Robot

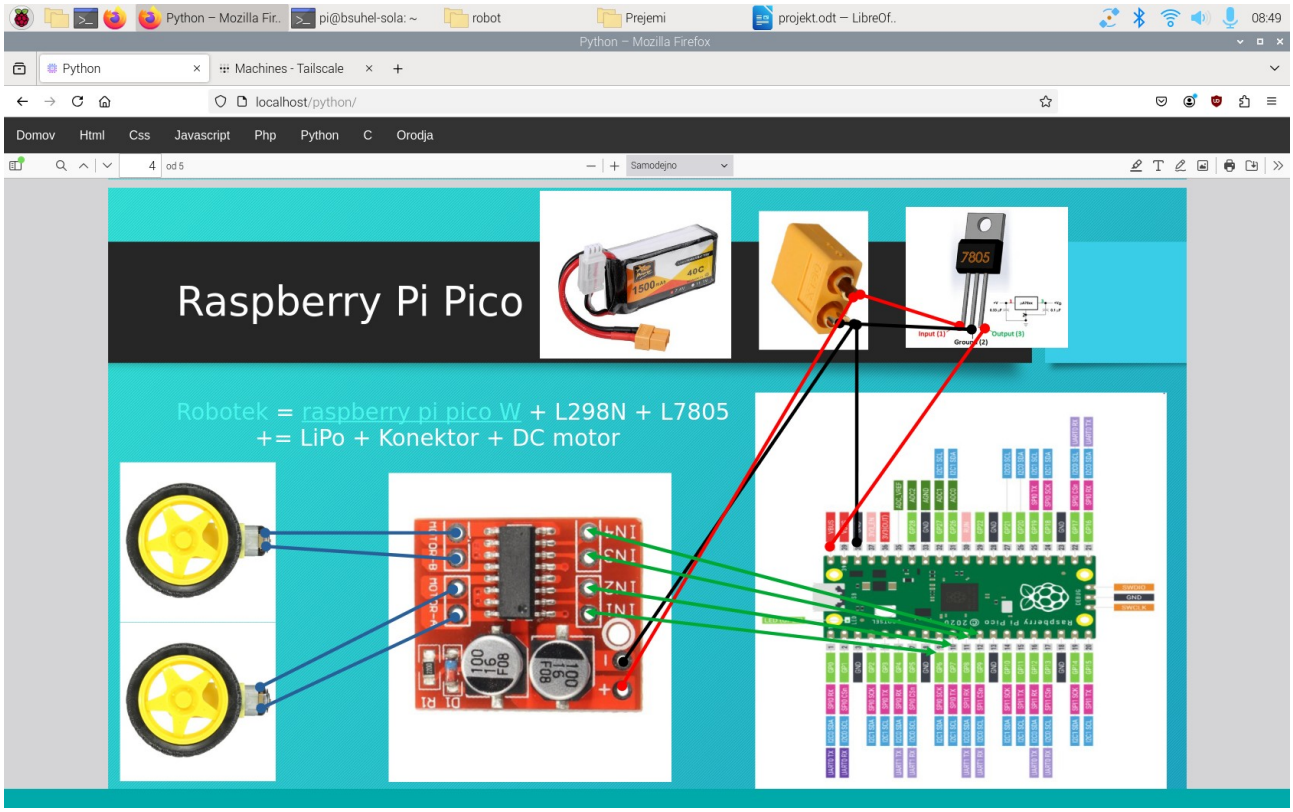
Avtor

Mag. Boštjan Šuhel

Kazalo vsebine

Elektronska oprema robota

Domov->Robot. Robot je delujoči šolski projekt. Šasijo robota, razvito v Tinkercad, natisnemo na 3D tiskalniku. Robot ima dva DC motorja z reduktorjem, H krmilnik za DC motorja, napajalnik, JeOS programski modul rp2040 in LiPo 2C baterijo 8,4V. Vodenje robota je možno s tipkovnico, nagibanjem pametnega telefona, s pomočjo rok(strojni vid) ali s pomočjo glasovnih ukazov.

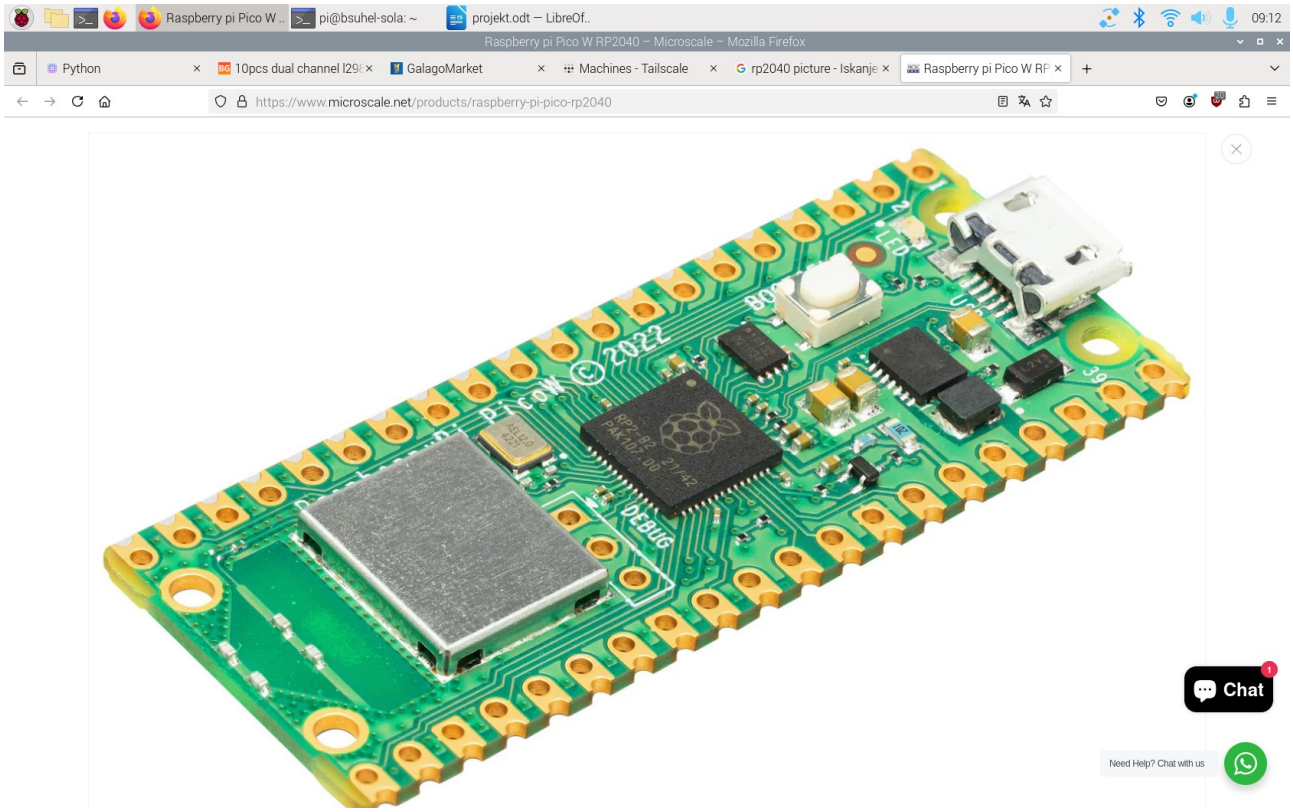


Slika 1: Vezalna shema robota

Programski modul RP2040 W

Raspberry Pi Pico W[Vir:1,2] je bil zasnovan kot poceni, a prilagodljiva razvojna platforma za RP2040 z brezžičnim vmesnikom 2,4 GHz in naslednjimi ključnimi funkcijami:
Mikrokrmilnik RP2040 z 2 MB flash pomnilnika
Vgrajeni enopasovni brezžični vmesniki 2,4 GHz (802.11n)
Vrata Micro USB B za napajanje in podatke (in za ponovno programiranje bliskavice)
40 zatičev 21 mm x 51 mm 'DIP' v slogu 1 mm debelega tiskanega vezja z 0,1-palčnimi zatiči s skoznjo luknjo, tudi z robovi.
26 večnamenskih 3,3 V V/I (GPIO)
23 GPIO je samo digitalnih, trije pa podpirajo tudi ADC
Lahko se namesti na površino kot modul
3-pinska vrata za odpravljanje napak serijske žice ARM (SWD).
Preprosta, a zelo prilagodljiva arhitektura napajanja
Različne možnosti za preprosto napajanje enote iz mikro USB, zunanjih napajalnikov ali baterij
Visoka kakovost, nizki stroški, visoka razpoložljivost
Dvojedrni Cortex M0+ do 133MHz
PLL na čipu omogoča spremenljivo frekvenco jedra
264kByte multi-bank visoko zmogljiv SRAM
Zunanji Quad-SPI Flash z eXecute In Place (XIP) in 16kByte predpomnilnika na čipu
Visoko zmogljiva tkanina vodila s polno prečko
Vgrajen USB1.1 (naprava ali gostitelj)

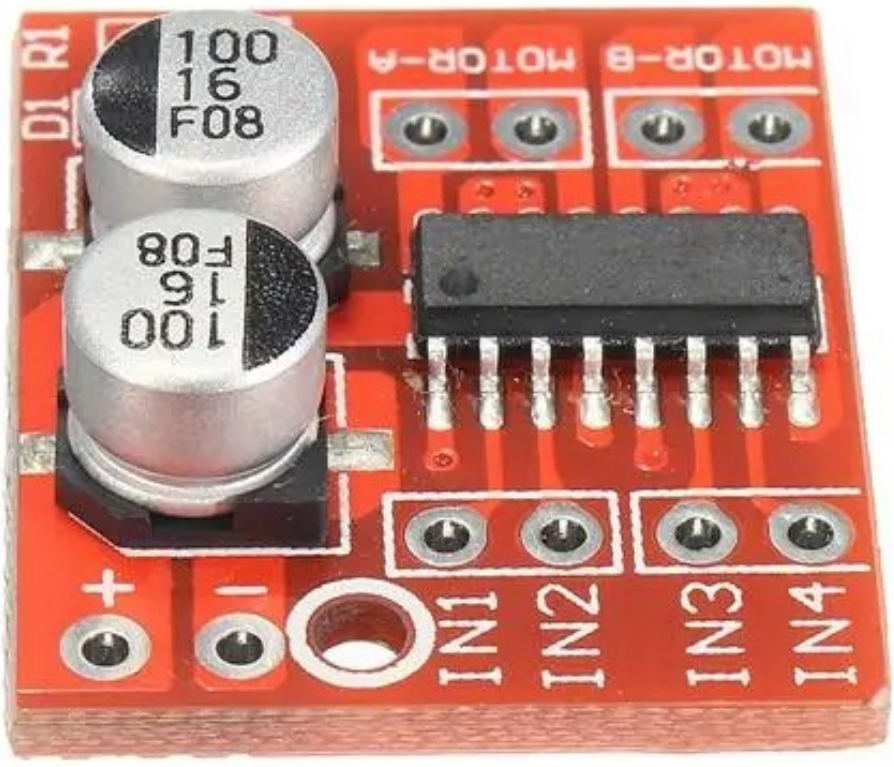
30 večnamenskih V/I za splošne namene (štiri se lahko uporabljajo za ADC)
V/I napetost 1,8–3,3 V
12-bitni 500ksps analogno-digitalni pretvornik (ADC)
Različne digitalne periferne naprave
2 × UART, 2 × I2C, 2 × SPI, 16 × kanali PWM
1 × časovnik s 4 alarmi, 1 × ura realnega časa
2 × programabilna V/I (PIO) bloka, skupaj 8 stanovskih avtomatov
Prilagodljiv hitri V/I, ki ga lahko programira uporabnik
Lahko posnema vmesnike, kot sta kartica SD in VGA
V/I napetost Raspberry Pi Pico W je fiksna na 3,3 V



Slika 2: Modul JeOS raspberry pico 2040 W

Motorski H krmilnik L280N

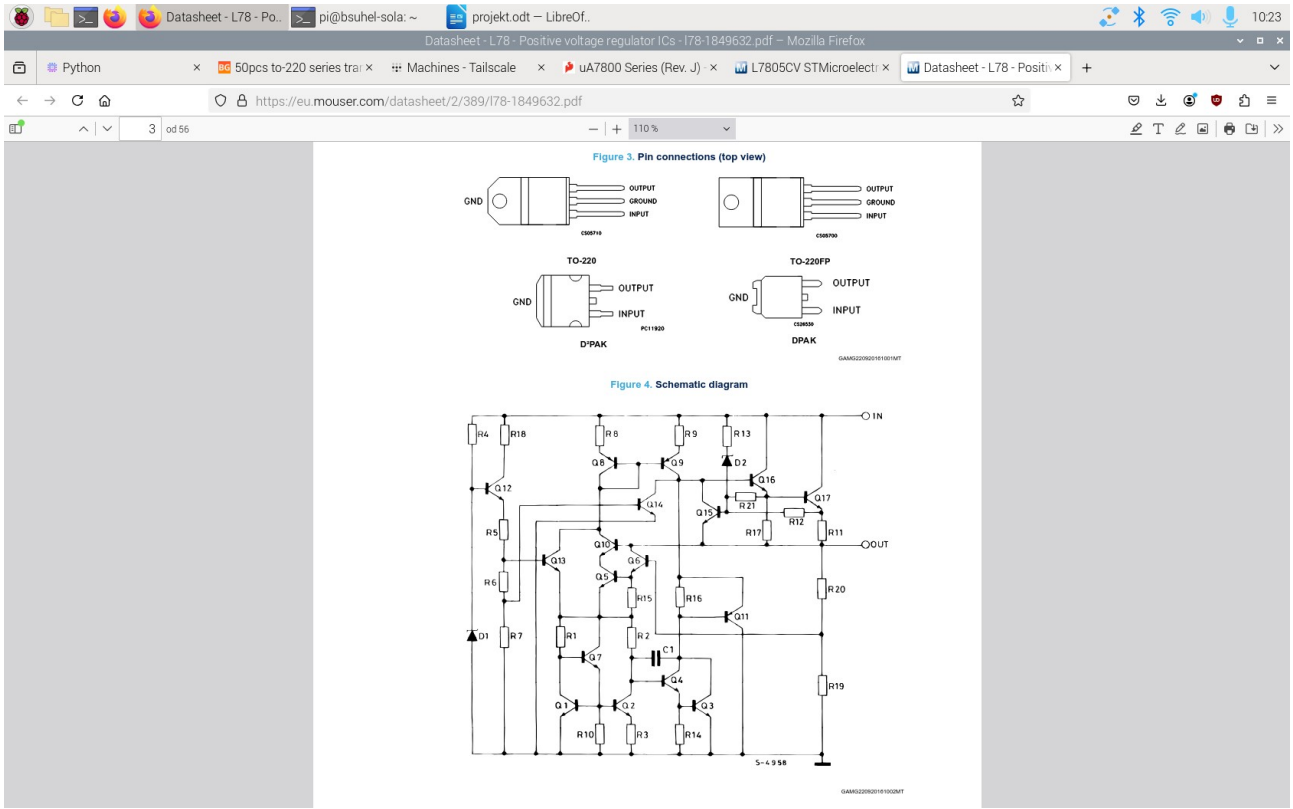
Dvojni H krmilnik motorja[Vir:3], lahko poganja dva enosmerna motorja ali 4-žilni dvofazni koračni motor
Vgrajen TSD, brez skrbi o zastoju motorja
Ultra-majhna velikost, primerna za montažo in avto
Napajalna napetost modula: DC 2V-10V
Vhodna napetost signala: DC 1,8-7 V
Enotni delovni tok: 1,5A
Temenski tok do 2,5 A
Nizek tok v stanju pripravljenosti (manj kot 0,1 uA)
Velikost: 24,7 x 21 x 7 mm
Premer montažne luknje: 2 mm



Slika 3: Modul dvojni H krmilnik L280N

Regulator napetosti L7805

Serija napetostnih regulatorjev[Vir:4,5,6] s fiksno napetostjo je zasnovana za široko paleto aplikacij. Vsak od regulatorjev lahko zagotovi do 1,5 A izhodnega toka. Tokovna in temperaturna zaščita jih naredijo imune na preobremenitev.
Poleg uporabe kot regulatorji fiksne napetosti se lahko uporabljajo z zunanjimi komponentami za nastavljive izhodne napetosti in tokove.



Slika 4: Napetostni regulator L7805

Akumulator LiPo

Uporabljena je 2 celična modelarska LiPo baterija nazivne napetosti 7,4V.
Parameter baterije[Vir:7]: ZOP Power 7.4V 1500mAh 40C 2S
Kapaciteta: 1500mAH
Velikost: 17*34*66MM
Stopnja neprekinjenega praznjenja: 40C
Teža: 78g
Barve: standardne barve
Opomba: zaradi razlike med meritvijo ima lahko dimenzija in teža baterije majhno napako

Slika 5: LiPo baterija

Priključek XT60

Blagovna znamka[Vir:8]: URUAV
Ime predmeta: XT60
Kovinski material: medenina pozlačena
Bullet priključek: 3,5 mm
Toplotno skrčljiva cev:
Premer: 3,5 mm
Barva: rdeča/črna

Slika 6: Bullet priključek XT60

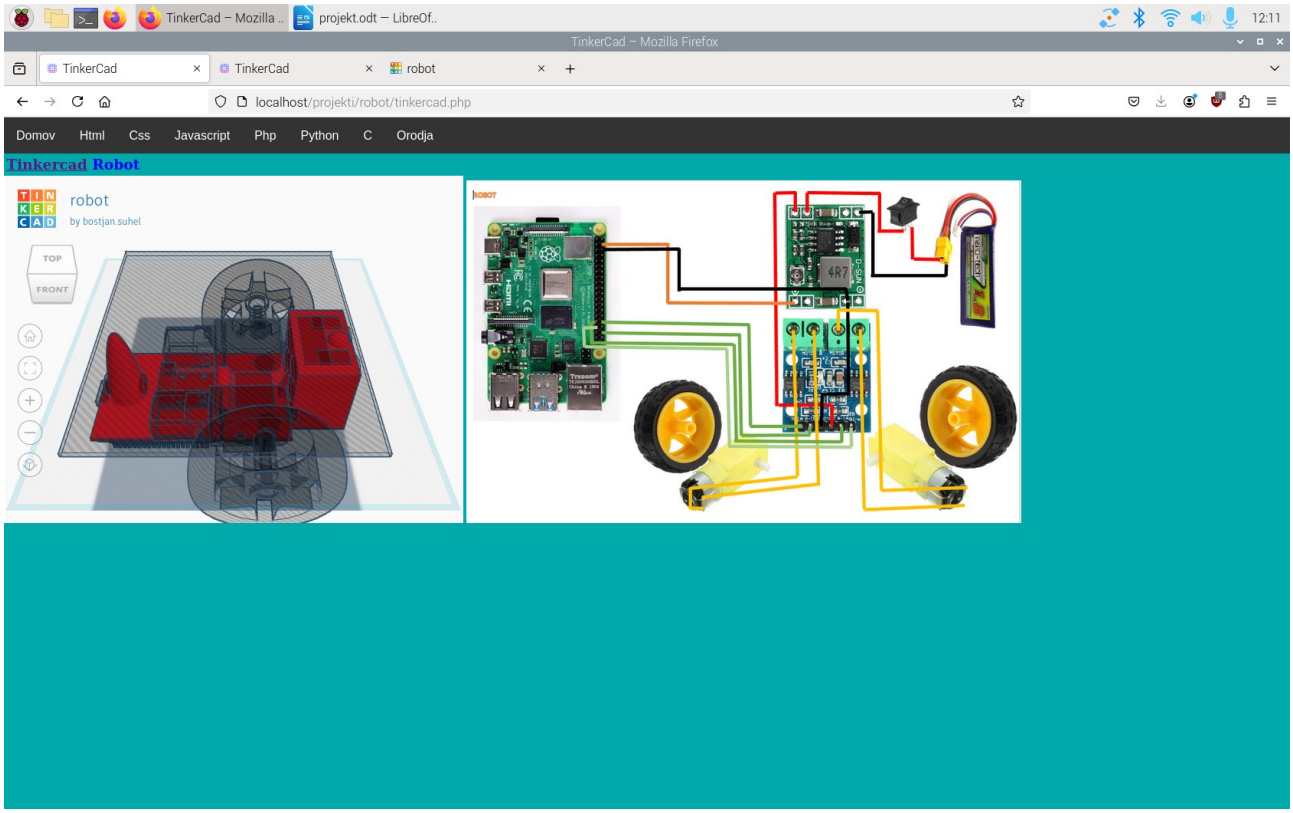
DC motor z reduktorjem in pnevmatiko

Motor[Vir:9]:
Napetost: DC 3V-6V
100 MA-120MA
Stopnja znižanja: 48: 1
RPM (s pnevmatiko): 100-240
Premer pnevmatike: 65 mm
Hitrost avtomobila (M/minuto): 20-48
Teža motorja (g): 29/kos
Velikost motorja: 70 mm X 22 mm X 18 mm
Hrup: <65dB
Pnevmatika:
Sredinska luknja: 5,3 mm x 3,66 mm
Velikost kolesa: 65 x 26 mm

Slika 7: DC motor z reduktorjem in pnevmatiko

Šasija robota

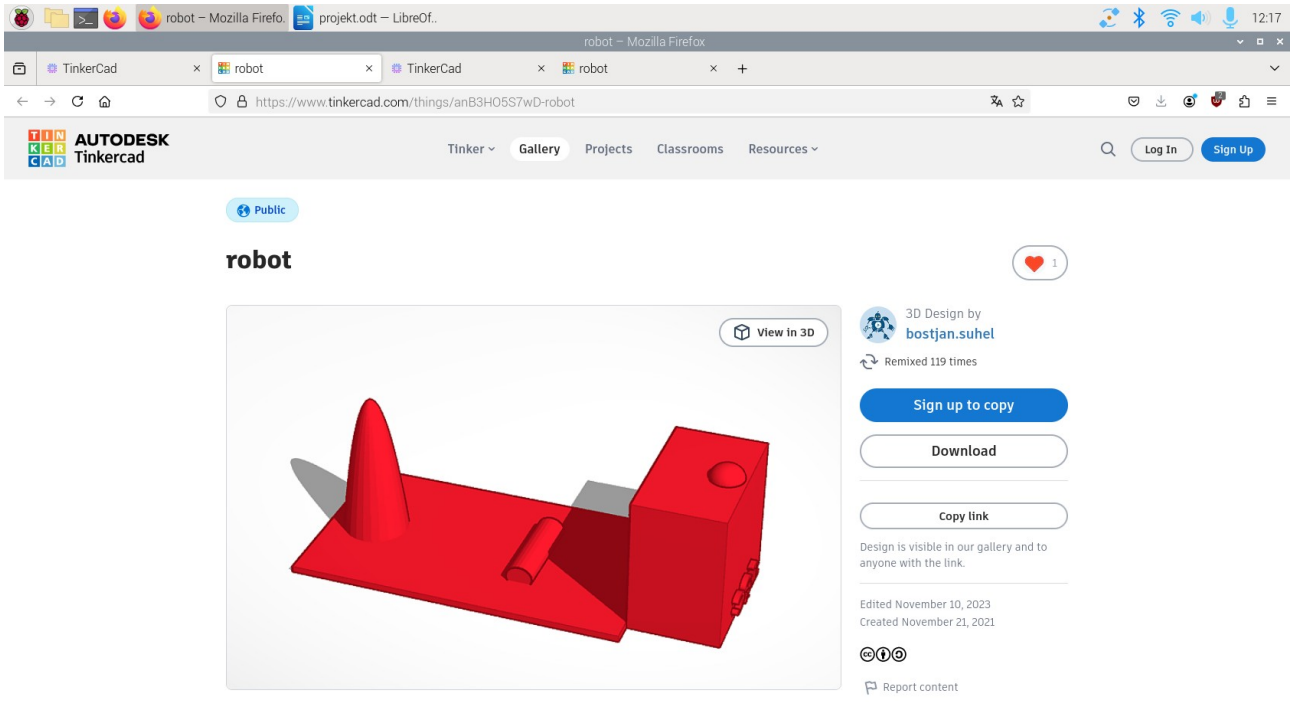
Domov->Robot->PHP->Python. Šasija robota je načrtovana v Tinkercad-u.



Slika 8: Slovenska izdaja s stranjo šasije robota

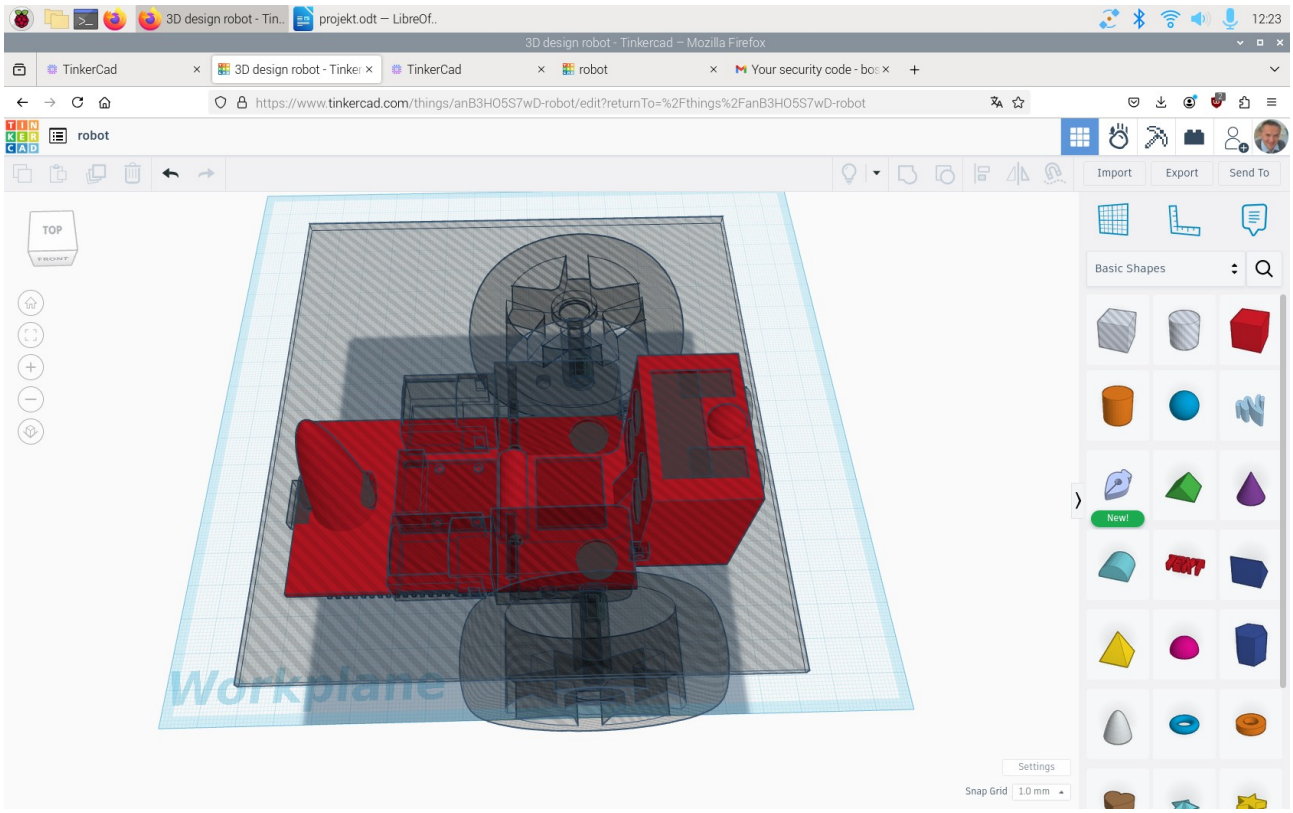
Tinkercad

Projekt tinkercad-u[Vir:10] šasije robota je javno dostopen. S kopijo lahko počnemo karkoli. Pri projektih navadno spreminjajo napis TŠČ in dodajajo razne nosilce za tipala in druge spremembe.



Slika 9: Šasija robota v tinkercad-u

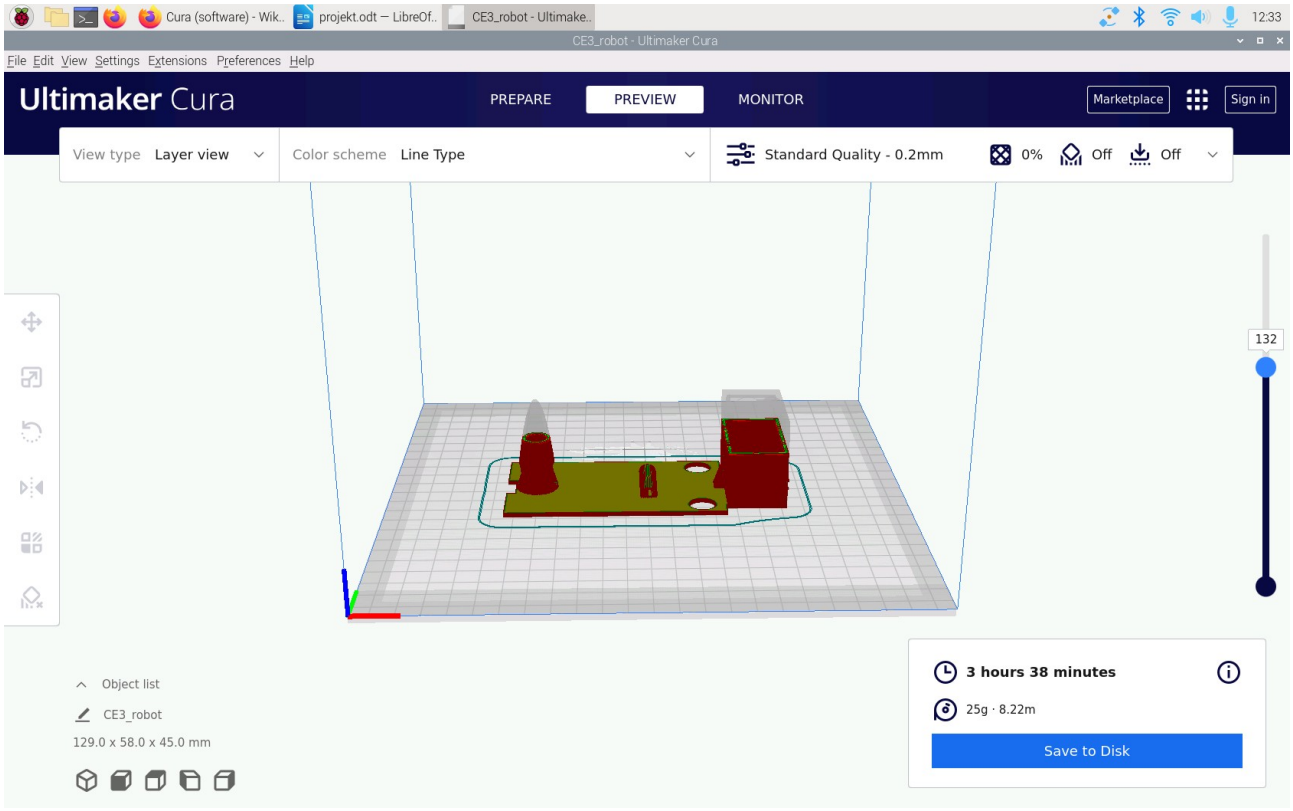
Odpremo 3D urejevalnik in popravimo originalno šasijo. Ko končamo izvozimo .stl datoteko.



Slika 10: 3D urejevalnik in izvoz stl datoteke

Cura

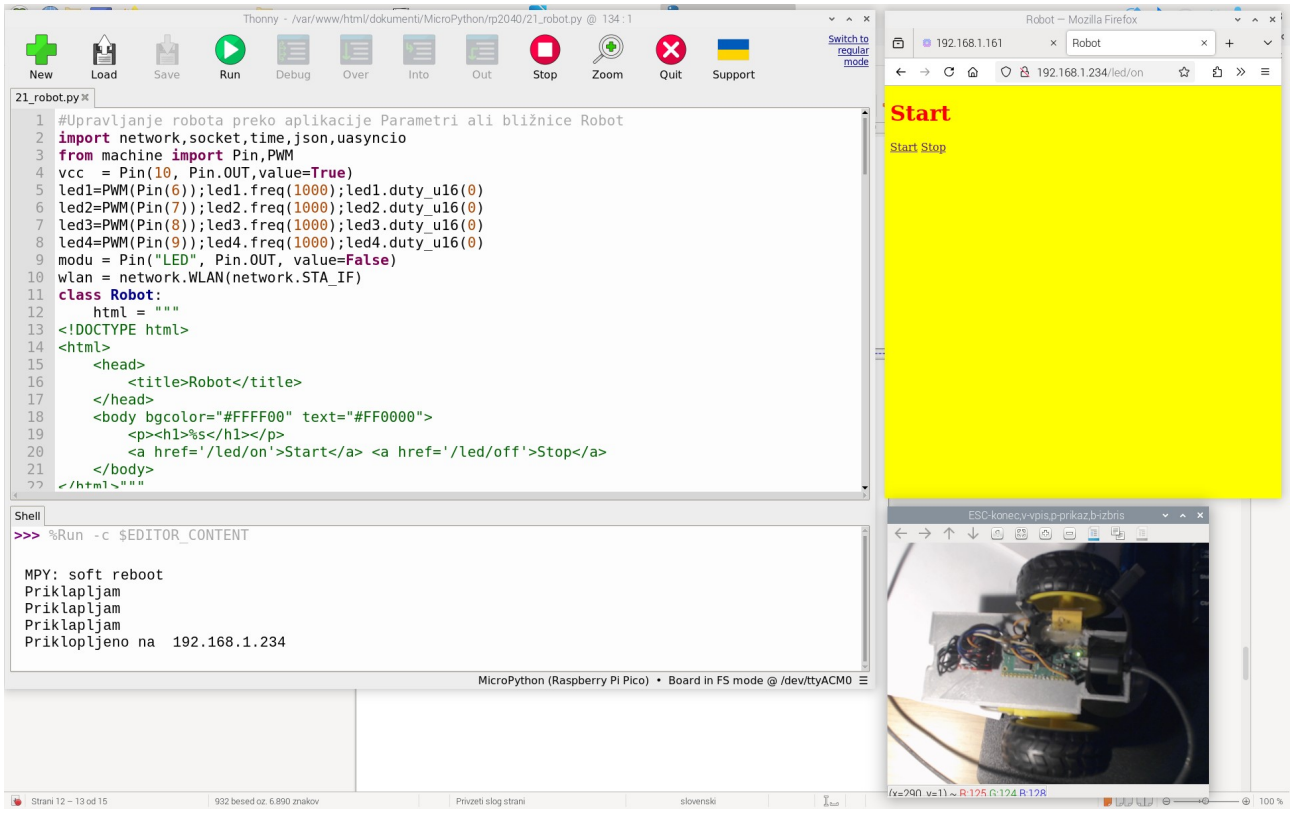
Cura[Vir:11] je program ki stl datoteko 3D urejevalnikov pretvori v g-kodo za CNC stroje. Program izračuna rezine(slice). Rezine si lahko pogledamo. Program nam izračuna potreben čas tiskanja za konkreteno šasijo in tiskalnik. Dobimo tudi informacijo o masi porabljenega filamenta. S to datoteko natisnemo šasijo na 3D tiskalniku.



Slika 11: Program Cura pretvori stl v gcode datoteko za 3D tiskalnike

Programska oprema robota

21_robot.py



Slika 12: 21_robot.py

Uporaba rp2040[Vir:1] je za v razred in razvoj idealna zaradi male porabe in dolgega delovanja na standardne modelarske dvo celične LiPo akumulatorje. Program 21_robot.py je pisan v Micropythonu[Vir:12].

```
test = Robot('pi','raspberry')
zanka = uasyncio.get_event_loop()
#zanka.run_until_complete(test.main('',20001))
zanka.create_task(test.main('',20001))
zanka.run_forever()
```

Konstrukcija primerka Test razreda Robot. Parametra pomenita ime in geslo dostopne točke na katero se priklopimo. Poženemo test metodo main. Zagotovimo še neskončno ponavljanje main(glavne) metode.

```
  def __init__(self,ssid,geslo):
    wlan.disconnect()
    wlan.active(True)
    wlan.config(pm = 0xa11140) # prepove power safe mode
    wlan.connect(ssid,geslo)
    while not wlan.isconnected():
      print('Priklapljam')
      modu.value(True)
      time.sleep(0.5)
      modu.value(False)
      time.sleep(0.5)
    print('Priklopljeno na ', wlan.ifconfig()[0])
```

Metoda def_init_ se izvede ob konstruiranju primerka objekta test.
Priklop na dostopno točko pi:raspberry se uspešno pravi, ko led modula preneha utripat.
Program nam sporoči IP naslov na katerega se je modul priklopil.

```
   async def tcp_strežnik(self,reader, writer):
      request_line = await reader.readline()
      while await reader.readline() != b"\r\n":
        pass
      request = str(request_line)
      led_on = request.find('/led/on')
      led_off = request.find('/led/off')
      status = ""
      if led_on == 6:
        modu.value(True)
        status = "Start"
      if led_off == 6:
        modu.value(False)
        status = "Stop"
      response = self.html % status
      writer.write('HTTP/1.0 200 OK\r\nContent-type: text/html\r\n\r\n')
      writer.write(response)
      await writer.drain()
      await writer.wait_closed()
```

Metoda tcp_strežnik je web strežnik, ki se odziva na vratih 80. Z brskalnikom dobimo odziv na tem IP naslovu. Spletna stran je enostavna in zgolj omogoča prižiganje in ugašanje led na modulu. Drug sklop je UDP strežnik na programskih vratih 20001.

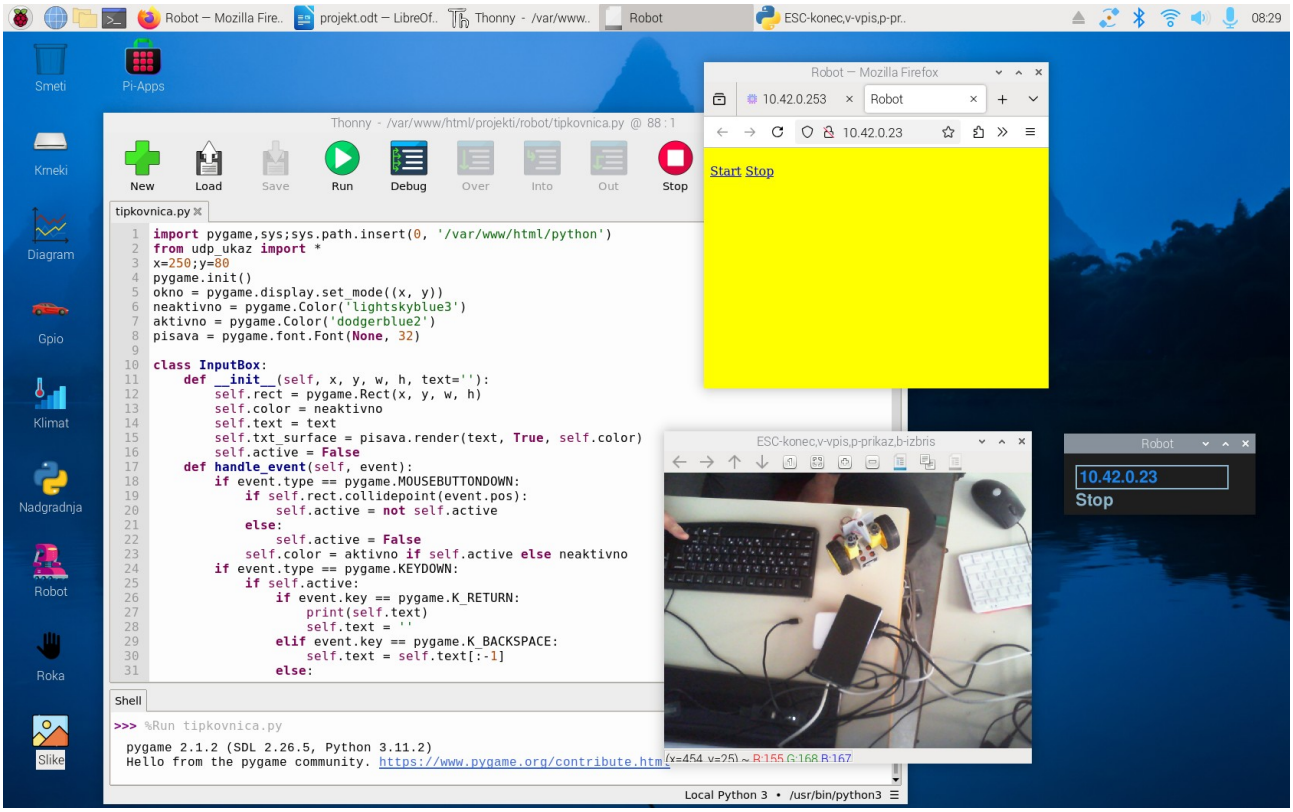
```
async def udp_strežnik(self):
  while True:
    await uasyncio.sleep_ms(5)
    try:
      data, self.naslov =  self.sock.recvfrom(1024)  # Prejmi UDP paket in podatke o viru
      uk = json.loads(data.decode())
      oddaja=json.dumps(uk)
      self.sock.sendto(oddaja,self.naslov)
      if uk['ukaz']=='g':#tipka gor naprej
        if uk['parameter']=='1':#tipka gor naprej
          led1.duty_u16(65000);led2.duty_u16(0)
          led3.duty_u16(65000);led4.duty_u16(0)
          modu.value(True)
        elif uk['parameter']=='3':#tipka desno
          led1.duty_u16(65000);led2.duty_u16(0)
          led3.duty_u16(0);led4.duty_u16(65000)
          modu.value(True)
        elif uk['parameter']=='4':#tipka  levo
          led1.duty_u16(0);led2.duty_u16(65000)
          led3.duty_u16(65000);led4.duty_u16(0)
          modu.value(True)
        elif uk['parameter']=='2':#tipka dol nazaj
          led1.duty_u16(0);led2.duty_u16(65000)
          led3.duty_u16(0);led4.duty_u16(65000)
          modu.value(True)
        elif uk['parameter']=='5':#konec
          led1.duty_u16(0);led2.duty_u16(0)
          led3.duty_u16(0);led4.duty_u16(0)
        elif uk['parameter']=='6':#Potrdi
          modu.value(False)
      elif uk['ukaz']=='p' and uk['index']==0:
          par=uk['vrednost']
          if modu.value()==True:
            self.naklon=par['pitch'];self.nagib=par['roll']
            a=self.n16_meje(self.naklon,-self.nagib,3000);led1.duty_u16(a)
            a=self.n16_meje(-self.naklon,self.nagib,3000);led2.duty_u16(a)
            a=self.n16_meje(self.naklon,self.nagib,3000);led3.duty_u16(a)
            a=self.n16_meje(-self.naklon,-self.nagib,3000);led4.duty_u16(a)
          else:
            led1.duty_u16(0);led2.duty_u16(0);led3.duty_u16(0);led4.duty_u16(0)
      elif uk['ukaz']=='s':#tipka Stop Start
          if uk['parameter']=='Stop':#tipka gor naprej
              led1.duty_u16(0);led2.duty_u16(0);led3.duty_u16(0);led4.duty_u16(0)
              modu.value(False)
          if uk['parameter']=='Start':#tipka gor naprej
              modu.value(True)
    except:
      pass
```

Metoda udp_strežnik čaka na udp telegrame na vratih 20001 in ustrezno reagira glede na vsebino telegrama. Telegram ima obliko slovarja. Ključ v slovarju ukaz ima vrednost g ali p.
Če je ukaz enak g ima parameter 1 do 9. Glede na parameter se nastavi izhode robota tako, da se leta premika v pravo smer. To se uporablja za sprejemanje ukazov iz tipkovnice.
Če je ukaz enak p in index enak 0 se prebere ključ pitch in roll. Vrednosti se zapiše v naklon in nagib, kar predstavlja usmeritev telefona. Te vrednosti uporabimo za ustrezno proporcionalno komando robotu. Robot usmerjamo z nagibanjem telefona.

```
async def main(self,ip, vrata):
  self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
  self.sock.bind((ip, vrata))
  self.sock.setblocking(False)
  #uasyncio.create_task(test.moja_prva_naloga())
  #uasyncio.create_task(test.moja_druga_naloga())
  uasyncio.create_task(test.udp_strežnik())
  uasyncio.create_task(uasyncio.start_server(test.tcp_strežnik, "0.0.0.0", 80))
```

Metoda main nastavi poslušanje UDP telegramov na vratih 20001 in požene metodo udp_strežnik in tcp_strežnik.

tipkovnica.py



Slika 13: Program tipkovnica, spletna stran in slika robota

Program tipkovnica zahteva vpis IP naslova robota. S pomočjo smernih tipk pošiljamp g ukaze robotu. V robotu imamo web in udp strežnik. Ti dve storitvi so značilne za internet stvari. Za šolo imamo vse odprto. Udp strežnik robota ustrezne g ukaze pretvori v krmiljenje motorjev.

```
class InputBox:
    def __init__(self, x, y, w, h, text=''):
        self.rect = pygame.Rect(x, y, w, h)
        self.color = neaktivno
        self.text = text
        self.txt_surface = pisava.render(text, True, self.color)
        self.active = False
    def handle_event(self, event):
        if event.type == pygame.MOUSEBUTTONDOWN:
            if self.rect.collidepoint(event.pos):
                self.active = not self.active
            else:
                self.active = False
            self.color = aktivno if self.active else neaktivno
        if event.type == pygame.KEYDOWN:
            if self.active:
                if event.key == pygame.K_RETURN:
                    print(self.text)
                    self.text = ''
                elif event.key == pygame.K_BACKSPACE:
                    self.text = self.text[:-1]
```

```
            else:
                self.text += event.unicode
            self.txt_surface = pisava.render(self.text, True, self.color)
    def update(self):
        width = max(200, self.txt_surface.get_width()+10)
        self.rect.w = width
    def draw(self, screen):
        screen.blit(self.txt_surface, (self.rect.x+5, self.rect.y+5))
        pygame.draw.rect(screen, self.color, self.rect, 2)
```

V programu Tipkovnica.py uporabimo knjižnico pygame in udp_ukaz. Pygame je znana python knjižnica za razvoj igric. Zaradi enostavne uporabe jo uporabimo tudi za programe vodenja. Razred Input box opiše metode in lastnosti vnosnega polja za IP naslove.

```
def main():
    clock = pygame.time.Clock()
    input_box1 = InputBox(15, 15, 15, 32)
    input_box1.text='127.0.0.1'
    input_boxes = [input_box1]
    pygame.display.set_caption('Robot');
    done = False;akcija='Stop'
    while not done:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                done = True
            for box in input_boxes:
                box.handle_event(event)
        for box in input_boxes:
            box.update()
        okno.fill((30, 30, 30))
        okno.blit(pisava.render(akcija, 1, neaktivno), (15, 50))
        for box in input_boxes:
            box.draw(okno)
        if not input_box1.active:
            try:
                if event.type == pygame.KEYDOWN:
                    if event.key == pygame.K_UP:#Naprej
                        hitri_g(input_box1.text,1);akcija='Naprej'
                    if event.key == pygame.K_DOWN:#Nazaj
                        hitri_g(input_box1.text,2);akcija='Nazaj'
                    if event.key == pygame.K_RIGHT:#Desno
                        hitri_g(input_box1.text,3);akcija='Desno'
                    if event.key == pygame.K_LEFT:#Levo
                        hitri_g(input_box1.text,4);akcija='Levo'
                    if event.key == pygame.K_SPACE:#Presledek
                        hitri_g(input_box1.text,6);akcija='Potrdi'

            except:
                i=1
        try:
```

```
            if event.type == pygame.KEYUP:
                hitri_g(input_box1.text,5);akcija='Stop'
        except:
            i=1
        pygame.display.flip()
        clock.tick(30)
```

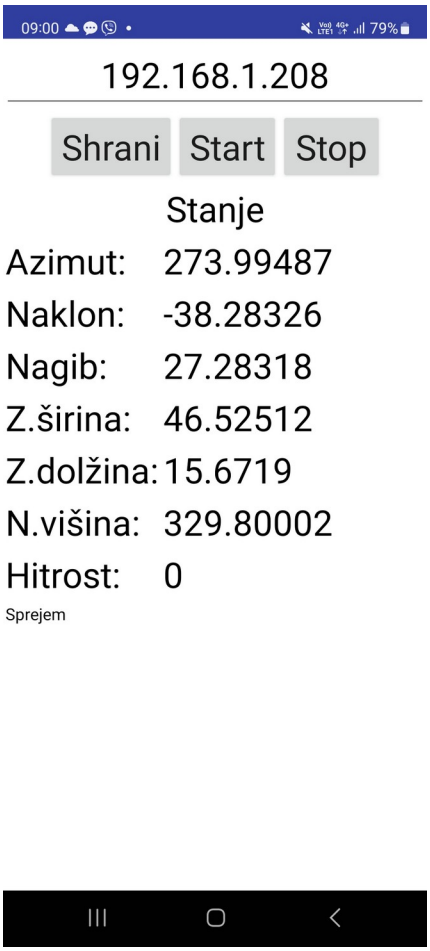
Metoda main se požene ob zagonu in preverja dogodke tipkovnice. S smernimi tipkami prožimo ustrezne g ukaze v metodi hitri_g. Hitri_g je metoda napisana za pošiljanje UDP telegramov brez čakanja na odziv robota. Metoda ima parameter IP naslov in parameter(g ukaza). IP naslov dobimo iz lastnosti input_box1.text, parameter g ukaza napišemo. UDP telegrami brez čakanja na odziv, so naj hitrejši način pošiljanja podatkov preko omrežja.

```
if __name__ == '__main__':
    main()
    pygame.quit()
```

Tu poženemo metodo main().

Tipala.aia

Razvojno orodje AppInventor[Vir:13] je primerno za pisanje android aplikacij. Aplikacija za android pametne telefone tipala.aia meri GPS, magnetno polje in pospeške. Izmerjene rezultate pošilja strukturirano v tabeli par(0). Udp strežnik robota prebere rezultate naklona in nagiba in ustrezno prilagodi krmiljenje robota.Prenašajo se številski rezultati meritev, ki jih krmilni del programa na robotu pretvori v ustrezne pulzne širine napajanja DC motorja. Robota vodimo z nagibom in naklonom telefona.



Slika 14: Android aplikacija tipala

Izvorna koda

/var/www/html/dokumenti/MicroPython/rp2040/21_robot.py

```
#Upravljanje robota preko aplikacije Parametri ali bližnice Robot
import network,socket,time,json,uasyncio
from machine import Pin,PWM
vcc  = Pin(10, Pin.OUT,value=True)
led1=PWM(Pin(6));led1.freq(1000);led1.duty_u16(0)
led2=PWM(Pin(7));led2.freq(1000);led2.duty_u16(0)
led3=PWM(Pin(8));led3.freq(1000);led3.duty_u16(0)
led4=PWM(Pin(9));led4.freq(1000);led4.duty_u16(0)
modu = Pin("LED", Pin.OUT, value=False)
wlan = network.WLAN(network.STA_IF)
class Robot:
    html = """
<!DOCTYPE html>
<html>
    <head>
        <title>Robot</title>
    </head>
    <body bgcolor="#FFFF00" text="#FF0000">
        <p><h1>%s</h1></p>
        <a href='/led/on'>Start</a> <a href='/led/off'>Stop</a>
    </body>
</html>"""
    naklon=0;nagib=0
    števec=0;naslov='127.0.0.1'
    def __init__(self,ssid,geslo):
      wlan.disconnect()
      wlan.active(True)
      wlan.config(pm = 0xa11140) # prepove power safe mode
      wlan.connect(ssid,geslo)
      while not wlan.isconnected():
        print('Priklapljam')
        modu.value(True)
        time.sleep(0.5)
        modu.value(False)
        time.sleep(0.5)
      print('Priklopljeno na ', wlan.ifconfig()[0])
    async def moja_prva_naloga(self):
        while test.števec<2:
            test.števec+=1
            print("Izvajanje prve naloge...")
            await uasyncio.sleep(1)
    async def moja_druga_naloga(self):
        while True:
            print('Azimut:'+str(self.azimut)+' Naklon:'+str(self.naklon)+' Nagib:'+str(self.nagib))
            await uasyncio.sleep(1)
    def n16_meje(self,x,y,i):
        z=(x+y)*i
```

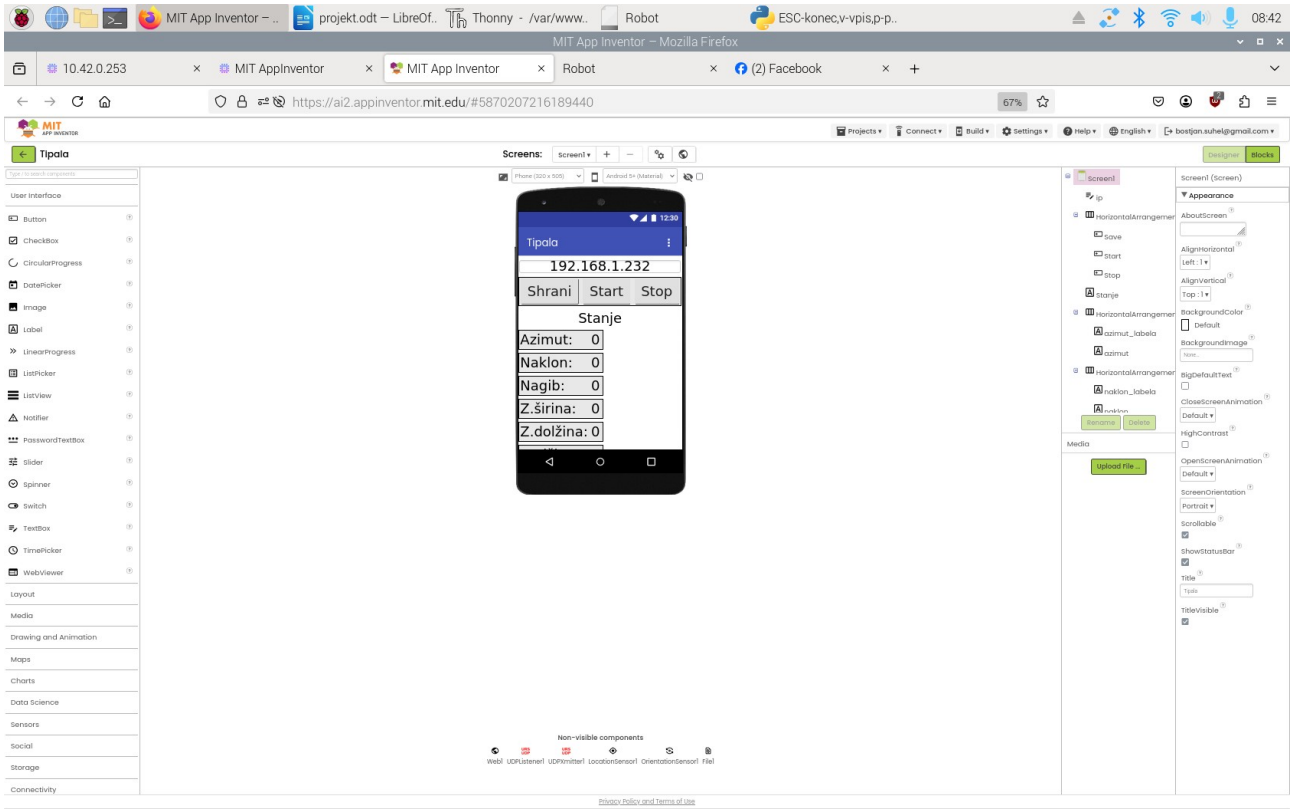
```
    if    z>65535:
       return int(65535)
    elif  z<=0:
       return int(0)
    else:
       return int(z)
async def udp_strežnik(self):
  while True:
    await uasyncio.sleep_ms(5)
    try:
      data, self.naslov =  self.sock.recvfrom(1024)  # Prejmi UDP paket in podatke o viru
      uk = json.loads(data.decode())
      oddaja=json.dumps(uk)
      self.sock.sendto(oddaja,self.naslov)
      if uk['ukaz']=='g':#tipka gor naprej
        if uk['parameter']=='1':#tipka gor naprej
          led1.duty_u16(65000);led2.duty_u16(0)
          led3.duty_u16(65000);led4.duty_u16(0)
          modu.value(True)
        elif uk['parameter']=='3':#tipka desno
          led1.duty_u16(65000);led2.duty_u16(0)
          led3.duty_u16(0);led4.duty_u16(65000)
          modu.value(True)
        elif uk['parameter']=='4':#tipka  levo
          led1.duty_u16(0);led2.duty_u16(65000)
          led3.duty_u16(65000);led4.duty_u16(0)
          modu.value(True)
        elif uk['parameter']=='2':#tipka dol nazaj
          led1.duty_u16(0);led2.duty_u16(65000)
          led3.duty_u16(0);led4.duty_u16(65000)
          modu.value(True)
        elif uk['parameter']=='5':#konec
          led1.duty_u16(0);led2.duty_u16(0)
          led3.duty_u16(0);led4.duty_u16(0)
        elif uk['parameter']=='6':#Potrdi
          modu.value(False)
      elif uk['ukaz']=='p' and uk['index']==0:
          par=uk['vrednost']
          if modu.value()==True:
            self.naklon=par['pitch'];self.nagib=par['roll']
            a=self.n16_meje(self.naklon,-self.nagib,3000);led1.duty_u16(a)
            a=self.n16_meje(-self.naklon,self.nagib,3000);led2.duty_u16(a)
            a=self.n16_meje(self.naklon,self.nagib,3000);led3.duty_u16(a)
            a=self.n16_meje(-self.naklon,-self.nagib,3000);led4.duty_u16(a)
          else:
            led1.duty_u16(0);led2.duty_u16(0);led3.duty_u16(0);led4.duty_u16(0)
      elif uk['ukaz']=='s':#tipka Stop Start
          if uk['parameter']=='Stop':#tipka gor naprej
              led1.duty_u16(0);led2.duty_u16(0);led3.duty_u16(0);led4.duty_u16(0)
              modu.value(False)
```

```
            if uk['parameter']=='Start':#tipka gor naprej
                modu.value(True)
        except:
          pass
    async def tcp_strežnik(self,reader, writer):
        request_line = await reader.readline()
        while await reader.readline() != b"\r\n":
          pass
        request = str(request_line)
        led_on = request.find('/led/on')
        led_off = request.find('/led/off')
        status = ""
        if led_on == 6:
          modu.value(True)
          status = "Start"
        if led_off == 6:
          modu.value(False)
          status = "Stop"
        response = self.html % status
        writer.write('HTTP/1.0 200 OK\r\nContent-type: text/html\r\n\r\n')
        writer.write(response)
        await writer.drain()
        await writer.wait_closed()
    async def main(self,ip, vrata):
      self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
      self.sock.bind((ip, vrata))
      self.sock.setblocking(False)
      #uasyncio.create_task(test.moja_prva_naloga())
      #uasyncio.create_task(test.moja_druga_naloga())
      uasyncio.create_task(test.udp_strežnik())
      uasyncio.create_task(uasyncio.start_server(test.tcp_strežnik, "0.0.0.0", 80))
test = Robot('pi','raspberry')
zanka = uasyncio.get_event_loop()
#zanka.run_until_complete(test.main('',20001))
zanka.create_task(test.main('',20001))
zanka.run_forever()
```

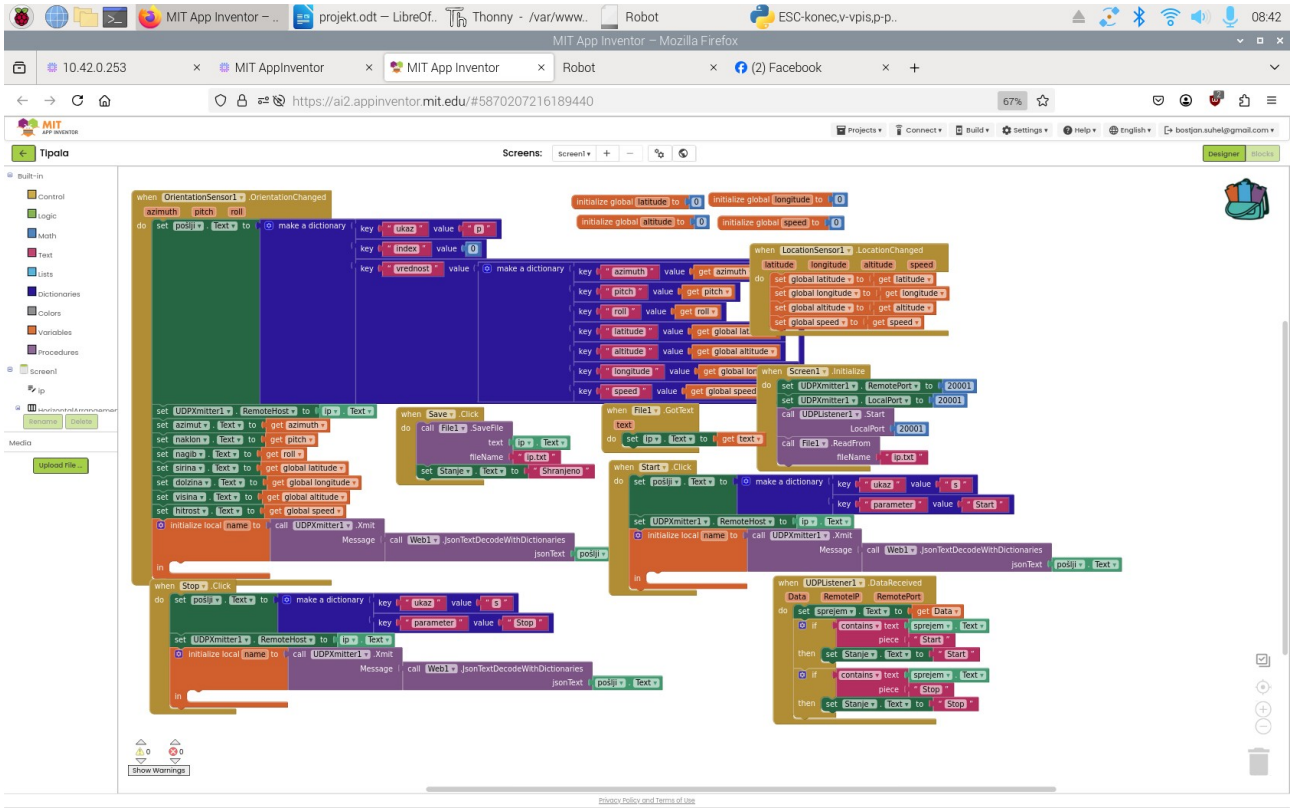
/var/www/html/projekti/robot/tipkovnica.py

```
import pygame,sys;sys.path.insert(0, '/var/www/html/python')
from udp_ukaz import *
x=250;y=80
pygame.init()
okno = pygame.display.set_mode((x, y))
neaktivno = pygame.Color('lightskyblue3')
aktivno = pygame.Color('dodgerblue2')
pisava = pygame.font.Font(None, 32)

class InputBox:
    def __init__(self, x, y, w, h, text=''):
        self.rect = pygame.Rect(x, y, w, h)
        self.color = neaktivno
        self.text = text
        self.txt_surface = pisava.render(text, True, self.color)
        self.active = False
    def handle_event(self, event):
        if event.type == pygame.MOUSEBUTTONDOWN:
            if self.rect.collidepoint(event.pos):
                self.active = not self.active
            else:
                self.active = False
            self.color = aktivno if self.active else neaktivno
        if event.type == pygame.KEYDOWN:
            if self.active:
                if event.key == pygame.K_RETURN:
                    print(self.text)
                    self.text = ''
                elif event.key == pygame.K_BACKSPACE:
                    self.text = self.text[:-1]
                else:
                    self.text += event.unicode
                self.txt_surface = pisava.render(self.text, True, self.color)
    def update(self):
        width = max(200, self.txt_surface.get_width()+10)
        self.rect.w = width
    def draw(self, screen):
        screen.blit(self.txt_surface, (self.rect.x+5, self.rect.y+5))
        pygame.draw.rect(screen, self.color, self.rect, 2)

def main():
    clock = pygame.time.Clock()
    input_box1 = InputBox(15, 15, 15, 32)
    input_box1.text='127.0.0.1'
    input_boxes = [input_box1]
    pygame.display.set_caption('Robot');
    done = False;akcija='Stop'
    while not done:
        for event in pygame.event.get():
```

```
            if event.type == pygame.QUIT:
                done = True
            for box in input_boxes:
                box.handle_event(event)
        for box in input_boxes:
            box.update()
        okno.fill((30, 30, 30))
        okno.blit(pisava.render(akcija, 1, neaktivno), (15, 50))
        for box in input_boxes:
            box.draw(okno)
        if not input_box1.active:
            try:
                if event.type == pygame.KEYDOWN:
                    if event.key == pygame.K_UP:#Naprej
                        hitri_g(input_box1.text,1);akcija='Naprej'
                    if event.key == pygame.K_DOWN:#Nazaj
                        hitri_g(input_box1.text,2);akcija='Nazaj'
                    if event.key == pygame.K_RIGHT:#Desno
                        hitri_g(input_box1.text,3);akcija='Desno'
                    if event.key == pygame.K_LEFT:#Levo
                        hitri_g(input_box1.text,4);akcija='Levo'
                    if event.key == pygame.K_SPACE:#Presledek
                        hitri_g(input_box1.text,6);akcija='Potrdi'

            except:
                i=1
        try:
            if event.type == pygame.KEYUP:
                hitri_g(input_box1.text,5);akcija='Stop'
        except:
            i=1
        pygame.display.flip()
        clock.tick(30)

if __name__ == '__main__':
    main()
    pygame.quit()
```



Slika 15: AppInventor Designer



Slika 16: AppInventor Blocks

Viri:

1. https://datasheets.raspberrypi.com/picow/pico-w-datasheet.pdf
2. https://www.galagomarket.com//item/display/2200
3. https://www.banggood.com/10pcs-Dual-Channel-L298N-DC-Motor-Driver-Board-PWM-Speed-Dual-H-Bridge-Stepper-Module-p-1171960.html
4. https://cdn.sparkfun.com/assets/1/7/7/3/2/LM7805.pdf
5. https://www.banggood.com/50PCS-TO-220-Series-Transistor-Assortment-Kit-10-Values-High-Power-Three-terminal-Regulated-Triode-LM317T-L7805-L7806-L7808-L7809-L7810-L7812-L7815-L7818-L7824-p-1980239.html?cur_warehouse=CN
6. https://eu.mouser.com/datasheet/2/389/l78-1849632.pdf
7. https://www.banggood.com/5Pcs-ZOP-Power-7_4V-1500mAh-40C-2S-Lipo-Battery-XT60-Plug-p-1762830.html
8. https://www.banggood.com/10-Pair-URUAV-XT60-Male-Female-Connectors-Power-Plugs-with-Heat-Shrink-Tube-for-Lipo-Battery-p-1396830.html
9. https://www.banggood.com/65-x-26mm-Plastic-Tire-Wheel-+-DC-3-6v-Gear-Motor-For-Smart-Car-p-997113.html
10. https://en.wikipedia.org/wiki/Tinkercad
11. https://en.wikipedia.org/wiki/Cura_(software)
12. https://micropython.org/
13. http://appinventor.mit.edu/

Kazalo slik