UNIWERSYTET PEDAGOGICZNY im. KEN w KRAKOWIE
INSTYTUT INFORMATYKI

Projekt inżynierski: Dokumentacja kodu

III rok Informatyka studia stacjonarne

16 czerwca 2021

Spis treści

1 Wybrany temat

Inteligentny robot typu Line-Follower

2 Zespół projektowy

Krzysztof Mazurek (krzysztof.mazurek@student.up.krakow.pl)
Mateusz Michalec (mateusz.michalec1@student.up.krakow.pl)

3 Krótki opis

Kontroler został napisany w Pythonie. Obsługuje on elementy robota e-puck takie jak: dwa czujniki ir i kamerę. Dodatkowo została dodana biblioteka OpenCV, która analizuje obraz z kamery. Kontroler umożliwia poruszanie się robota po narysowanej ścieżce, jak również analizę obrazu do dostosowania odpowiedniej prędkości na prostej i zakręcie. Analizowany jest obraz z kamery o rozdzielczości 640x640, który jest przedzielony na pół. Suma pikseli umożliwiająca jazdę z maksymalną prędkością musi wynosić powyżej 103000 na całym obrazie.

4 Analiza kodu

Implementacja bibliotek

```
from controller import Robot
from controller import Camera
from controller import DistanceSensor, Motor
import time
import cv2
import numpy as np
import sys
from controller import Display
```

Implementacja bibliotek potrzebnych do stworzenia robota
Funkcja do poruszania się robota

```
def run_robot(robot):
    time_step = 32
    step = -1
    #motors
    left_motor = robot.getDevice('left wheel motor')
    right_motor = robot.getDevice('right wheel motor')
    left_motor.setPosition(float('inf'))
    right_motor.setPosition(float('inf'))
    left_motor.setVelocity(0.0)
    right_motor.setVelocity(0.0)

    while robot.step(time_step) != -1:
        left_ir_value = left_ir.getValue()
        right_ir_value = right_ir.getValue()
        left_speed = max_speed
        right_speed = max_speed

        if (left_ir_value > right_ir_value) and (6 < left_ir_value < 15):
```

```
        left_speed = -max_speed
    elif (right_ir_value > left_ir_value) and (5 < right_ir_value < 15):
        right_speed = -max_speed

    left_motor.setVelocity(left_speed)
    right_motor.setVelocity(right_speed)
```

Pobranie od robota nazwy urządzeń do porusznia, następnie ustawienie wektora prędkości. Pętla "while" odpowiada za poruszanie sie robota ustawienie predosci maksymalnej na prostej oraz do sketu robota poprzez zapytanie "if" z odpowiednimi parametrami
Aktywacja sensorów

```
#irsensors
    left_ir = robot.getDevice('ir0')
    left_ir.enable(time_step)

    right_ir = robot.getDevice('ir1')
    right_ir.enable(time_step)
```

Przypisanie sensorów do zmiennych i ich aktywacja
Włączenie kamery

```
#camera
    camera = Camera('camera1')
    camera.enable(time_step)
```

Przypisanie kamery do zmiennej oraz włączenie jej.
Analiza obrazu za pomocą biblioteki OpenCV

```
    camera.saveImage("cam.png", 20)
    icv = cv2.imread("cam.png")
    imCrop = icv[ 0:640, 0:640]
    bw = icv[ 0:640, 0:640]
    width = camera.getWidth()
    height = camera.getHeight()
    x1, y1 = 318, 0
    x2, y2 = 318, 640
    cv2.line(imCrop, (x1, y1), (x2, y2), (0, 0, 255), thickness=4)
    (thresh, blackAndWhiteImage) = cv2.threshold(bw, 127, 255, cv2.THRESH_BIN
    cv2.imwrite( 'bw.png', blackAndWhiteImage)
    bwcount = cv2.imread("bw.png", 0)
    left_px = bwcount[:, 0:319]
    right_px = bwcount[:, 320:640]

    left_black = 204160- cv2.countNonZero(left_px)
    right_black = 204800 - cv2.countNonZero(right_px)
    print ("Black px on left: ", left_black, " on right px: ", right_black)
```

Kamera zapisuje obraz o rozdzielczosci 640x640 następie pobiera go i dzieli na poł następnie zmienia jego color na biało czarny i zapisauje pod nazwa "bw.png". Kolejno wczytuje ten obraz i przypisuje jedną połowę do zmiennej "left$_px$"$idrugdozmiennej$"$right_px$".$Nakoncuwypisujeilepxznajdujesiepolewejiprawej.$

Dostosowanie odpowiedniej prędkości

```
    if(right_black+left_black==0):
        max_speed = 1.5
```

```
        elif((right_black==0) or (left_black==0)):
            max_speed = 2
        else:
            max_speed = 6.28
```

Poprzez zastosowanie funkcji warunkowej zostaje przypisana prędokość robota. Jeśli robot nie widzi lini to zwalnia do 1.5, jeśli w którejś części nie widać lini zwalnia do 2, natomiast kiedy jest prosta jedzie z maksymalną prędkościa 6.28

Funkcja main

```
    if __name__ == "__main__":

    my_robot = Robot()
    run_robot(my_robot)
```

Deklaracja robota i wywołanie go

Cały kod

```
from controller import Robot
from controller import Camera
from controller import DistanceSensor, Motor
import time
import cv2
import numpy as np
import sys

def run_robot(robot):
    time_step = 32
    step = -1

    #camera
    camera = Camera('camera1')
    camera.enable(time_step)

    #motors
    left_motor = robot.getDevice('left wheel motor')
    right_motor = robot.getDevice('right wheel motor')
    left_motor.setPosition(float('inf'))
    right_motor.setPosition(float('inf'))
    left_motor.setVelocity(0.0)
    right_motor.setVelocity(0.0)

    #irsensors
    left_ir = robot.getDevice('ir0')
    left_ir.enable(time_step)

    right_ir = robot.getDevice('ir1')
    right_ir.enable(time_step)

    # Step simulation
    while robot.step(time_step) != -1:

        left_ir_value = left_ir.getValue()
```

```
        right_ir_value = right_ir.getValue()

        camera.saveImage("cam.png", 20)
        icv = cv2.imread("cam.png")
        imCrop = icv[ 0:640, 0:640]
        bw = icv[ 0:640, 0:640]
        width = camera.getWidth()
        height = camera.getHeight()
        x1, y1 = 318, 0
        x2, y2 = 318, 640
        cv2.line(imCrop, (x1, y1), (x2, y2), (0, 0, 255), thickness=4)
        (thresh, blackAndWhiteImage) = cv2.threshold(bw, 127, 255, cv2.THRESH_BIN
        cv2.imwrite( 'bw.png', blackAndWhiteImage)
        bwcount = cv2.imread("bw.png", 0)
        left_px = bwcount[:, 0:319]
        right_px = bwcount[:, 320:640]

        left_black = 204160- cv2.countNonZero(left_px)
        right_black = 204800 - cv2.countNonZero(right_px)
        print ("Black px on left: ", left_black, " on right px: ", right_black)

        if(right_black + left_black > 103000):
            max_speed = 6.28
        else:
            max_speed = 2.4

        left_speed = max_speed
        right_speed = max_speed

        if (left_ir_value > right_ir_value) and (6 < left_ir_value < 15):
            left_speed = -max_speed
        elif (right_ir_value > left_ir_value) and (5 < right_ir_value < 15):
            right_speed = -max_speed

        left_motor.setVelocity(left_speed)
        right_motor.setVelocity(right_speed)
        print("Speed: ", max_speed)

if __name__ == "__main__":

    my_robot = Robot()
    run_robot(my_robot)
```