Univerzitet u Beogradu
Elektrotehnički fakultet

Đorić Dušan, 0150/2014
Nisić Đorđe, 0421/2014
Aćimović Miloš, 0146/2014

Bežično upravljanje Stellaris Evalbot-om

projekat iz predmeta Principi modernih telekomunikacija

mentor:
prof. dr Milan Bjelica

Beograd, jul 2017.

Sažetak

Ovaj projekat je imao za cilj realizaciju bežičnog upravljanja *Stellaris Evalbot*-om korišćenjem TCP/IP steka i malog računara *Raspberry Pi*. U nastavku dokumentacije biće opisano rešenje, uključujući šemu i korišćen softverski stek. Redosled izlaganja je prilagođen redosledu razvijanja tj. ideći od dna steka ka vrhu.

Ključne reči: embedded, ethernet, Android Studio, Raspberry Pi, EvalBot, LM3S9B92, lwip, python, robot, Stellaris, tkinter, IoT

Sadržaj

Glava 1

Uvod

Cilj ovog projekta je bila nadogradnja na projekat koji je izrađen na ovom predmetu. 1.1 Njegova realizacija je imala sledeću šemu:

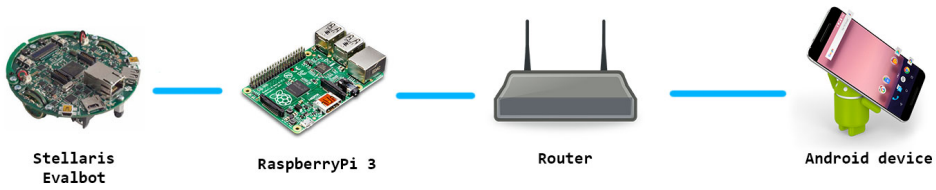
1. PC računar
2. Ruter
3. Stellaris Evalbot

Slika 1.1: Šema prethodnog rešenja.

Šema koja je u okviru ovog projekta realizovana ima sledeću strukturu:

1. PC Računar ili Android uređaj
2. Ruter
3. Raspberry Pi 3
4. Stellaris Evalbot

Slika 1.2: Šema rešenja korišćenjem računara.



Slika 1.3: Šema rešenja korišćenjem Android uređaja.

Glava 2

Opis komunikacije uređaja

Stellaris Evalbot i *Raspberry Pi* čine jedan sistem. *Stellaris Evalbot* dobija napajanje od tri AA baterije a *Raspberry Pi* se može napajati ili preko istih ili(onako kako je u ovom projektu realizovano) preko *Power Banke*. *Stellaris Evalbot* i *Raspberry Pi* su povezani *Ethernet* vezom. GUI je realizovan u programskom jeziku Python i sastoji se od dugmadi koje korisniku omogućavaju da pokrene svog robota u željenom smeru kao i da ga zaustavi ili promeni brzinu. *Raspberry Pi* kreira priključnicu na kojoj osluškuje poruke od računara(ili Android uređaja) i kreira priključnicu ka *Stellarisu*. Klikom na dugme ostvaruje se TCP/IP komunikacija sa *Raspberry Pi* računarom, naime šalje se poruka. Po prijemu poruke *Raspberry Pi* prosleđuje poruku *Stellaris-u* koji će po prijemu preko svoje serverske priključnice pozvati funkciju koja tumači sadržaj poruke i na osnovu kog izvršava odgovarajuću akciju.

2.1 Opis Stellaris Evalbot-a

Ploča *Texas Instruments*® *Stellaris*® *EK-EVALBOT* [3] je platforma koja se može koristiti za razvoj softvera i zabavno istraživanje mogućnosti malog robota. Takođe se može koristiti kao vodič za dizajn prilagođene ploče koristeći *Stellaris-ov* mikrokontroler. EK-EVALBOT uključuje *Stellaris ARM*® *CortexTM-M3* 2.1 mikrokontroler i sledeće:

Slika 2.1: LM3S9B92 mikrokontroler.

- OLED displej
- Dva motora povezana s točkovima
- Senzori točkova za merenje rotacije točkova
- Bump senzora na prednjoj strani
- Dugmad za korisnika
- Ethernet kontroler/interfejs
- USB host i uređaj
- Slot za SD karticu
- I2S audio DAC i zvučnik
- Zaglavlje EM konektora za bežične module

2.2 Raspberry Pi

Raspberry Pi je jeftin, osnovni računar koji je prvobitno bio nameren za podsticanje interesa za računarstvo među školskom decom. *Raspberry Pi* se nalazi na jednoj pločici i ima priključke za:

- HDMI
- USB 2.0
- Kompozitni videozapis
- Analogni zvuk
- Napajanje
- Internet
- SD kartica

Računar radi u potpunosti na open-source softveru i daje učenicima mogućnost mešanja i usklađivanja softvera prema aplikaciji koju žele da izrade.

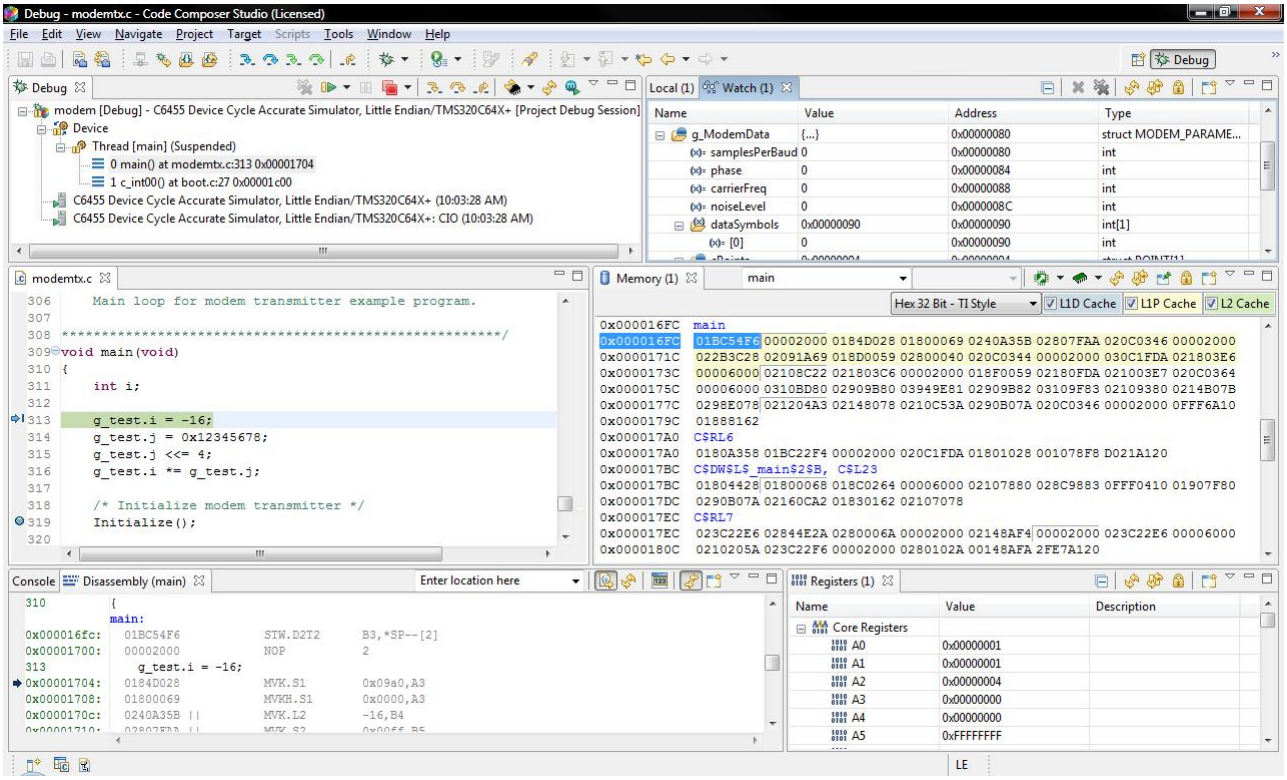
Raspberry Pi je debitovao u februaru 2012. godine. Grupa iza razvoja računara - *Raspberry Pi* Foundation - pokrenula je projekt kako bi računarstvo postala zabava studentima, a istovremeno stvara zanimanje za rad na računaru na osnovnom nivou. Za razliku od korišćenja upakovanog računara od strane proizvođača, *Raspberry Pi* pokazuje svoju suštinu iza plastike. Čak je i softver, zahvaljujući open-source filozofiji, prilika studentima da istraže temeljni kôd - ako to žele. Veruje se da je *Raspberry Pi* idealan alat za učenje, jer je jeftin, jednostavan za zamenu i treba samo tastaturu i monitor za pokretanje.

Glava 3

Programiranje robota

3.1 Okruženje

Za okruženje je korišćen Code Composer Studio v63.1, iako je dostupna novija verzija v7 u njoj nestaje podrška za procesor LM3S9B92.



Slika 3.1: Code Composer Studio.

3.2 Upravljanje robotom

Programiranje robota je urađeno korišćenjem *EK-EVALBOT Firmware Development Package*[1]. Ova biblioteka pruža podršku za dva modela programiranja: direktno registarsko i softverski model. Svaki model može se koristiti samostalno ili kombinovano, u zavisnosti od potreba aplikacije ili programskog okruženja koje odredi programer. Svaki model programiranja ima prednosti i nedostatke. Upotreba direktnog pristupa registru daje manji i efikasniji kod od korišćenja softverskog modela. Međutim, model direktnog pristupa registru zahteva detaljno poznavanje rada svakog od registara i polja bitova, kao i njihova interakcija i bilo koji slijed koji je potreban za pravilno funkcionisanje periferija. Programer je izolovan od tih detalja upotrebom softverskog modela, koji zahteva manje vremena za razvoj aplikacije. U ovom rešenju je korišćen softverski model jer glomaznost koju unosi softverski model ne utiče na željene performanse, a smanjuje vreme izrade aplikacije.

3.3 Glavni program robota

Na početku inicijalizuju se periferije koje upravljaju motorima i LED svetlima.

```
LEDsInit();
MotorsInit();
MotorSpeed(LEFT_SIDE, speed << 8);
MotorSpeed(RIGHT_SIDE, speed << 8);
MotorDir(LEFT_SIDE, FORWARD);
MotorDir(RIGHT_SIDE, FORWARD);
```

Inicijalizuju se druge periferije među kojima je Ethernet kontroler i njegove LED lampice.

```
ROM_SysCtlPeripheralEnable(SYSCTL_PERIPH_ETH);
ROM_SysCtlPeripheralReset(SYSCTL_PERIPH_ETH);

ROM_SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
GPIOPinConfigure(GPIO_PF2_LED1);
GPIOPinConfigure(GPIO_PF3_LED0);
GPIOPinTypeEthernetLED(GPIO_PORTF_BASE, GPIO_PIN_2 | GPIO_PIN_3);
```

Omogućavaju se prekidi.

```
ROM_IntMasterEnable();
```

3.3.1 Korićenje lwIP biblioteke

Pošto robot nema svoj operativni sistem i u okviru njega podršku za internet protokole, korišćena je lwIP biblioteka[2]. Ovom linijom

```
lwIPInit(pucMACArray, 0, 0, 0, IPADDR_USE_DHCP);
```

se ona inicijalizuje da koristi DHCP protokol za dobijanje adrese. Kada se povežu *Stellaris* i *Raspberry Pi* preko Ethernet kabla *Stellaris* će poslati DHCP Discover poruku koja je i MAC i IP broadkast poruka time tražeći DHCP server. DHCP server onda odgovara sa DHCP Offer porukom u kojoj nudi IP adresu datom host-u. Host onda šalje DHCP Request poruku tražeći od DHCP servera da mu dodeli datu IP adresu i na kraju DHCP server odgovara sa DHCP ACK porukom. Ovo iziskuje da se na *Raspberry Pi* izvršava DHCP server. Ostalo je još da robot uspostavi serversku priključnicu na portu 23 i u beskonačnoj petlji osluškuje konekcije.

```
struct tcp_pcb *ptel_pcb;
ptel_pcb = tcp_new();
tcp_bind(ptel_pcb, IP_ADDR_ANY, 23);
while (1) {
ptel_pcb = tcp_listen(ptel_pcb);
tcp_accept(ptel_pcb, echo_accept);
}
```

Na ovaj način može se prekinuti konekcija zatvaranjem soketa i ponovo uspostaviti dinamički.

```
tcp_accept()
```

metoda prihvata kao callback funkciju echo_accept koja takođe poziva funkciju

```
tcp_recv()
```

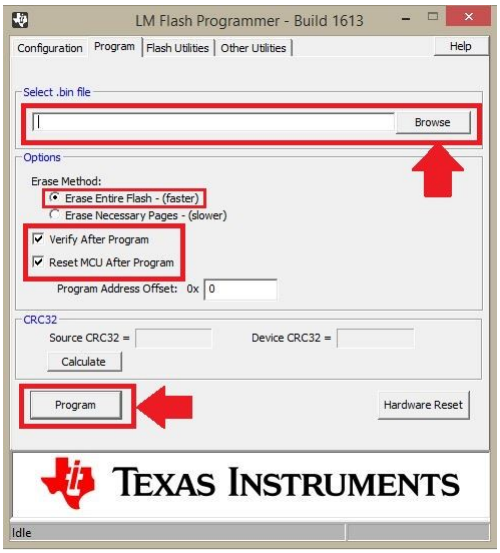
koja takođe prima kao callback funkciju echo_recv koja poziva funkciju

```
movementHandler()
```

u kojoj se odvija logika zadavanja odgovarajućih komandi robotu. movementHandler funkcija na osnovu primljenog karaktera pokreće/zaustavlja motore i menja njihov smer. Kao dodatna funkcija je stavljeno da se preko jedne promenljive označi kada se ide unazad i tada se u interrupt handler funkciji za prekid od timer-a smenjuju lampice nakon određenog vremena.

3.4 Spuštanje programa na robota

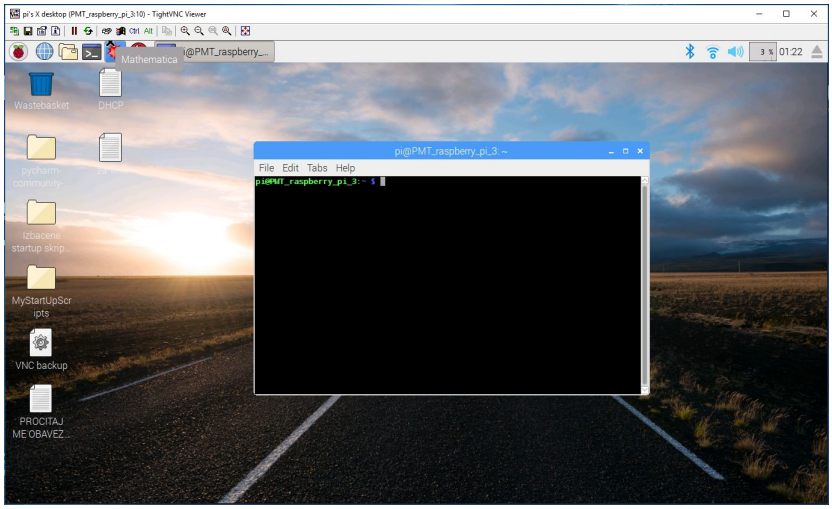
Za spuštanje programa korišćen je alat *LM Flash Programmer*. Ovaj alat ima veoma pogodan korisnički interfejs. Povezivanjem robota USB kablom ka računaru, odabiranjem prethodno prevedenog programa i klikom na dugme *Program* proces spuštanja programa na robota je završen.3.2



Slika 3.2: LM Flash Programmer.

3.5 Raspberry Pi server

Stellaris Evalbot i korisničke aplikacije su povezani preko serverske aplikacije koja se nalazi na *Raspberry Pi*-u. Razlog postojanja *Raspberry Pi*-a kao nekog posrednika jeste to što *Stellaris Evalbot* nema podršku za WiFi, već samo ethernet interfejs. *Raspberry Pi* je treće generacije, model B. Koristi *Noobs with Pixel* operativni sistem koji je vrsta Raspbian operativnog sistema.

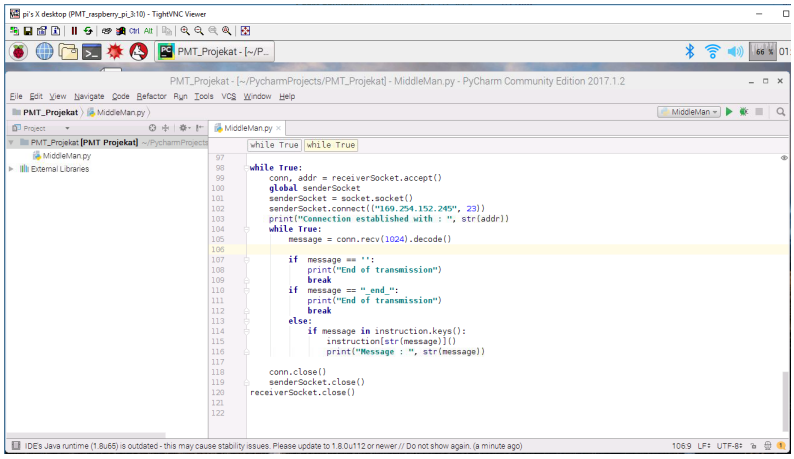


Slika 3.3: Noobs with Pixel (Raspbian OS).

Na *Raspberry Pi*-u je podignut DHCP server koji *Stellaris Evalbot*-u dodeljuje IP i MAC adresu kako bi on mogao da učestvuje u WiFi komunikaciji. Celokupni sistem funkcioniše tako što se prilikom podizanja sistema na *Raspberry Pi*-u pokreće aplkacija koja startuje DHCP server. Onda se *Stellaris Evalbot*-u (koji je pritom povezan sa Pi-om) dodeljuju IP i MAC adrese. Nakon toga se, u pozadini, pokreće *MiddleMan* aplikacija koja čeka da pristignu poruke sa korisničke aplikacije, i prosleđuje ih *Stellaris Evalbot*-u.

3.5.1 Struktura serverske aplikacije

Na Raspberry Pi-u se izvršava *MiddleMan* aplikacija. Ona u stvari predstavlja server. Pisana je u Python programskom jeziku, koristeći editor *PyCharm*.



Slika 3.4: MiddleMan aplikacija u PyCharm editor-u.

Raspberry Pi je konfigurisan tako da ima static IP adresu : 192.168.1.33. Serverska aplikacija otvara socket za ovu IP adresu na portu 4001. Čim se neko uspostavi vezu preko ovog socket-a, ulazi se u while petlju u kojoj čeka da joj pristignu poruke. Poruke koje stižu su tipa "_forward_", "_backward_"itd. Kada poruka stigne, ona se prosleđuje *Raspberry Pi*-u. Ovaj ciklus se vrši sve dok se ne primi poruka "_end_", kada se izlazi iz while petlje i čeka da neko ponovo uspostavi vezu sa *Raspberry Pi*-om.

3.6 PC klijent

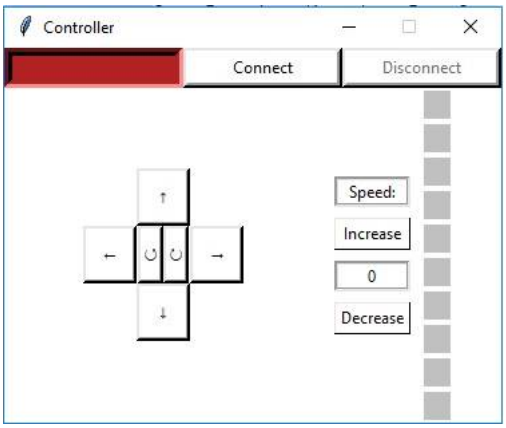
PC klijent aplikacija je izrađena u Pythonu. GUI je veoma jednostavan i intuitivan za korišćenje. Na početku je potrebno uneti odgovarajuću IP adresu Evalbota, kome saljemo komande, nakon čega je omogućen dalji rad sa aplikacijom. Postoje dve grupe dugmeta:

- za kretanje u nekom pravcu ili rotaciju
- -za kontrolisanje brzine(ukupno 10 mogućih brzina).

Pored dugmenta, aplikacija obezbeđuje i komunikaciju preko tastature. Na pojedinim tasterima su postavljeni ActionListeneri, tako da pritiskom na taster, aplikacija salje odgovarajuću komandu Evalbotu. Tasteri koji se koriste su:

- Strelice(Arrowkeys) za pomeranje u nekom pravcu
- Q za povećavanje brzine
- E za smanjivanje brzine
- A za rotaciju u levo
- D za rotaciju u desno

Za izradu aplikacije korišćena je "Tkinter"biblioteka (`https://docs.python.org/2/library/tkinter.html`).



Slika 3.5: Desktop GUI.

3.6.1 Struktura klijentske aplikacije

Svako dugme u aplikaciji ima prikacen neki ActionListener za sebe. Pritiskom na dugme se aktivira "ButtonPress"i "ButtonRelease"osluškivači koji su vezani za to dugme.

```
self.w.bind("<ButtonPress>", self.up_press)
    self.w.bind("<ButtonRelease>", self.up_release)
```

Ovi osluškivači pozivaju odgovarajuće funkcije koje prosleđuju komandu Evalbotu.

```
def up_press(self,event=None):
        if self.connected == 1:
            self.senderSocket.send("_forward_".encode())
            sleep(0.2)
    def up_release(self,event=None):
    self.senderSocket.send("_stop_".encode())
```

Što se tiče tastature, postoji poseban osluškivač koji detektuje "keypress"i "keyrelease". Uz pomoć dodatne funkcije i boolean niza aplikacija odredjuje koji je taster pritisnut i na osnovu toga poziva odgovarajuću funkciju.

```
    Pomoćna funkcija koja određuje koji taster je pritisnut :

    def keyup(self,event=None):
        inst= event.keysym
        if inst== "Up":
            self.heldDown[0]=0
            self.up_release()
        if inst== "Left":
            self.heldDown[1]=0
            self.left_release()
        if inst== "Down":
            self.heldDown[2]=0
            self.down_release()
        if inst== "Right":
            self.heldDown[3]=0
            self.right_release()
        if inst== "a":
            self.heldDown[4]=0
            self.rl_release()
        if inst== "d":
```

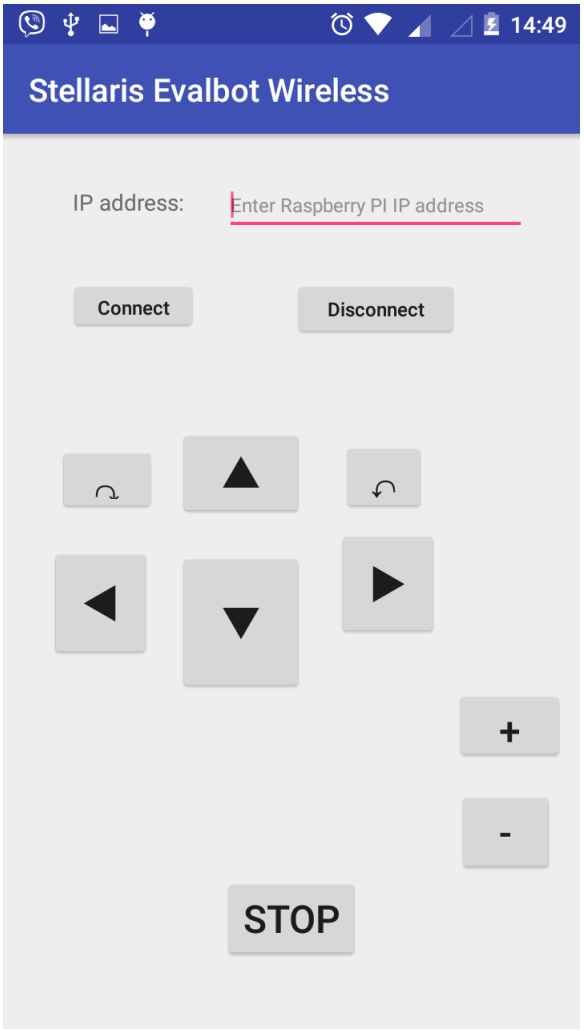
```
        self.heldDown[5]=0
        self.rr_release
```

Pomoćna funkcija koja određuje koji taster je pušten :

```
def keyup(self,event=None):
    inst= event.keysym
    if inst== "a":
        self.heldDown[4]=0
        self.rl_release()
    if inst== "d":
        self.heldDown[5]=0
        self.rr_release()
    if inst== "Up":
        self.heldDown[0]=0
        self.up_release()
    if inst== "Left":
        self.heldDown[1]=0
        self.left_release()
    if inst== "Down":
        self.heldDown[2]=0
        self.down_release()
    if inst== "Right":
        self.heldDown[3]=0
        self.right_release()
```

3.7 Android aplikacija

Ova aplikacija 3.6 ima veoma jednostavno strukturu. Pri pokretanju postavljaju se osluškivači za pritisnuta dugmad koji šalju odgovarajuće komande na IP adresu čija se vrednost unosi preko tekstualnog polja. Dugmad su veoma intuitivna za korišćenje. Za izradu aplikacije je korišćen *Android studio* i veb sajt (`https://developer.android.com/index.html`)



Slika 3.6: Android aplikacija sa GUI interfejsom za upravljanje.

3.7.1 Struktura klijentske aplikacije

Kao što je već spomenuto, u okviru ove aplikacije se postavljaju osluškivači na pritisak dugmad. Pošto se GUI na Android uređajima ne sme zaustavljati vremenski skupom operacijom kao što je otvaranje priključnice(*engl. socket*), ono se mora izdvojiti u posebnu nit. U tu svrhu je kreirana klasa *NetworkTask* koja proširuje klasu *AsyncTask* i nadjačava njenu metodu *execute* koja se izvršava u posebnoj niti.

```
connectButton.setOnClickListener(new Button.OnClickListener(){
    public void onClick(View v){
        ipAddr = ipField.getText().toString();
        networktask.execute();
        connectButton.setEnabled(false);
        ipField.setEnabled(false);
    }
});
```

Što se tiče dugmadi za upravljanje robotom na njima se nalaze osluškivači za pritiskanje i otpuštanje dugmeta. Funkcije koje obrađuju ove događaje pozivaju metodu klase *NetworkTask* koja opet zbog skupe operacije slanja poruke preko priključnice pravi novu nit u kojoj sprovodi datu akciju.

```
Button upTurn = (Button) findViewById(R.id.button3);
upTurn.setOnTouchListener(new Button.OnTouchListener(){
    public boolean onTouch(View v, MotionEvent theMotion){
        switch(theMotion.getAction()){
            case MotionEvent.ACTION_DOWN:
                networktask.sendDataToNetwork('w');
                break;
            case MotionEvent.ACTION_UP:
                networktask.sendDataToNetwork('x');
                break;
        }
        return true;
    }
});
```

Glava 4

Zaključak

Cilj ovog projekta bio je nadogradnja na postojeća inženjerska rešenja kao i primena i unapređenje znanja iz oblasti mikrokontrolera, programiranja mrežnih aplikacija, programskog jezika Python i Android aplikacija stečenih na predmetu Principi modernih telekomunikacija.

Literatura

[1] *EK-EVALBOT Firmware Development Package.*

[2] *Using the Stellaris® Ethernet Controller with Lightweight IP (lwIP).* Texas Instruments Inc., 2011.

[3] *Stellaris® Robotic Evaluation Board User's Manual.* Texas Instruments Inc., 2011.

[4] *Stellaris® Peripheral Driver Library USER'S GUIDE SW-DRL-UG-10007.* Texas Instruments Inc., 2013.

Slike