

»MLADI ZA NAPREDEK MARIBORA 2017«

34. SREČANJE

Krmiljenje avta s Raspberry Pi

Raziskovalno področje: Računalništvo in informatika

RAZISKOVALNA NALOGA

PROSTOR ZA NALEPKO

Avtor: MARKO KURNIK, ŽIGA PODHOSTNIK, DAMIJAN ROBNIK
Mentor: MARJAN URANJEK
Šola: SREDNJA ELEKTRO-RAČUNALNIŠKA ŠOLA MARIBOR

Februar 2017, Maribor

»MLADI ZA NAPREDEK MARIBORA 2017«

34. SREČANJE

Krmiljenje avta s Raspberry Pi

Raziskovalno področje: Računalništvo in informatika

RAZISKOVALNA NALOGA

PROSTOR ZA NALEPKO

Februar 2017, Maribor

KAZALO VSEBINE

KAZALO VSEBINE	2
KAZALO SLIK	3
1. POVZETEK	4
2. UVOD	5
3. ZAHVALA.....	6
4. VSEBINSKI DEL	7
4.1 Delovanje daljinsko vodenih igráč.....	7
4.2 Primerjava našega izdelka z obstoječimi.....	8
4.3 Izbira opreme	9
4.4 Raspberry Pi – Konfiguracija.....	11
4.5 Nastavitve mjpeg streamer-ja in spletne kamere	12
4.6 Kreiranje uporabniškega vmesnika (spletne strani).....	13
4.7 Koda spletne strani.....	14
4.8 Koda za kontroliranje avtomobilčka.....	17
5. ZAKLJUČEK	20
6. DRUŽBENA ODGOVORNOST.....	21
7. SEZNAM VIROV IN LITERATURE.....	22

KAZALO SLIK

Slika 1: Daljinsko voden avtomobilček	7
Slika 2: Raspberry Pi	9
Slika 3: WiPi	9
Slika 4: Spletna kamera	10
Slika 5: Nameščanje Apache strežnika	11
Slika 6: Testiranje kamere	12
Slika 7: Izgled uporabniškega vmesnika (spletne strani).....	13
Slika 8: Datoteka file.php	16
Slika 9: Python logo	17
Slika 10: Python koda za premikanje avtomobilčka	18

1. POVZETEK

Naša raziskovalna naloga predstavlja izboljšanje daljinsko vodenih igrac s pomočjo spletne strani, ki deluje kot uporabniški vmesnik. V nasprotju z večino takšnih igrac se za kontroliranje avtomobilčka ne uporablja daljinec, ampak tipkovnica. Prav tako ni potrebno biti prisoten ob igraci, ali jo fizično videti, ker lahko pot njene vožnje vidimo na spletni strani, ki je povezana s kamero na avtomobilčku. To bistveno poveča razdaljo oz. prostor, kjer lahko vozimo avtomobilček, saj je ta v večini primerov omejen z ovirami (zidovi), ki nam onemogočajo pogled na avtomobilček. Uporabniški vmesnik oz. spletna stran prikazuje vožnjo avtomobilčka iz perspektive, ki je običajna za veliko število računalniških igrac, zato naš izdelek ni primeren le za ljubitelje daljinsko vodenih igrac, temveč tudi za vse ljudi, katerim so všeč dirkalne računalniške igre. Za izdelavo spletne strani smo uporabili jezike, kot so HTML, JavaScript in PHP, za pisanje kode, ki je namenjena kontroliranju avtomobilčka, pa smo uporabili programski jezik Python.

2. UVOD

Za izdelavo te raziskovalne naloge smo se odločili, saj se nam je zdelo zanimivo upravljanje »avtomobilčka« na daljavo s pomočjo računalnika. Že nekaj časa smo razmišljali, da bi sprogramirali daljinsko voden igračo, kar pa nas je še posebej nagovorilo k raziskovalni nalogi, pa je bila ideja, da bi lahko vodili avtomobilček s pomočjo tipkovnice in spletne strani, na kateri bi lahko videli, kam vozimo. S uporabo tipkovnice želimo poenostaviti vodenje take igrače, prav tako pa bi lahko z pomočjo kamere in spletne strani vozili v oddaljenih prostorih, brez da bi bili fizično prisotni. Takšen daljinsko voden avtomobilček bi lahko uporabili v zabavne namene, lahko pa bi bil uporabljen tudi za raziskovalne namene, ali pa v primerih, ko bi želeli preiskati nevarna področja, do katerih sami ne bi morali priti. Avtomobilček bi programirali s pomočjo Raspberry Pi 2B v jeziku Python. Spletno stran pa bi oblikovali z HTML in CSS jeziki, prikaz kamere na strani pa s pomočjo PHP jezika.

3. ZAHVALA

Radi bi se zahvalili našemu mentorju, ki nam je dal idejo za to raziskovalno nalogo, nas vzpodbujal, da si pridobimo veliko novega znanja in nam nudil pomoč v primerih, ko smo naleteli na kakšne težave.

4. VSEBINSKI DEL

4.1 Delovanje daljinsko vodenih igrač

Daljinsko vodene igrače v večini delujejo na principu radijskega oddajnika in sprejemnika. Daljinec ima pri tem vlogo oddajnika, igrača pa vlogo sprejemnika. Podatki pri tem potujejo le v eno smer, od oddajnika (daljinec) proti prejemniku (igrača). Domet je pri takih igračah ponavadi nekje do 20 metrov.



Slika 1: Daljinsko voden avtomobilček

Poleg kratkega dometa se slabosti pojavijo tudi pri večjem številu takšnih igrač, saj se lahko zgodi, da pride do motenj. Te motnje pa nastanejo, ker so vse igrače na isti frekvenci. Pasovna širina radijskega oddajnika je zelo ozka, kar pomeni, da lahko oddaja le malo število podatkov, brez da bi pri tem prišlo do zakasnitev.

4.2 Primerjava našega izdelka z obstoječimi

Odločili smo se, da bi s pomočjo našega znanja odpravili ali vsaj zmanjšali nekatere izmed teh slabosti in celo dodali nekaj novih funkcij.

Slabosti, ki smo jih nameravali odpraviti:

- Zmanjšanje zakasnitev
- Povečanje pasovne širine
- Zvišanje hitrosti (teoretično do 10 Gb/s)
- Dvosmerna komunikacija
- Povečanje dometa

Dodane funkcije:

- Uporabniški vmesnik
- Video nadzor (preko spletne kamere)

4.3 Izbira opreme

Pri naši raziskovalni nalogi smo uporabili Raspberry Pi 2B, zanj smo se odločili, ker je odprtokoden, lahko prenosljiv in ima veliko podporo za hardware (kamera, wifi, display,...) in software (Apache, SQL, Python,...). Služil nam je kot strežnik za uporabniški vmesnik in kot nadzor nad avtomobilčkom.



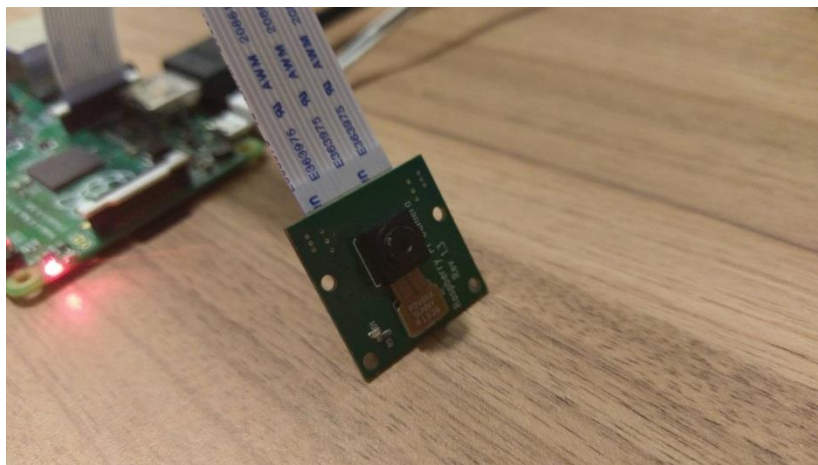
Slika 2: Raspberry Pi

Uporabili smo tudi modul WiPi za Wifi povezavo, ki ima največjo teoretično hitrost 100 Mb/s.



Slika 3: WiPi

Ključen del našega izdelka je bila spletna kamera, ki nam je omogočala nadzor poti avtomobilčka. Slika pa je bila prikazana na spletni strani, ki je delovala kot uporabniški vmesnik.

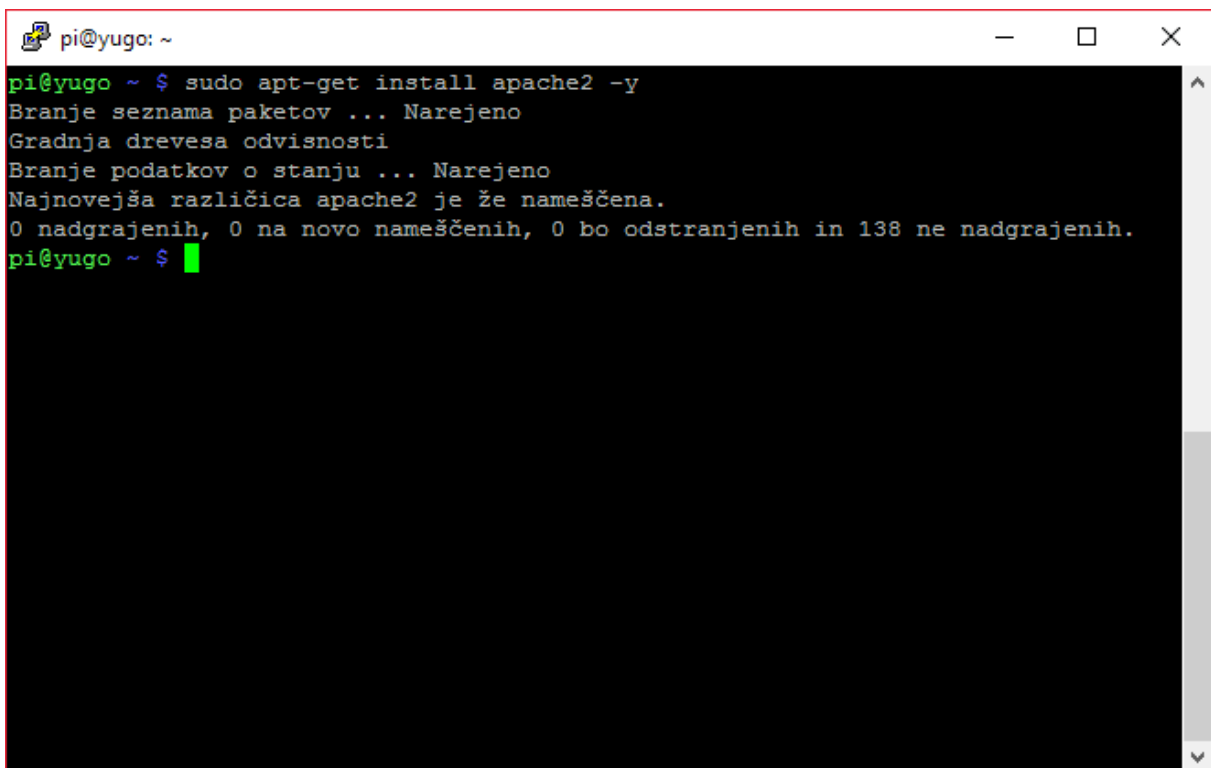


Slika 4: Spletna kamera

Avtomobilček, ki smo ga uporabili, deluje na frekvenci 2,4 GHz in podpira pošiljanje analognih vrednosti, kar omogoča naprednejšo vodljivost . Brezžični usmerjevalnik pa smo uporabili kot posrednik med Raspberry Pi-jem in spletno stranjo.

4.4 Raspberry Pi – Konfiguracija

Da smo lahko začeli z našo nalogo, smo najprej morali Raspberry Pi priključiti na monitor in na napajanje. Priključili smo tudi tipkovnico. Nato smo začeli s spreminjanjem nastavitev. Prva stvar, ki smo jo naredili, je bila posodobitev operacijskega sistema na Raspberry Pi-ju. Zaradi lažje uporabe smo spremenili osnove, kot so npr. časovni pas in razporeditev tipkovnice. Za tem smo namestili Apache strežnik, kateremu smo dodali PHP modul.

A screenshot of a terminal window titled 'pi@yugo: ~'. The window has standard window controls (minimize, maximize, close) in the top right corner. The terminal text shows the command 'sudo apt-get install apache2 -y' being executed. The output indicates that the package list is being updated, dependencies are being resolved, and the latest version of apache2 is being installed. It also shows that 0 packages will be upgraded, 0 will be newly installed, 0 will be removed, and 138 will not be upgraded. The prompt returns to 'pi@yugo ~ \$' with a green cursor.

```
pi@yugo ~ $ sudo apt-get install apache2 -y
Branje seznama paketov ... Narejeno
Gradnja drevesa odvisnosti
Branje podatkov o stanju ... Narejeno
Najnovejša različica apache2 je že nameščena.
0 nadgrajenih, 0 na novo nameščenih, 0 bo odstranjenih in 138 ne nadgrajenih.
pi@yugo ~ $
```

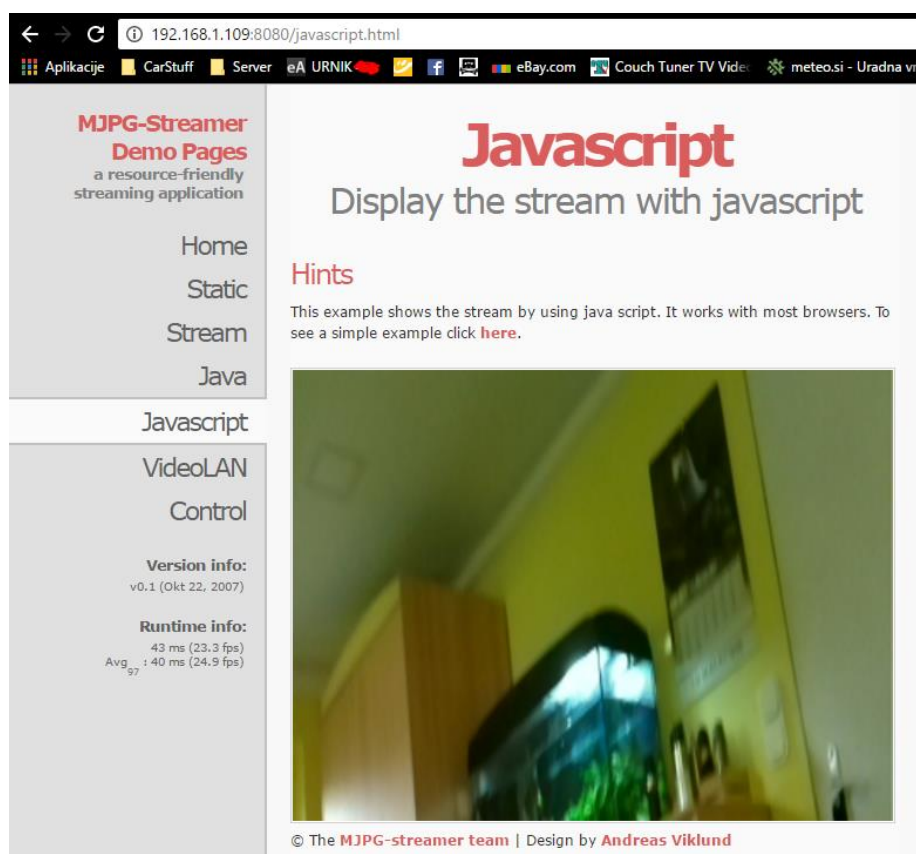
Slika 5: Nameščanje Apache strežnika

4.5 Nastavitve mjpeg streamer-ja in spletne kamere

Naložili smo še mjpeg streamer, ki je odprtokodni program in služi prenosu slike v realnem času, ki je vidna na spletni strani. V programu smo nastavili resolucijo (352x140px) in število slik na sekundo (25 fps). Poskušali smo različne resolucije, vendar smo se na koncu odločili za takšno vrednost, ker je pri višjih resolucijah prišlo do velikih zakasnitev zaradi omejene hitrosti WiPi modula (100 Mb/s).

Ukaz, ki smo ga uporabili za zagon mjpeg streamer-ja:

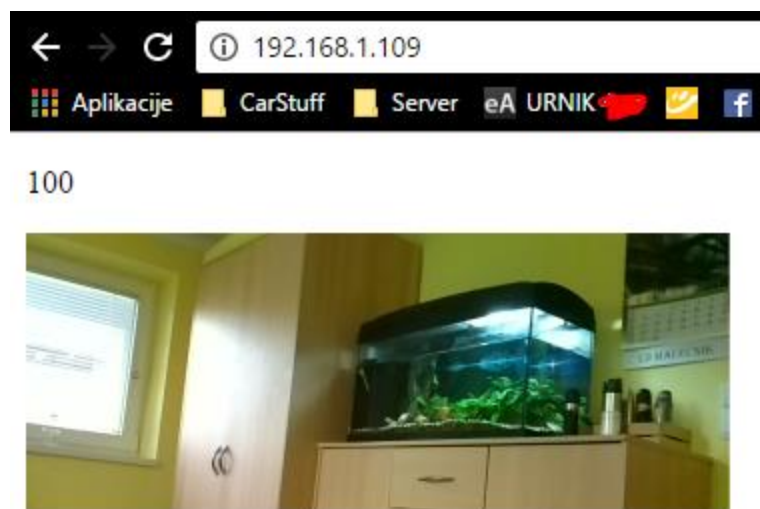
```
LD_LIBRARY_PATH=./ ./mjpeg_streamer -i "input_raspicam.so -fps 25 -x 352 -y 140" -o "output_http.so -w ./www"
```



Slika 6: Testiranje kamere

4.6 Kreiranje uporabniškega vmesnika (spletne strani)

Osnovo uporabniškega vmesnika smo naredili v Codepen-u. To je spletno okolje, v katerem lahko testiramo svoje spletne strani. Končni izdelek pa smo naredili na Raspberry Pi-ju s pomočjo Apache strežnika. Spletna stran je zgrajena iz mjpeg streamer-ja, kjer je prikazana slika (oz. kaže kam se vozi avtomobilček).



Slika 7: Izgled uporabniškega vmesnika (spletne strani)

Nad sliko pa so 3 številke, prva številka je vedno 1, in se uporablja za preverjanje povezave. Druga številka nam pove premikanje naprej ali nazaj. Če je druga številka 0, potem to pomeni da avtomobilček stoji na mestu, če je 1, potem se premika naprej (držimo smerno tipko za naprej), če pa je 2, pa se premika nazaj (smerna tipka za nazaj). Tretje število pa nam pove premikanje v levo ali desno. Če je 0, avtomobilček ne spreminja smeri vožnje, če je 1, potem se premika levo (smerna tipka za levo) in če je 2, pa se premika v desno (smerna tipka za desno).

4.7 Koda spletne strani

```
<!DOCTYPE html>
<html>
<head>

<script src="jquery-3.1.1.min.js"></script>
<script>
ld = 0;
nz = 0;
com = 100;
$(document).ready(function(){
TimerStart();
    $(document).keydown(function(e) {
    switch(e.which)
    {
        case 37: {ld = 1;break;}
        case 38: {nz = 1;break;}
        case 39: {ld = 2;break;}
        case 40: {nz = 2;break;}
        default: return; // exit this handler for other keys
    }
    Update();
    e.preventDefault(); // prevent the default action (scroll / move caret)
});

$(document).keyup(function(e) {
    switch(e.which)
    {
        case 37: {ld = 0;break;}
        case 38: {nz = 0;break;}
        case 39: {ld = 0;break;}
        case 40: {nz = 0;break;}
    }
    default: return; // exit this handler for other keys
    }
    Update();
    e.preventDefault(); // prevent the default action (scroll / move caret)
});

function Update()
{
```

```
com = ld + (nz*10)+100;  
$(".COM").text(com);
```

```
$.ajax({  
  url: "file.php",  
  type: "get", //send it through get method  
  data: {  
    move: com  
  }  
});  
}
```

```
function TimerStart()  
{  
  setTimeout(TimerStop,500);  
}
```

```
function TimerStop()  
{  
  Update();  
  TimerStart();  
}
```

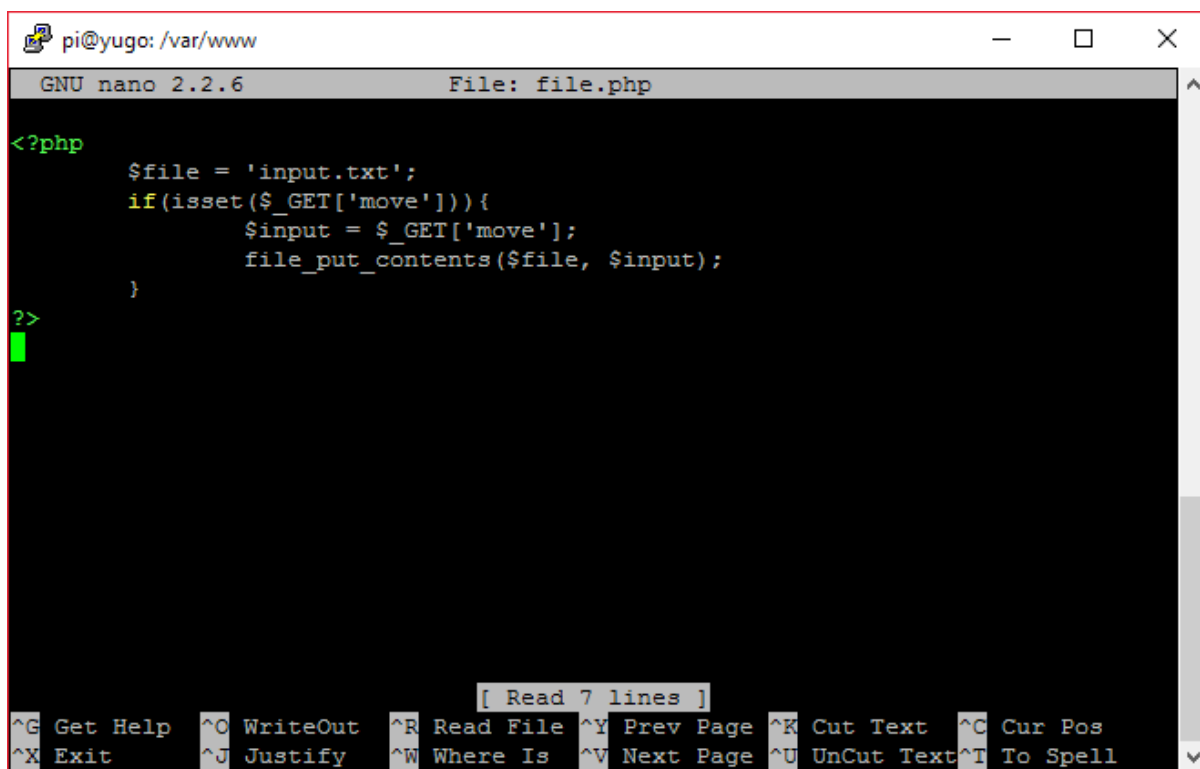
```
});  
</script>  
</head>  
<body>
```

```
<p class="COM">000</p>  

```

```
</body>  
</html>
```


Koda deluje tako, da ko pritisnesmo določeno tipko, npr. za naprej, JavaScript prepozna, katero tipko smo stisnili, in podatke o tej tipki z metodo get pošlje v datoteko file.php, ta pa ustvari tekstovno datoteko input.txt, v katero zapiše vse potrebne podatke. To datoteko kasneje naš program, napisan v programskem jeziku Python, obdela in ugotovi, v katero smer se naj avtomobilček premakne.



```
pi@yugo: /var/www
GNU nano 2.2.6 File: file.php

<?php
    $file = 'input.txt';
    if(isset($_GET['move'])){
        $input = $_GET['move'];
        file_put_contents($file, $input);
    }
?>
```

[Read 7 lines]

^G Get Help	^O WriteOut	^R Read File	^Y Prev Page	^K Cut Text	^C Cur Pos
^X Exit	^J Justify	^W Where Is	^V Next Page	^U UnCut Text	^T To Spell

Slika 8: Datoteka file.php

4.8 Koda za kontroliranje avtomobilčka

Kodo, namenjeno za premik avtomobilčka, smo napisali v jeziku Python.

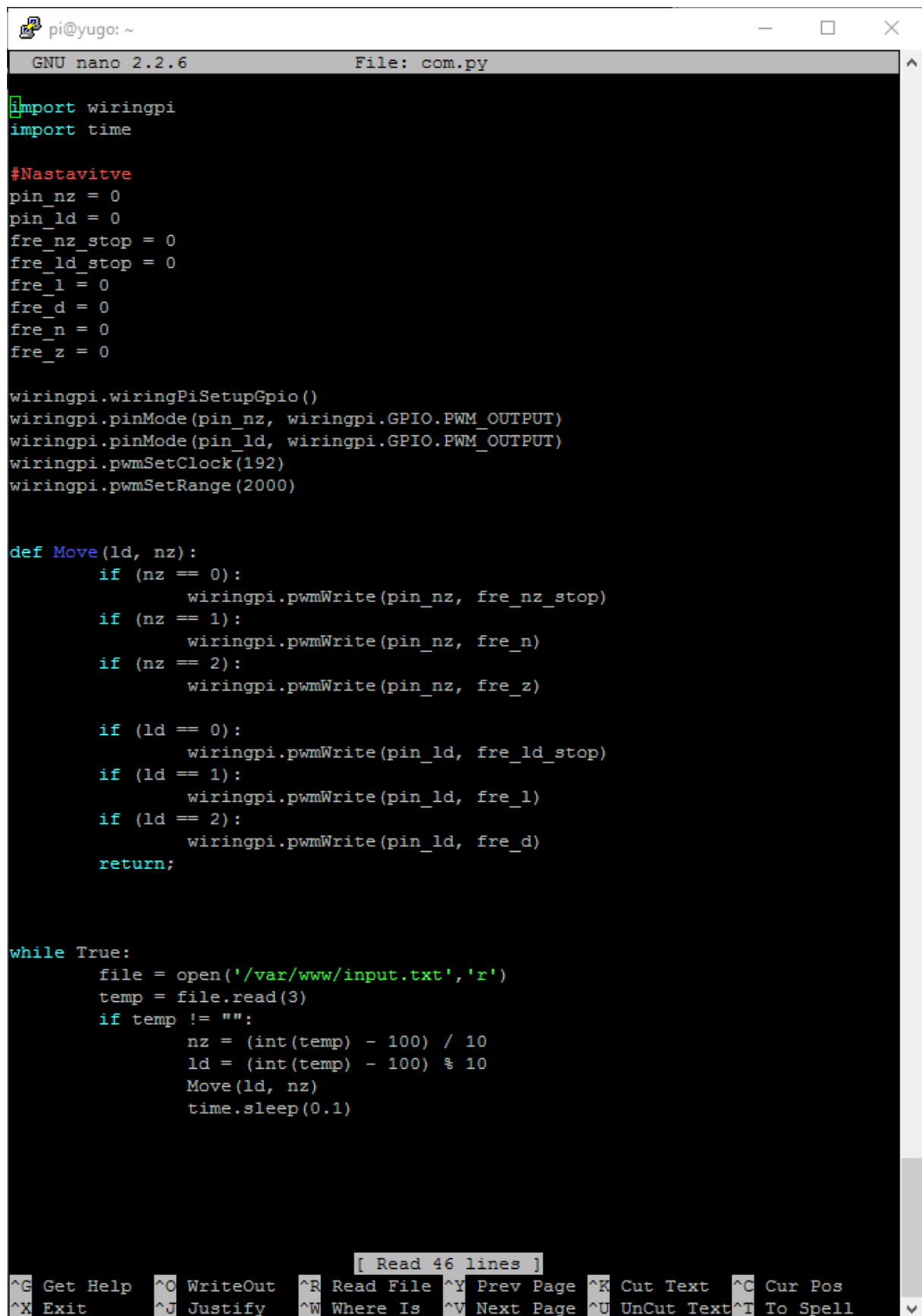
Python je programski jezik, ki ga je ustvaril Guido van Rossum leta 1990. Python ima dinamične podatkovne tipe, samodejno upravlja s pomnilnikom in podpira funkcionalno, imperativno oziroma proceduralno, strukturirano in objektno orientirano računalniško programsko paradigmo. Zaradi dinamičnih podatkovnih tipov je podoben jezikom Perl, Ruby, Scheme, Smalltalk,...

Ta programski jezik smo izbrali, ker omogoča tolmačenje kode v realnem času, kar nam omogoča optimizirano porabo centralne procesorske enote in ostalih virov (resource-ov). Zaradi tega lahko izvajamo zahtevnejše programe tudi na bolj omejeni strojni opremi.



Slika 9: Python logo

Deluje tako, da prebere datoteko input.txt, ki smo jo prej kreirali. Na podlagi podatkov v tekstovni datoteki, program pošlje ukaz avtomobilčku, da se naj premakne v določeno smer.



```
pi@yugo: ~  
GNU nano 2.2.6 File: com.py  
  
import wiringpi  
import time  
  
#Nastavitve  
pin_nz = 0  
pin_ld = 0  
fre_nz_stop = 0  
fre_ld_stop = 0  
fre_l = 0  
fre_d = 0  
fre_n = 0  
fre_z = 0  
  
wiringpi.wiringPiSetupGpio()  
wiringpi.pinMode(pin_nz, wiringpi.GPIO.PWM_OUTPUT)  
wiringpi.pinMode(pin_ld, wiringpi.GPIO.PWM_OUTPUT)  
wiringpi.pwmSetClock(192)  
wiringpi.pwmSetRange(2000)  
  
def Move(ld, nz):  
    if (nz == 0):  
        wiringpi.pwmWrite(pin_nz, fre_nz_stop)  
    if (nz == 1):  
        wiringpi.pwmWrite(pin_nz, fre_n)  
    if (nz == 2):  
        wiringpi.pwmWrite(pin_nz, fre_z)  
  
    if (ld == 0):  
        wiringpi.pwmWrite(pin_ld, fre_ld_stop)  
    if (ld == 1):  
        wiringpi.pwmWrite(pin_ld, fre_l)  
    if (ld == 2):  
        wiringpi.pwmWrite(pin_ld, fre_d)  
    return;  
  
while True:  
    file = open('/var/www/input.txt','r')  
    temp = file.read(3)  
    if temp != "":  
        nz = (int(temp) - 100) / 10  
        ld = (int(temp) - 100) % 10  
        Move(ld, nz)  
        time.sleep(0.1)  
  
[ Read 46 lines ]  
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos  
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

Slika 10: Python koda za premikanje avtomobilčka

V kodo smo vključili dve knjižnici, in sicer wiringpi in time. Wiringpi je namenjen kontroliranju servo motorjev, knjižnica time pa je namenjena za časovno zakasnitev dogodkov (npr. 300 ms). Deklarirali smo spremenljivke, katerim bomo kasneje priredili natančne vrednosti, glede na avtomobilček.

Vrstica `wiringpi.wiringPiSetupGpio()` inicializira knjižnico wiringpi. Z naslednjima vrsticama nastavimo izhode na output. Funkcija `Move` ima dve spremenljivki, `ld` in `nz`, `ld` je številka za levo/desno, `nz` pa naprej/nazaj. Na podlagi teh spremenljivk se avtomobilček nato premakne naprej, nazaj, levo ali desno. V glavnem delu kode se prebere tekstovna datoteka `input.txt`, nato se izračunata spremenljivki `nz` in `ld`, kateri nato pošljemo v zgoraj omenjeno funkcijo `Move`.

Računalnik, ki ga uporabljamo kot uporabniški vmesnik, mora biti v istem omrežju kot je avtomobilček. Za ta namen smo uporabili router, v katerem se nahajata računalnik in Raspberry Pi.

5. ZAKLJUČEK

Pri izdelavi te raziskovalne naloge smo pridobili veliko novega znanja s področja elektronike in programiranja Raspberry Pi-ja. Največji izziv nam je bila prilagoditev na urejanje kode v Raspberry Pi-ju, saj smo se z njim pri tej raziskovalni nalogi prvič srečali in nismo imeli nobenih predhodnih izkušenj. Pri uporabniškem vmesniku pa so nam pomagale izkušnje iz spletnega programiranja v jezikih HTML, Javascript in PHP. Izdelava izdelka nam je bila zelo zanimiva, saj smo lahko kombinirali znanje programiranja z strojno opremo. Tako smo lahko naredili izdelek, katerega spremembe niso vidne le na zaslonu računalnika, temveč tudi v realnem svetu.

6. DRUŽBENA ODGOVORNOST

V 21. stoletju sta postali zelo pomembni gospodarski panogi zabavna industrija in industrija igrač, kar pomeni, da ta sektor pridobiva vedno večjo vlogo v razvitem industrijskem svetu. V industriji igrač so zelo priljubljene daljinsko vodene igrače, zato bi lahko bil naš izdelek primeren za vsa podjetja, ki izdelujejo takšno vrsto igrač. Prav tako se daljinsko vodene naprave uporabljajo v vojski, medicini in za znanstvene raziskave, saj lahko z njihovo pomočjo dostopamo do nevarnih območji, brez da bi z tem poškodovali sebe ali druge osebe. Ker ima naš izdelek kamero, lahko vidimo dogajanje okoli avtomobilčka, brez da smo osebno prisotni, kar predstavlja veliko prednost na področju varnosti.

7. SEZNAM VIROV IN LITERATURE

Viri vsebine:

<http://wiringpi.com/pins/> (6.11.2016)

<http://projectedneuralactivity.blogspot.si/2012/12/controlling-servo-using-raspberry-pi.html> (12.11.2016)

<https://www.python.org/> (12.11.2016)

<https://codepen.io/> (15.11.2016)

[https://sl.wikipedia.org/wiki/Python_\(programski_jezik\)](https://sl.wikipedia.org/wiki/Python_(programski_jezik)) (20.11.2016)

Slikovni viri:

<https://images.hobbytron.com/ZX-9112C-lg.jpg> (6.11.2016)

<https://realpython.com/learn/python-first-steps/images/pythonlogo.jpg> (12.11.2016)

Osebni arhiv (12.12.2016)