SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 2451

Klasifikacija pokreta ljudskog tijela temeljena na podacima s inercijskih senzora

Ivan Trubić

Zagreb, svibanj 2021.

Umjesto ove stranice umetnite izvornik Vašeg rada.
Da bi ste uklonili ovu stranicu obrišite naredbu `\izvornik`.

SADRŽAJ

1. Uvod

Karakteristike pokreta ljudskoga tijela vrlo su individualne te ovise o mnogo čimbenika kao što su genetika, odgoj te fizička sprema. Ti pokreti su toliko jedinstveni da se mogu koristiti za identifikaciju osoba dok su s druge strane toliko slični da čak i manje devijacije u tim pokretima također mogu ukazivati na neke zdravstvene probleme. Ljudski pokreti mogu se klasificirati na razne načine koristeći računalni vid ili razne senzore postavljene na ljudskome tijelu. Svaka od metoda ima svoju granu primjene kao: identifikacija osoba temeljenog na hodu koristeći računalni vid na nadzornim kamerama (C, 2019), pračenje pokreta igrača u interakciji sa igrama u virtualnoj stvarnosti (Zhang, 2017), korištenje inercijskih (IMU) senzora za precizno snimanje hoda u svhu otkrivanja bolesti i rehabilitacije te mnoge druge.

Klasifikacija pokreta vrlo je složen problem te kao takav nema dobro rješenje koristeći klasične algoritme. Razvojem moči računala te metoda strojnoga učenja ovaj problem postaje rješiv. Za snimanje pokreta može se koristiti kamera ili senzori. Koristeći kameru, na snimci se koriste metode računalnog vida te se traže karakteristike ljudskog tijela kako bi se na snimci prepoznala osoba te koristeći te karakteristične točke analizira se hod. Također, kamera može snimati osobu sa posebno postavljenim vizualnim oznakama po djelovima tijela te koristeći te vizualne oznake analizirati pokrete. Nedostatak kamera je taj što snimaju iz jedne perspektive te zbog toga može doći do okluzije oznaka. Inercijski (IMU) senzori eliminiraju kamere te ne pate od problema okluzije. Inercijski senzori su relativno jeftini i mali uređaji koji se postave na ključne djelove ljudskoga tijela te pružaju vrlo dobar uvid u ljudske pokrete. Primjerice mogu se staviti na ruke te upravljati igrama i uređajima ali se mogu koristiti i u medicinske svrhe za analizu hoda i diagnosticiranje zdravstvenih problema kao i za provođenje terapijskih vježbi bez nadzora stručnjaka. Ovaj rad će se više fokusirati na medicinski aspekt klasifikacije pokreta, preciznije analizu hoda (*eng.* gait) i terapiju koljena.

2. Razrada

2.1. Bolesti koljena

2.2. Dosadašnja rješenja

2.3. Rješenje koristeći IMU senzore

2.4. Analiza IMU senzora

2.5. Analiza dostupnih baza podataka

2.6. Stvaranje vlastite baze podataka

Za stvaranje vlastite baze podataka potrebno je definirati svrhu i ciljeve te baze a zatim odrediti metodu prikupljanja podataka. U ovome će radu fokus biti primarno na stvaranje baze podataka korisne za analizu pokreta koja bi se mogla upotrijebiti u svrhe fizikalne terapije koljena. Za snimanje općenitih pokreta može se iskoristiti mnogo senzora postavljenih po cijelome tijelu ali za potrebe snimanja jednoga zgloba potrebno je koristiti dva senzora, po jedan sa svake strane uda kojeg zglob povezuje. U konkretnom primjeru koljena to su nadkoljenica i potkoljenica.

Ovisno o razini preciznosti, cjeni i dostupnosti potrebno je odabrati i same senzore za prikupljanje podataka. Redovito se sustavi senzora za istraživačke svrhe rade baš za tu namjenu te postoji mnogo sustava otvorenog koda i otvorenih komponenti koji se mogu iskoristiti. Razni proizvođači nude i svoja već gotova komercijalna rješenja za koje garantiraju rad i pružaju podršku ali takvi sustavi ponekad nisu adekvatni za istraživačke svrhe zbog svoje zatvorenosti i potencijalnog manjka interoperabilnosti sa drugim uređajima i programima. U ovome radu, zbog jednostavnosti, koristiti će se pametni telefon.

Tipičan životni vijek jednog pametnog telefona u prosjeku je dvije godine nakon čega korisnici redovito kupuju nove modele jer stariji postaju neadekvatni po pitanju trajanja baterije, količine memorije, performansama komponenti i slično. Svaki takav stariji model telefona vrlo često stoji nekorišten iako je još uvijek funkcionalan. Gotovo svaki pametni telefon u sebi sadrži IMU senzor te u sebi već ima povezivosti poput *WiFi i Bluetooth*. Uzimajući u obzir također da pametni telefoni imaju i ugrađenu bateriju koja može još uvijek biti dovoljno dugotrajna, ekran osjetljiv na dodir kao metodu interakcije i to sve ukomponirano u uređaj koji svojim dimenzijama stane u džep dolazimo do zaključka da su pametni telefoni i više nego adekvatni za prikupljanje podataka. Također, vrlo je realna pretpostavka da svaka osoba ima pristup minimalno jednome, vjerojatno i dva pametna telefona što čini prikupljanje velike količine podataka od većeg broja sudionika znatno jednostavnijim. Koristeći držaće pametnog telefona za trčanje ili marama, uređaji se mogu postaviti osobi na bilo koji dio tijela vrlo jednostavno i svaka bi osoba iz svoga doma mogla pridonjeti stvaranju baze podataka.

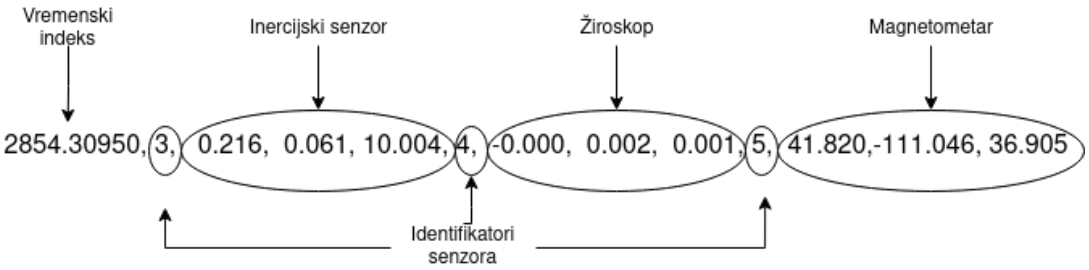
Aplikacija koja očitava vrijednosti može biti napravljena posebno, no na tržištu postoje gotove aplikacije, redovito otvorenoga koda, koje se već time bave. Bitno je pronaći odgovarajuću aplikaciju no u slučaju da takva ne postoji potrebno je izraditi svoju. Mnoge aplikacije u trgovini ne nude selekciju samo određenih senzora već snimaju sve senzore kojima pametni telefon raspolaže, te u većini slučajeva nude formatiranu pohranu podataka isključivo u memoriju uređaja što je vrlo neadekvatno ako bi se uređaji koristili primjerice za obradu informacija u stvarnome vremenu ili za centralizirano prikupljanje podataka primjerice na računalo. Neke od takvih aplikacija su *Sensor Data* i *phyphox* koje se koriste u edukaciji te su u doba COVID pandemije nezamjenjivi alati za vršenje fizikalnih eksperimenata kao dio školske zadaće. Nedostatak ovih aplikacija je taj što sve podatke snimaju na lokalnoj memoriji uređaja te nisu primjerene za obradu podataka u stvarnome vremenu. Aplikacija *Sensorstream IMU+GPS* ne nudi grafičke prikaze podataka već u svojoj vrlo bazičnoj funkcionalnosti nudi odabir senzora uz prikaz trenutnih vrijednosti istih te odabir akcije koju želimo napraviti sa tim podacima. Ponuđene su funkcije spremanja vrijednosti u CSV (*Comma separated value*) formatu, slanje podataka koristeći UDP protokol te kombinaciju oboje. Za slanje podataka korištenjem UDP protokola potrebno je navesti određenu IP adresu uređaja te *port* na kojemu uređaj očekuje promet. Također nudi 4 frekvencije uzorkovanja označene sa *slow, medium, fast i fastest*. Te frekvencije uzorkovanja nisu dobro dokumentirane te nigdje nije navedeno koliko one iznose zapravo. Za mjerenje tih frekvencija može se iskoristiti alat *wireshark*. Wireshark je alat otvo-

renoga koda koji služi za analizu mrežnoga prometa. Vrlo raširen i nezamjenjiv alat za svaku granu računarstva koja se bavi mrežnim prometom. Nudi razne opcije od kojih je jedna od močnijih ugrađeni filter koji vrlo precizno može naći određeni paket unutar snimke mrežnog prometa. Koristeći tu mogučnost promet se može snimiti na računalu te kasnije filtrirati samo one pakete koje mobilni uređaj šalje. Gledajući vremenske indekse prvoga i posljednjega paketa dobivamo ukupno trajanje snimanja te znajući točan broj paketa može se izračunati približna frekvencija uzorkovanja te su one prikazane u tablici 2.1.

Uzorkovanje	Trajanje (s)	Broj poruka	Frekvencija
Slow	12.21	61	5
Medium	3.98	61	15
Fast	6.85	344	50
Fastest	2.14	299	124

Tablica 2.1: Izmjerene frekvencije uzorkovanja

Također koristeći wireshark alat može se jedan paket analizirati i vidjeti format podataka koju uređaj šalje. Podaci su u CSV formatu u kojima se šalje vremenski indeks, identifikacijski brojevi senzora te same vrijednosti senzora kako se vidi u slici 2.1. Svi podaci prikazuju vrijednosti redom x, y, i z osi.



Slika 2.1: Primjerak primljenih podataka

Kako bi se ti podaci primili i pohranili na adekvatan način potrebno je napraviti program klijent koji osluškuje na određenim portovima. Implementacija tog programa u ovome radu biti će napravljena koristeći skriptni jezik python.

Python je vrlo popularan jezik mnogih mogučnosti koji je vrlo jednostavan za naučiti i koristiti te zbog svoje popularnosti raspolaže mnogim bibliotekama. Neke od tih vrlo korisnih biblioteka koje će biti korištene u ovome radu su *numpy*, *matplotlib* te *socket*. Python se kao skriptni jezik izvodi liniju po liniju te se svaka linija iterpretira tek kad ona dođe na red za izvršavanje što ukida proces prevođenja programa u

strojni kod kao što je to primjerice potrebno kod programskog jezika *C*. Iz tog razloga je vrijeme izvršavanja takvoga koda znatno dulje što znatno utječe na performanse programa. Kako bi se izvođenje kompliciranijih matematičkih operacija znatno ubrzalo te kako bi se ponudio veći opseg već gotovih funkcija napravljena je biblioteka *numpy*. Numpy je biblioteka koja nudi gotovo sve potrebne funkcije za izradu programa koji se koriste u znanosti. Implementira višedimenzionalne matrice te sve funkcije za operaciju nad njima uključujući promjenu oblika matrice te sve matrične operacije. Također nudi i funkcije za Fourierovu transformaciju, statističku obradu, generatore nasumičnih brojeva i slično. Numpy implementacija matrica je znatno bliža onakvim matricama kao što su u programskome jeziku C te su kao takve memorijski znatno manje zahtjevne od originalnih matrica koje nudi python. Također kako bi operacije nad matricama bile brze operacije su implementirane u programskome jeziku C te prevedene unaprijed tako da biblioteka nudi vrlo jednostavnu sintaksu uz jako veliku brzinu izvođenja te manju veličinu datoteke pri pohrani podataka. Biblioteka *socket* nudi funkcije za umreživanje koje su potrebne kako bi se podaci uspješno primili sa mobilnih uređaja je *matplotlib* koji je potreban za vizualizaciju podataka u obliku grafova.

Listing 2.1: Ostvarivanje veze na strani klijenta

```
1  def init_client(srv_port):
2      client_socket = socket.socket(family=socket.AF_INET,\\
3                                    type=socket.SOCK_DGRAM)
4      client_socket.bind((self_ip, srv_port))
5      return client_socket
6
7  client = []
8
9  client.append(init_client(port))
10 print("UDP client up")
```

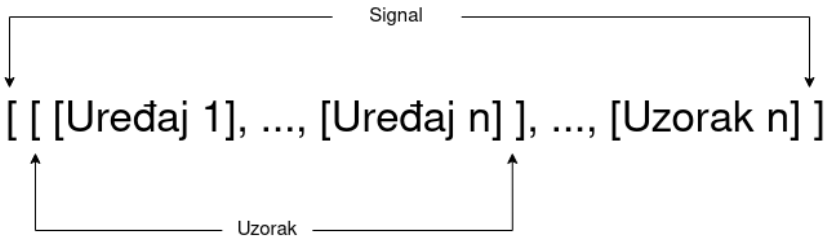
U kodu 2.1 je vidljiv način na koji se koristi biblioteka *socket*. Svaki mobilni uređaj predstavlja instanca klase *socket* koja se sprema u listu klijenata `client` što omogučuje povezivanje proizvoljnog broja mobilnih senzora pri čemu svaki senzor treba imati svoj poseban port. Port na računalu bi trebao biti slobodan odnosno niti jedna druga aplikacija ne bi smjela koristiti taj port za svoj promet. Dobra praksa nalaže da se za osobne potrebe koriste portovi brojeva većih od 1024 kako bi se izbjegli moguči zauzeti portovi nekakvih standardnih protokola kao što su *https* i slični. U ovome slučaju mobilni uređaj s indeksom 0 koristi port broja 5555 te svaki sljedeći zauzima port $5555 + i$ pri čemu je i indeks uređaja. Točan broj je potrebno unjeti ručno na svaki

uređaj koji šalje podatke. Stvaranje instance *socket* odvija se u liniji 2 i 3 pri čemu se u argumentu treba navesti tip veze. Za familiju veze se navodi `AF_INET` što predstavlja IP protokol te pod tip se navodi `SCOK_DGRAM` što označava korištenje UDP protokola. UDP (eng. *User Datagram Protocol*) je jedan od osnovnih transportnih protokola koji podatke prenosi po principu *best effort*, pretpostavlja se da su datagrami došli do odredišta u ispravnome redosljedu te se ne potvrđuje primitak datagrama. Ovakav protokol ne garantira cjelovitost podataka no nudi manje zagušenje prometa zbog nedostatka potvrđivanja svakog primljenog datagrama i retransmisije u slučaju gubitka te se iz tog razloga koristi za prijenos videa ili VoIP (eng. *Voice over IP*) pozive. U liniji 4 se instanca razreda *socket* povezuje s odgovarajučim portom na odgovarajučem mrežnom uređaju koji je predstavljen vlastitom IP adresom `self_ip`. Na kraju funkcija `init_client` vraća instancu kako bi bila spremljena u klijentsku listu.

Listing 2.2: Primanje podataka

```
1  while(True):
2      sample = []
3      for phone in client:
4          bytes_adress_pair = phone.recvfrom(buffer)
5          data = []
6          message = bytes_adress_pair[0].decode().split(",")
7          for i in message:
8              data.append(i.strip())
9
10         # filteri podataka
11         if "4" not in data:
12             continue
13
14         for i in data_filter:
15             data.pop(i)
16
17         if len(data) != 0:
18             sample.append(data[:6])
19
20     if len(sample) != 0:
21         signal.append(sample)
22     else:
23         continue
```

U odsječku koda 2.2 prima se datagram svakog mobilnog uređaja. Svaki datagram se uzima iz međuspremnika koji je veličine 1024 bajta. Svaki datagram se sastoji od samih podataka (u 6. liniji koda pod indeksom 0) te IP adrese pošiljatelja koja u ovoj implementaciji nije potrebna te se može izostaviti. Nad podacima se vrše obrade poput uklanjanja bjelina (linija 8.) te rastavljanje podataka iz jednog monolitnog zapisa podataka u više posebnih podataka pomoću `split` funkcije uzimajući zarez kao graničnik. Zatim te podatke valja filtrirati pri čemu se uklanja *vremenski indeks*, identifikatori senzora te podaci magnetometra vidljivi na slici 2.1. Eksperimentalno se pokazalo da se pri prvih nekoliko mjerenja od početka slanja podataka sa mobilnih uređaja izostavljaju vrijednosti žiroskopa te se gledajući prisutnost tih vrijednosti dodatno vrši čekanje na spremnost svih senzora u mobilnome uređaju. U ovome kodu to se vrši uvjetovanjem u liniji 11. Struktura podataka je prikazana u slici 2.2.



Slika 2.2: Struktura podataka

Uzorak jednoga uređaja je lista vrijednosti njegovih senzora. Jedan uzorak je lista uzoraka **svih** uređaja. Signal je samo lista uzoraka. Na ovaj se način akcija može snimiti i koristeći biblioteku *numpy* pohraniti na računalo za daljnju obradu.

2.7. Implementacija metode strojnoga učenja

2.8. Rezultati

3. Zaključak

Zaključak.

LITERATURA

Seckiner D Mallett X Maynard P Meuwly D Roux C. Forensic gait analysis - morphometric assessment from surveillance footage. *Forensic science international*, 296 (57-66), 2019.

Wei Fang Lianyu Zheng Huanjun Deng Hongbo Zhang. Real-time motion tracking for mobileaugmented/virtual reality using adaptivevisual-inertial fusion. *Sensors*, 2017.

Klasifikacija pokreta ljudskog tijela temeljena na podacima s inercijskih senzora

Sažetak

Sažetak na hrvatskom jeziku.

Ključne riječi: Ključne riječi, odvojene zarezima.

Title

Abstract

Abstract.

Keywords: Keywords.