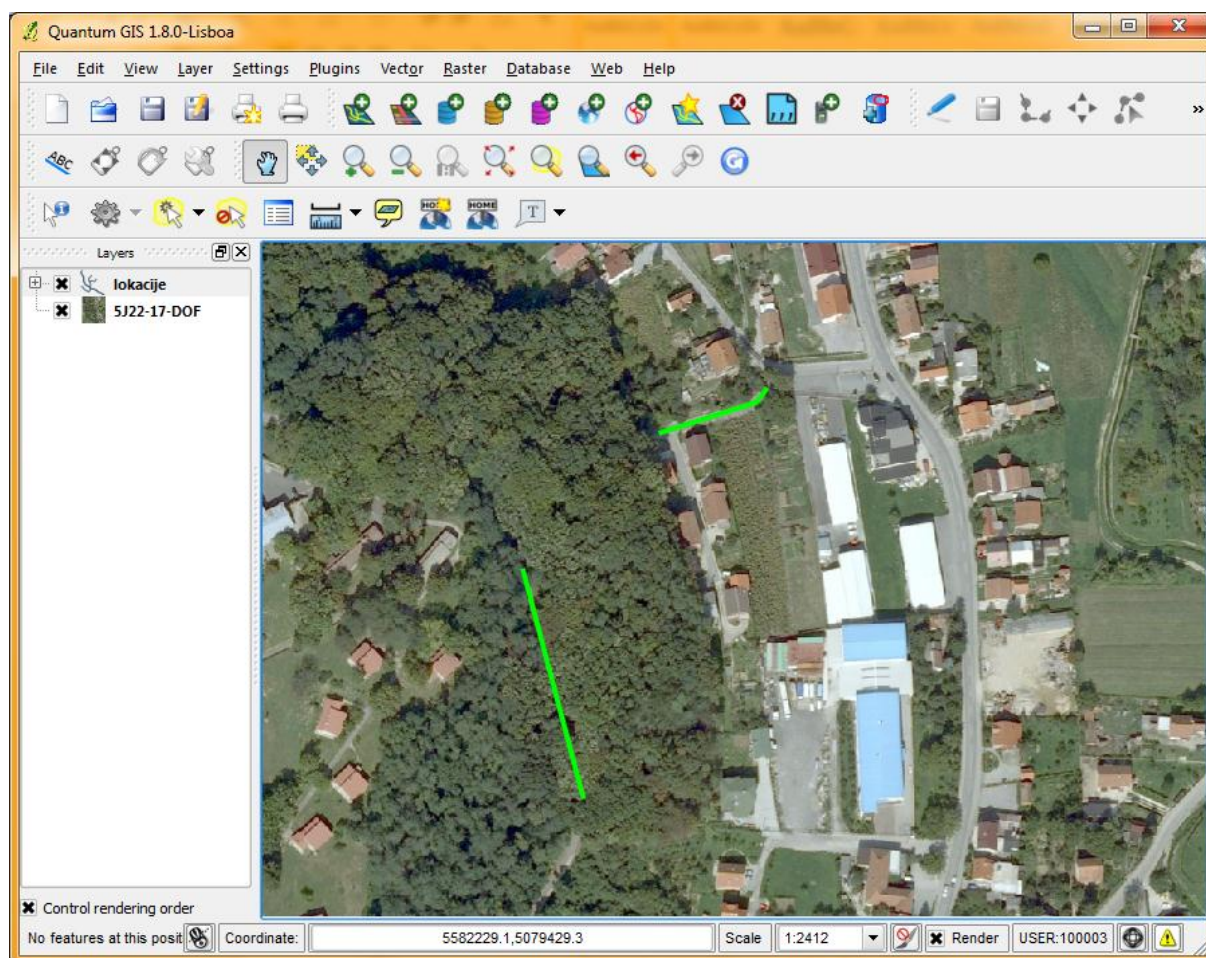
Slika 29. Montiran računalni miš

Senzori su spojeni na prijenosno računalo, koje je zajedno sa kablovima stavljeno u ruksak kako bi se vozač bicikla mogao nesmetano voziti tj. imati slobodne ruke.

6. 6. 2. Mjerenje pomoću integriranih senzora

Za obavljanje mjerenja odabrane su dvije lokacije, jedna na kojoj je cesta više ravniija, a druga na kojoj je cesta konstantnog uspona (Slika 30). Prva mjerena ruta je dužine 67 m, a druga 107 m.

Obje rute su izmjere geometrijskim nivelmanom koristeći visokoprecizni digitalni nivelir koji s fiberglas letvom daje točnost razlike visine od 1 mm. Usporedba rezultata dobivenih integriranjem senzora i geometrijskim nivelmanom dana je u nastavku.



Slika 30. Prikaz lokacija u QGIS-u

Nakon montiranja senzora na bicikl, potrebno je bicikl postaviti u horizontalnu ravninu kako bi horizontirana IMU jedinica doista mjerila vertikalni kut određen kretnjom bicikla. Bicikl je postavljen na drvenu dasku koja je horizontirana pomoću precizne libele i pomoću IMU jedinice koristeći aplikaciju koja dolazi s IMU jedinicom (3DM-GX1 aplikacija). Tako je ujedno i obavljena provjera kalibracije IMU jedinice.

7. Diskusija

Prednosti prilikom mjerenja uzdužnog profila po principima i algoritmima trigonometrijskog nivelmana integriranim senzorima, IMU jedinicom i računalnim mišem, su eliminacije utjecaja koji uzrokuju odstupanja kao što su :

- utjecaj refrakcije,
- utjecaj zakrivljenosti Zemlje.

No kao što postoje prednosti, postoje i mane, koje su:

- ograničenje maksimalne brzine kretanja prilikom mjerenja,
- niska točnost.

7. 1. Prenamjena računalnog miša u odometar

Prenamjena računalnog miša u odometar nije bio lagan posao. Računalni miševi bili oni laserski, optički ili mehanički nisu se pokazali kao najbolji kandidat za potrebe mjerenja duljine jer su namjenjeni mjerenju manjih udaljenosti u vrlo kratkom vremenu. Ta velika osjetljivost miša je ograničavajući faktor brzine kretanja platforme na koju je miš montiran. Unatoč mogućnosti postavljanja rate pošiljanja podataka miša, pokazalo se da se rata *scroll* kotača ne može podići na ponuđene frekvencije 125, 250, 500 i 1000 Hz, nego ostaje na otprilike 60 Hz. Kod operativnog sustava windows 7 općenito je otežano promijeniti ratu miša, pa sam s toga probao promijeniti ratu i u virtualnom windows XP sustavu, ali bezuspješno. Zbog principa određivanja pomaka miša, napravljena maskica u jednom okretu prouzroči dva očitavanja pomaka, što predstavlja i najmanji mogući broj očitavanja. To uvjetuje maksimalno kretanje bicikla brzinom nešto manjom od 10 km/h, što bi trebalo podići u novim istaživanjima, jer je prosječna brzina vožnje bicikla po ne brdovitoj cesti oko 20 km/h.

7. 2. Rezultati mjerenja udaljenosti pomoću Arduino modula

Nakon više od desetak obavljenih mjerenja došao sam do zaključka da Arduino ima najvišu ratu od 62 Hz, što je dobiveno razlikom vremena dvaju podataka koje Arduino daje računalu. Pokušavao sam podići ratu, ali bezuspješno. Kada se port namjesti na komunikaciju od 9600 bita po sekundi, sve radi kako spada, ali kada broj nije 9600, ne dobivaju se podaci u dobrom vremenu.

Mjerena je bila trasa od 900 m, a podaci dobiveni mjerenjem su bili 206 m kad sam koristio otpornik i 773 m kada nisam koristio otpornik. Problem kod interpretacije podataka kada se ne koristi otpornik je u tome što vrijednost analognog ulaza kada je sklopka zatvorena varira

od vrijednosti 1000 do 1023. To onemogućava postavljanje egzaktnog uvjeta za dobivanje udaljenosti. Isprobani su uvjeti: razlika vrijednosti veća od 4 i veća od 5, ali bezuspješno.

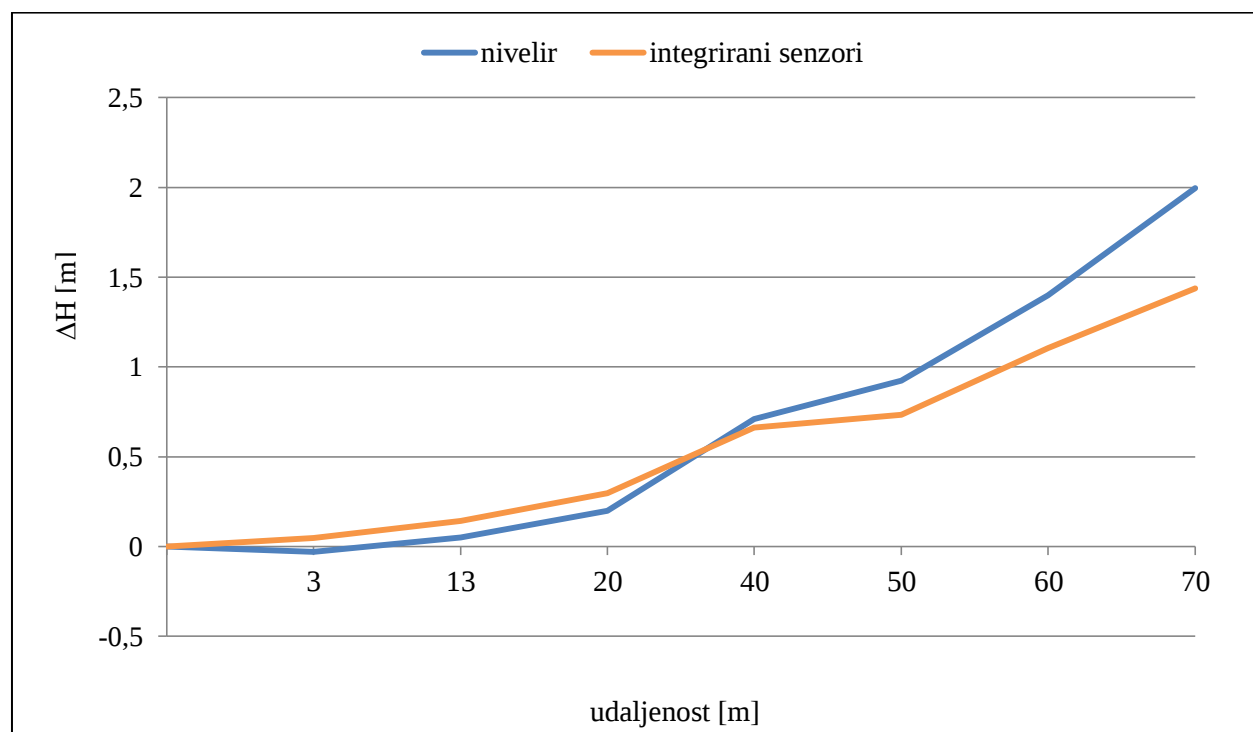
Pošto su podaci dobiveni korištenjem Arduino modula pokazali loše rezultate, odustao sam od takvog načina mjerenja udaljenosti i vratio se na mjerenje pomoću PS/2 miša.

7. 3. Rezultati mjerenja integracijom senzora

Slijede rezultati dobiveni mjerenjem integriranih senzora i mjerenja dobivenih geometrijskim nivelmanom na dvije lokacije. Podaci vezani za prvu lokaciju:

- duljina: 67 m,
- minimalna brzina kretanja: 1 km/h,
- maksimalna brzina kretanja: 5 km/h,
- prosječna brzina kretanja: 4 km/h,
- trajanje mjerenja: 47 sekundi.

Usporedba rezultata mjerenja na prvoj lokaciji dana je u nastavku.



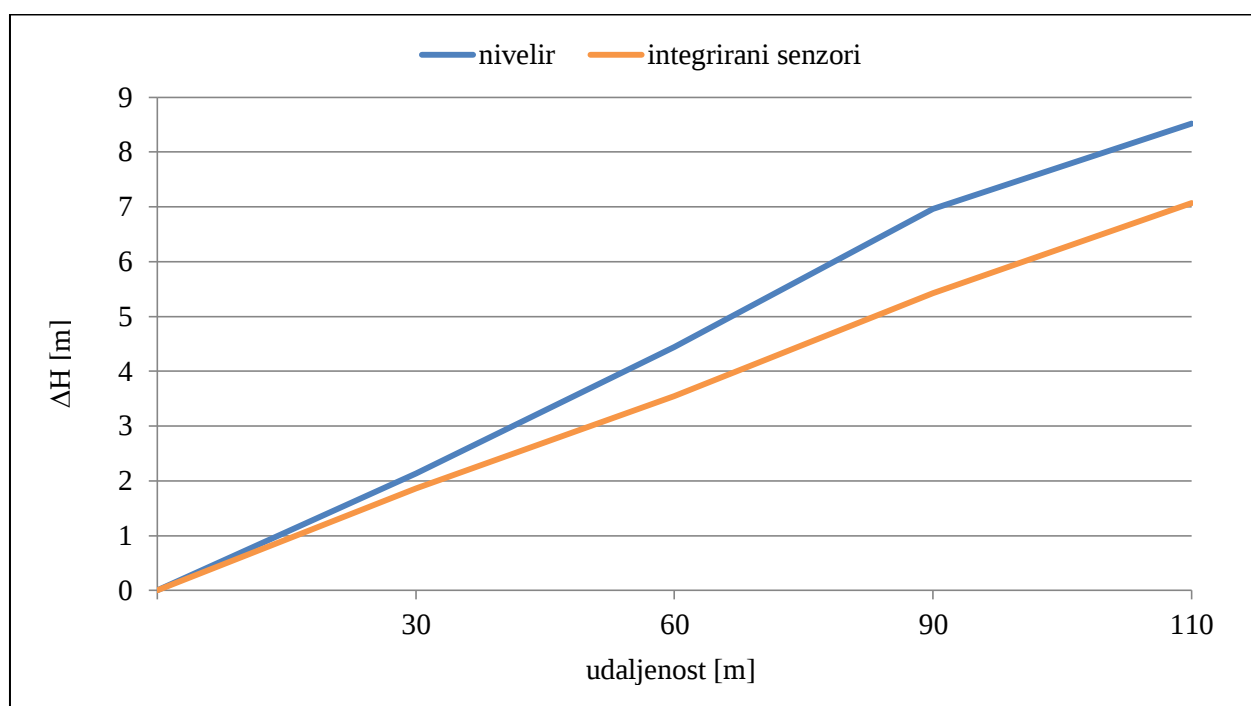
Grafikon 1. Usporedba mjerenja na prvoj lokaciji

Podaci vezani za drugu lokaciju:

- duljina: 107 m,

- minimalna brzina kretanja: 1 km/h,
- maksimalna brzina kretanja: 6 km/h,
- prosječna brzina kretanja: 5 km/h,
- trajanje mjerenja: 2 minute i 17 sekundi.

Usporedba rezultata mjerenja na drugoj lokaciji dana je u nastavku.



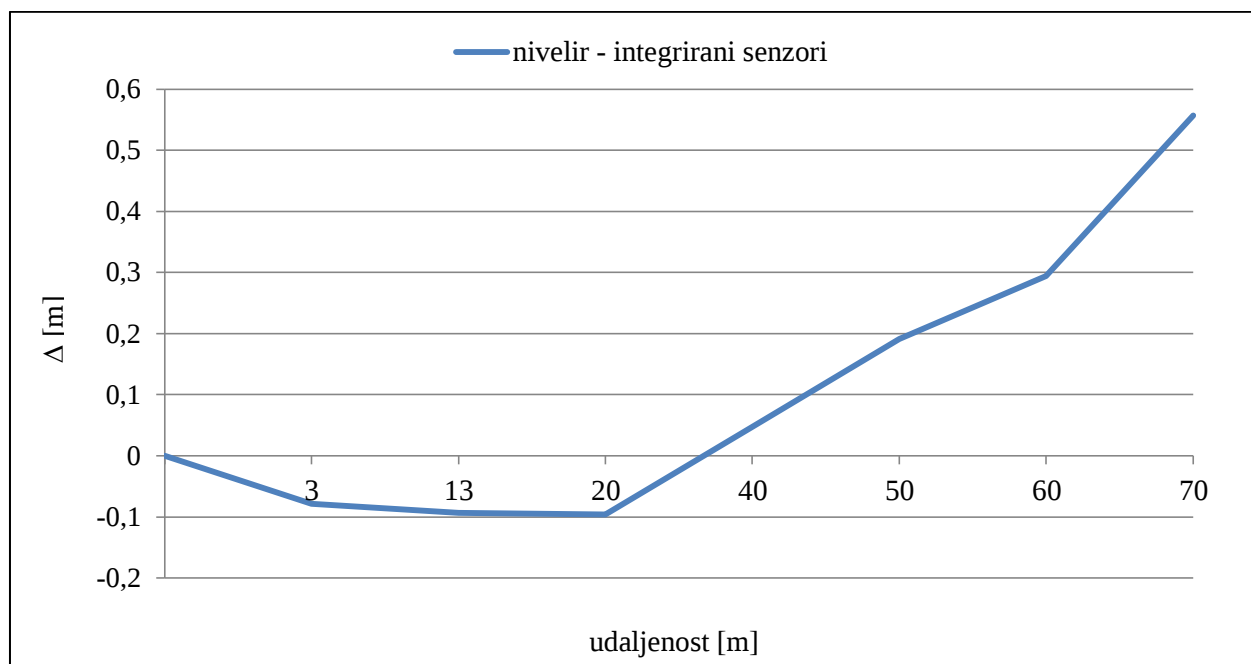
Grafikon 2. Usporedba mjerenja na drugoj lokaciji

Poznato je da je točnost mjerenja IMU jedinicom ovisna o vremenu, pa zbog toga ona opada sa povećanjem udaljenosti (jer kretanje kroz prostor zahtijeva vrijeme). Kod mjerenja razlike visina odstupanje je vezano uz duljinu trase koja se nivelira (npr. 2mm na 1km). Zbog toga je razlika mjerenja nivelira i integriranih senzora po udaljenosti dana na idućim grafikonima. Razlika je računata po principu: razlika = nivelir - integrirani senzori.

Tablica 5. Razlika na prvoj lokaciji

Udaljenost [m]	Razlika (nivelir – integrirani senzori) [m]
3	-0.0782
13	-0.0934
20	-0.0964
40	0.0474
50	0.1911
60	0.2941

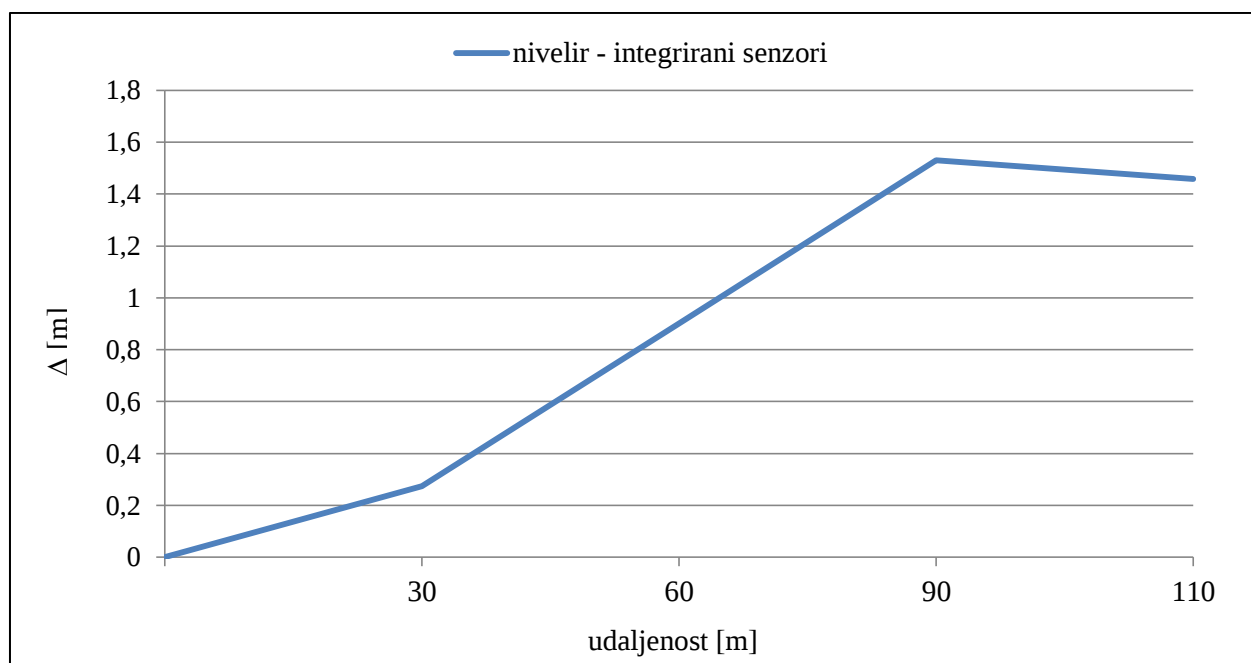
70	0.5568
----	--------



Grafikon 3. Razlika, prva lokacija

Tablica 6. Razlika na drugoj lokaciji

Udaljenost [m]	Razlika (nivelir – integrirani senzori) [m]
30	0.2735
60	0.9002
90	1.5304
107	1.458



Grafikon 4. Razlika, druga lokacija

Realiziranim mjernim sustavom pokazalo se da se nadolazeća tehnologija pristupačne cijene još ne može koristiti za precizna mjerenja uzdužnog profila trase. Takav mjerni sustav nema primjene u geodetskim mjerenjima, ali može se koristiti za približna mjerenja. Ovim istraživanjem je utvrđeno da se odstupanje povećava sa vremenom, i da na relativno malim udaljenostima (do 100 m) iznose do 1.5 m. Kod mjerenja ravnije trase, dobiveno je odstupanje od 0.56 m na udaljenosti od 60 m, a kod trase konstantnog uspona odstupanje od 1.5 m na udaljenosti od 107 m. Možda je dio odstupanja sistematska pogreška koja se može otkloniti u kodu aplikacije nakon obavljene kalibracije sustava. Dobiveni rezultati pokazuju da trenutno inercijalni senzori pristupačne cijene ne mogu zamijeniti geodetske instrumente koji se koriste za mjerenje razlike visina. Sa daljnim razvojem senzora i padom cijena istih, vjerojatno će se uskoro dobivati bolji rezultati koristeći senzore pristupačne cijene. Glavno ograničenje izrađenog mjernog sustava montiranog na bicikl je maksimalna brzina kretanja, koja iznosi 9.8 km/h. Brzina bi se mogla povećati zamjenom računalnog miša sa nekim prikladnijim odometrom. U ovom sam radu pokušao izraditi odometar koristeći Arduino modul i dijelove ciklokompjutera, ali nisam dobio zadovoljavajuće rezultate. Možda su korišteni dijelovi ciklokompjutera sadržavali elemente koji su ometali rad Arduina. Nisam uspio detaljnije istražiti ovu problematiku.

U vrijeme pisanja ovog rada, postoje na tržištu odometri namijenjeni za mjerenje brzine kretanja bicikla koji šalju podatke preko *bluetooth* veze. U daljnim istraživanjima bilo bi

zanimljivo isprobati integraciju IMU mjerne jedinice i takvog praktičnog odometra koji je prikladniji za mjerenje udaljenosti od računalnog miša. Ti odometri imaju primarnu namjenu spajanja na smartphone ili tablet uređaje, što je povoljna činjenica, jer ako bi se prijenosno računalo zamijenilo tabletom ili smartphoneom, sustav kakav je realiziran za potrebe ovog rada bio bi puno praktičniji. Većina današnjih smartphone i tablet uređaja posjeduje senzore kao što su akcelerometar, žiroskop i kompas koji su trenutno niske točnosti. Vjerojatno će uskoro ta točnost rasti, a to će biti prilika za realizaciju puno praktičnijeg mjernog sustava integracijom IMU jedinice i odometra. Jer ako platforma sama sadrži potrebne senzore za mjerenje vertikalnog kuta, a moguće je bežično spajanje odometra, takav sustav bi bilo puno lakše montirati na bicikl. Time bi se eliminirala i smetnja žica koje spajaju senzore i računalo prilikom vožnje bicikla.

Maksimalna brzina koju izrađeni mjerni sustav nudi je 9.8 km/h, što je puno brže od postupka niveliranja koristeći nivelir (geometrijski nivelman) ili totalnu mjernu stanicu (trigonometrijski nivelman). Kada bi se podigla točnost ovakvog mjerenog sustava, zasigurno bi se mogao koristiti u geodetskoj praksi jer nudi dovoljno prednosti.

8. Zaključak

U ovom radu istražena je integracija inercijalne mjerne jedinice i računalnog miša. Senzori su montirani na brdski bicikl, te su korišteni za mjerenje vertikalnog kuta i duljine. Mjerenja senzora korištena su za izračun uzdužnog visinskog profila ceste po kojoj se bicikl kreće. Uzdužni visinski profil je računat prema pravilima i algoritmima trigonometrijskog nivelmana.

IMU jedinica je pokazala dobre rezultate u odnosu na cjenovni rang, dok se računalni miš nije pokazao kao najbolji odometar. Računalni miš ograničava maksimalnu brzinu kretanja bicikla na 9.8 km/h. Rezultati dobiveni mjerenjima integriranim senzorima su uspoređeni s mjerenjima visokopreciznog digitalnog nivelira. Usporedba je pokazala da inercijalni senzori pristupačne cijene, koji se danas nude na tržištu, ne mogu zamijeniti precizne geodetske instrumente u geodetskim zadaćama. Usporedbom je određeno odstupanje od 0.56 m na ravnoj cesti, pri duljini od 60 m i odstupanje od 1.46 m na cesti konstantnog uspona, pri duljini od 107 m.

Inercijalna mjerna jedinica (IMU) se pokazala kao praktično rješenje za mjerenje kuteva orijentacije platforme na koju je montirana, pa će zasigurno kada visokoprecizni primjerci budu pristupačne cijene imati veću primjenu u geodeziji, a vjerojatno će imati i primjenu u nekim zadaćama u kojima se trenutno ne koriste.

Popis kratica i stranih riječi

Kratica	Puni izvorni naziv	Prijevod i značenje	Str
IMU	Inertial Measurement Unit	Inercijalna mjerna jedinica	1
GNSS	Global Navigation Satellite System	Globalni navigacijski satelitski sustav	1
MEMS	Microelectromechanical Systems	Vrlo maleni elektomehanički sustavi	6
PWM	Pulse-width modulation	Modulacija širine impulsa	12
ICSP	In-circuit serial programming	Upetljano serijsko programiranje	12
LED	Light Emiting Diode	Dioda koja emitira svjetlo	12
USB	Universal Serial Port	Univerzalni serijski <i>port</i>	12
SRAM	Static random-access memory	Statična radna memorija	13
EPROM	Erasable Programmable Read Only Memory	Izbrisiva programibilna nepromjenjiva memorija	13
OS	Opearting System	Operativni sustav	16

Literatura

Bačić, Ž. (2012.): Predavanja iz kolegija Integrirani sustavi u geomatici, Zagreb.

Bačić, T. (2010.): Predavanja iz kolegija Državna izmjera, Zagreb.

Džapo, M. (2008.): Predavanja iz kolegija Izmjera zemljišta.

Popis URL-ova

URL 1 Hights, http://www.esri.com/news/arcuser/0703/graphics/geoid2_lg.gif (27.3.2013.)

URL 2 4QD: Mice: how they work, <http://www.4qdtex.com/meece.html> (27.3.2013.)

URL 3 How mechanical mouse works? - Another Computers Blog,
<http://akomaenablog.blogspot.com/2011/09/how-mechanical-mouse-works.html> (27.3.2013.)

URL 4 How to Increase USB Sample Rate in Windows Vista/7,
<http://www.ngohq.com/news/15043-how-to-increase-usb-sample-rate-in-windows-vista-7-a.html> (27.3.2013.)

URL 5 IMU Odometry, by David Anderson, http://geology.heroy.smu.edu/~dpa-www/robo/Encoder/imu_odo/index_IE.htm#sec6 (27.3.2013.)

URL 6 Photo Gallery,
https://webfiles.uci.edu/cagell/Dreambot/photo_gallery/photo_gallery.htm (27.3.2013.)

URL 7 3DM-GX1®, <http://www.microstrain.com/inertial/3DM-GX1>, (27.3.2013.)

URL 8 Module pythoncom, <http://timgolden.me.uk/pywin32-docs/pythoncom.html> (27.3.2013.)

URL 9 SourceForge.net: PyHook Tutorial – pyhook,
http://sourceforge.net/apps/mediawiki/pyhook/index.php?title=PyHook_Tutorial#tocpyHook_Tutorial4 (27.3.2013.)

URL 10 Welcome to pySerial's documentation — pySerial v2.6 documentation,
<http://pyserial.sourceforge.net/> (27.3.2013.)

URL 11 7.3. struct — Interpret strings as packed binary data — Python v2.7.5 documentation, <http://docs.python.org/2/library/struct.html> (27.3.2013.)

URL 12 8.1. datetime — Basic date and time types — Python v2.7.5 documentation, <http://docs.python.org/2/library/datetime.html> (27.3.2013.)

URL 13 9.2. math — Mathematical functions — Python v2.7.5 documentation, <http://docs.python.org/2/library/math.html?highlight=math#math> (27.3.2013.)

URL 14 Python for Windows extensions | Free software downloads at SourceForge.net, <http://sourceforge.net/projects/pywin32/> (27.3.2013.)

URL 15 Win32api, <http://docs.activestate.com/activepython/2.4/pywin32/win32api.html> (27.3.2013.)

URL 16 34.2. msvcrt – Useful routines from the MS VC++ runtime — Python v2.7.5 documentation, <http://docs.python.org/2/library/msvcrt.html> (2.8.2013.)

URL 17 Arduino Playground – Python, <http://playground.arduino.cc/interfacing/python> (2.8.2013.)

URL 18 Shapely 1.2.18 : Python Package Index, <https://pypi.python.org/pypi/Shapely> (2.8.2013.)

URL 19 Matplotlib 1.3.0 : Python Package Index, <https://pypi.python.org/pypi/matplotlib> (2.8.2013.)

URL 20 Arduino Bike Speedometer, <http://www.instructables.com/id/Arduino-Bike-Speedometer/> (30.5.2013.)

URL 21 SDL30 Digital Level | SOKKIA USA, <http://us.sokkia.com/products/levels-accessories/automatic-levels/sdl30-digital-level> (27.8.2013.)

URL 22 Arduino – ArduinoBoardDuemilanove, <http://arduino.cc/en/Main/arduinoBoardDuemilanove> (27.8.2013.)

Popis grafikona

Grafikon 1. Usporedba mjerenja na prvoj lokaciji	34
Grafikon 2. Usporedba mjerenja na drugoj lokaciji	35
Grafikon 3. Razlika, prva lokacija.....	36
Grafikon 4. Razlika, druga lokacija.....	37

Popis slika

Slika 1. Robot IMU odometry (URL 5)	2
Slika 2. Robot Dreambot Project (URL 6)	3
Slika 3. Geometrijski nivelman (Bašić, T. DI 09, 2010/2011).....	5
Slika 4. Trigonometrijski nivelman (Bašić, T., DI 09, 2010/2011)	6
Slika 5. Prikaz crijeva napunjenog vodom (Bašić, T. DI 09, 2010/2011).....	7
Slika 6. Veza između elipsoidne i ortometrijske visine (URL 1).....	7
Slika 7. Princip rada jedinice (URL 7)	9
Slika 8. Stabilna platforma	10
Slika 9. Sučelje aplikacije 3DM-GX1	11
Slika 10. Prikaz enkodera miša	12
Slika 11. Arduino (URL 22).....	13
Slika 12. Nivelir Slika 13. Nivelmanska letva	15
Slika 14. Prikaz konstrukcije korištenog miša	17
Slika 15. Izrađena maskica i postojeći kotač.....	18
Slika 16. Zatvoren dio za očitavanje okretanja <i>scroll</i> kotača miša.....	19
Slika 17. Razlika polumjera obruča i kotača.....	19
Slika 18. Shema.....	21
Slika 19. Primjer vrijednosti analognog ulaza (vrijednost, vrijeme).....	22
Slika 20. Spajanje Arduino modula na postolje za ciklokomputer	23
Slika 21. Objašnjenje koordinatnog sustava i kuteva (Bašić, Ž., ISuG_P7 2012/2013).....	24
Slika 22. Prikaz spajanja senzora na prijenosno računalo (Arduino lijevo, IMU desno).....	26
Slika 23. Prikaz spajanja senzora na prijenosno računalo (PS/2 miš lijevo, IMU desno).....	26
Slika 24. Princip određivanja visinske razlike	27
Slika 25. Spremljeni podaci mjerenja.....	28
Slika 26. Prikaz rezultata Python skripte.....	29
Slika 27. Odabir prikaza rezultata	30
Slika 28. Montirana IMU jedinica.....	30
Slika 29. Montiran računalni miš	31
Slika 30. Prikaz lokacija u QGIS-u	32

Popis tablica

Tablica 1. Vrijednosti člana c_1 ovisno o udaljenosti d (Izmjera zemljišta, Džapo, M., 2008)...	6
Tablica 2. Specifikacije inercijalne mjerne jedinice (URL 7).....	8
Tablica 3. Specifikacije Arduina (URL 22):	13
Tablica 4. Specifikacije digitalnog nivelira (URL 21)	15
Tablica 5. Razlika na prvoj lokaciji.....	35
Tablica 6. Razlika na drugoj lokaciji.....	36

Prilog 1 Izvorni kod skripte za prikupljanje podataka mjernog sustava sa objašnjenjima

```

import pythoncom, pyHook
import win32api
import win32con
import serial
import struct
from time import localtime, strftime
from sys import exit                                ## - učitavanje
potrebnih modula

i = -1                                              ## -1 jer želim dobiti
prvo očitavanje miša kao 0-to mjerenje
udalj = -0.04889                                  ## - da se prvim očitanjem
pomaka kotača miša dobije udaljenost 0
convertFactor = (360.0/65536.0)                  ## - faktor kojim množimo IMU
podatke da dobijemo kuteve u stupnjevima
main_thread_id = win32api.GetCurrentThreadId()
## - uzimanje trenutnog tread-a

ser = serial.Serial(0, 38400, timeout=1, dsrdtr=True)
## - otvaranje serijskog porta (port 1)
datoteka = open('lokacija\\datoteka.txt', 'w')
## - otvaranje txt datoteke

def OnMouseEvent(event):                          ## - funkcija koja će
prikupljati podatke (Euler kuteve) svaki put kada se pomakne kotač miša
    global i
    global main_thread_id
    global udalj

    i = i+1
    udalj = udalj + 0.04889

    if(i%2==0):                                  ## - pošto imamo 2 mjerenja u jednom okretaju
    (punom krugu) kotača miša, da bismo dobili podatke IMU-a uzimamo podatke
    samo jednom u krugu
        ser.write(unichr(0x0E))                  ## - šaljem
mikrokontroleru IMU-a naredbu za slanje Euler kuteva
        odgovor = ser.read(11)                   ## - čitanje odgovora
mikrokontrolera (11 je broj bytova)
        odg_c = struct.unpack("11c", odgovor);   ## - pretvaranje C tipa
podataka (bytova) u čitljive podatke u Pythonu (char)
        a = ord(odg_c[1])*256 + ord(odg_c[2])    ## - pretvaranje char
podataka u int prema uputama proizvođača IMU-a za rotacioni kut
        b = ord(odg_c[3])*256 + ord(odg_c[4])    ## - pretvaranje char
podataka u int prema uputama proizvođača IMU-a za kut posrtanja
        c = ord(odg_c[5])*256 + ord(odg_c[6])    ## - pretvaranje char
podataka u int prema uputama proizvođača IMU-a za smjerni kut

        roll = a * convertFactor                 ## - spremanje dobivenih
kuteva u varijable uz zadovoljavanje uvjeta
        if(roll>180):
            roll = roll-360

```

```

pitch = b * convertFactor
if(pitch>90):
    pitch = pitch - 360

yaw = c * convertFactor
if(yaw>180):
    yaw = yaw - 360

    datoteka.write("%.2f" % pitch + ", " + "%.2f" % yaw + ", " + "%.2f"
% roll + ", " + "%.2f" % udalj + ' m, ' + strftime("%H:%M:%S", localtime())
+ "\n")    ## - spremnaje podataka u txt datoteku

    return udalj                                ## - vraćanje vrijednosti
globalne varijable udaljenosti
    return True                                ## - vraćanje vrijednosti
True (zahtjev modula pyHook)

def OnKeyboardEvent(event):                    ## - funkcija za prekidanje
skripte pritiskom na bilo koju tipku tipkovnice
    global main_thread_id
    global ser
    global datoteka
    win32api.PostThreadMessage(main_thread_id, win32con.WM_QUIT, 0, 0); ##
- naredba za prekidanje slanja poruka sustava - PumpMessages()
    ser.close()                                ## - zatvaranje
porta
    datoteka.close()                           ## - zatvaranje
datoteke
    print ('Prijedjeno je: ' + "%.2f" % udalj + 'metara') ## - ispisivanje
poruke o prijedenoj udaljenosti
    print 'Završeno !!!'                        ## - ispisivanje
poruke da je skripta završena
    return True                                ## - vraćanje
True vrijednosti (zahtjev modula pyHook)
    exit()                                    ## - izlaz iz
Python skripte

hm = pyHook.HookManager()                      ## - manager za korištenje
unosa miša i tipkovnice
hm.MouseWheel = OnMouseEvent                   ## - pokretanje funkcije
OnMouseEvent kada se pokrene Scroll kotač miša
hm.HookMouse()                                ## - povezivanje na miša
hm.KeyDown = OnKeyboardEvent                  ## - pozivanje funkcije OnKeyboardEvent
kada se pritisne bilo koja tipka tipkovnice
hm.HookKeyboard()                             ## - povezivanje na tipkovnicu

pythoncom.PumpMessages()                      ## - naredba koja vraća
poruke sustava (pritisak tipke ili pomak miša)

```

Prilog 2 Izvorni kod skripte za obradu podataka mjernog sustava i prikaz rezultata sa objašnjenjima

```

from shapely.geometry import Point, LineString
from matplotlib import pyplot
import math                                     ## - učitavanje
potrebnih modula

i=0
k=0
l=0
udalj = -0.09778
pitch = []
yaw = []
roll = []
d_h = []
lista_tocaka = []

fig = pyplot.figure(figsize=(11, 8), dpi=90)      ## - definiranje grafa
za prikaz rezultata
ax = fig.add_subplot(111)                       ## - definiranje grafa
za prikaz rezultata

datoteka = open('lokacija\\datoteka.txt', 'w')    ## - otvaranje
datoteke za spremanje rezultata mjerenja

generalizacija = input("Unesite približnu udaljenost točaka u metrima: ")
## - postavljanje generalizacije grafa profila (udaljenosti između točaka u
metrima) - unos tipkovnicom
visina = input("Unesite približnu visinu početne točke: ")
## - postavljanje visine početne točke profila (visina u metrima) - unos
tipkovnicom

with open('lokacija\\podaci_mjerenja.txt', 'r') as f:    ## - otvaranje
datoteke sa podacima izmjere uz rješavanje problema zadnjeg retka
    linije=[L[:-1] for L in f.readlines()]

for linija in linije:                                   ## - prikupljanje
kuteva iz datoteke sa mjerenjima u petlji
    pitch.append(float(linija.split(',')[0]))
    yaw.append(float(linija.split(',')[1]))
    roll.append(float(linija.split(',')[2]))

while i<len(pitch):                                     ## - računanje
razlika visina u petlji
    d_h.append((0.09778 * math.sin(math.radians(pitch[i-1])))*-1)  ## -
množenje sa -1 jer je kut pitch dobiven iz IMU-a pozitivan ispod horizonta,
a negativan iznad horizonta
    i+=1

interval = generalizacija/0.09778                     ## - računanje intervala računanja
visine ovisno o zadanoj udaljenosti između točaka

for dh in d_h:                                         ## - računanje visina točaka
profila u petlji
    visina = visina + dh
    udalj = k*0.09778
    if(k%(int(interval))==0):

```

```

        datoteka.write("Udaljenost: %.2f" % udalj + " m  Visina h: %.2f" %
visina + " m\n")    ## - spremnje rezultata u txt datoteku
        tocka = Point(udalj,visina)    ## - kreiranje točke koja će se
prikazati na grafu
        lista_tocaka.append(tocka)
        k=k+1

while l<len(lista_tocaka)-1:    ## - crtanje grafa profila u petlji
    linija =
    LineString([(lista_tocaka[l].x,lista_tocaka[l].y),(lista_tocaka[l+1].x,list
a_tocaka[l+1].y)])
    l=l+1
    xLista,yLista = linija.xy
    ax.plot(xLista,yLista,"rd")
    ax.plot(xLista,yLista,"k-")

print 'Završeno !!'
datoteka.close()
pyplot.show()    ## - prikazivanje grafa

```

Životopis

CURRICULUM VITAE

Osobni podaci

Ime i prezime	Dalibor Sruk
Adresa	Čugovečki put 7, 10040 Zagreb
Telefon	+385 91 798 3378
E-mail	deska27@gmail.com
Godina rođenja	1987.

Radno iskustvo

ljeto 2010.; zima 2010.	Geoprojekt d.o.o. , Zagreb Zadaci: - terenska izmjera
ljeto 2013.	USUS FRUCTUS d.o.o. , Zagreb Zadaci: - terenska izmjera

Školovanje

rujan 2006. - rujan 2013.	Geodetski fakultet , Zagreb
Fakultet	Smjer: Geoinformatika
listopad 2002. - lipanj 2006.	XV Gimnazija , Zagreb
Srednja škola	Smjer: Prirodoslovno - matematički

Strani jezici**Engleski jezik**

Čitanje	Napredno
Pisanje	Napredno
Govor	Napredno

Njemački jezik

Čitanje	Osnovno
Pisanje	Osnovno
Govor	Osnovno

Znanja i vještine

Tehničke sposobnosti

Geodetske sposobnosti

Kratak opis: Teorijsko znanje stečeno tijekom petogodišnjeg studija. Praktično iskustvo stečeno tijekom honorarnih poslova na terenu i u uredu uglavnom katastarske domene.

Računalne sposobnosti

Kratak opis: Vrlo dobro poznavanje AutoCAD softverskog paketa. Odlično snalaženje u MS Office paketu, točnije Wordu, Excelu i PowerPointu. Napredno znanje rada u aplikacijama GRASS GIS i SAGA GIS.

Programski jezici

Kratak opis: Vrlo dobro poznavanje programskog jezika Python zajedno s Qt i QGIS platformom. Osnovno znanje izrade aplikacije. Osnovno znanje C++, Pascal jezika, javascripta, php-a, html-a i java.