|  |  |
| --- | --- |
|  | |
| **Detekcia farby objektov a ich triedenie pre**  **My Cubot 280** | |
|  | |
| Ján Lušňák | |
|  | |
|  |  |
| Seminárna práca AK2LO  2024 | Popis: fai_logo_cz |
|  |  |
|  | |

1. [1 Úvod 3](#_Toc21052)
2. [2 Kód v Jazyku python 4](#_Toc4691)
3. [3 Vysvetlenie funkcionality kódu 9](#_Toc19097)

[3.1.1 Import knižnic 9](#_Toc11227)

[3.1.2 Inicializácia - Nastavenie počiatočnej polohy robota 9](#_Toc20167)

[3.1.3 Funkcia pre detekciu farby: 9](#_Toc8987)

[3.1.4 Inicializácia kamery: 9](#_Toc31559)

[3.1.5 Inicializácia robotického ramena: 9](#_Toc3580)

[3.2 Funkcie pre manipuláciu s uchopovačom - Gripperom 10](#_Toc29724)

[3.2.1 Zatvorenie uchopovača: 10](#_Toc22323)

[3.2.2 Otvorenie uchopovača: 10](#_Toc23806)

[3.3 Funkcia pre pohyb ramena a manipuláciu s objektmi 10](#_Toc12316)

[3.4 Hlavný cyklus programu 12](#_Toc29752)

1. [4 Popis funkcionality kódu za sebou 15](#_Toc28337)
2. [5 Záver 17](#_Toc27980)
3. [Seznam použité literatury 18](#_Toc8215)

# Úvod

Cieľom mojej seminárnej práce je naprogramovať v jazyku Python kód ktorý naučí robota My Cubot 280 rozpoznavať farby objektov a následne triedit tieto objekty podľa farby a premiestňovať ich na vopred určene miesta do kontajnerov .

# Kód v Jazyku python

import cv2

import numpy as np

from pymycobot.mycobot import MyCobot # Import knižnice pre robotické rameno

import time

pociatok = [15.55,(-17.26),(-141.92),79.98,16.17,(-23.73)]

# Funkcia pre detekciu farby

def detect\_color(hsv, lower\_bound, upper\_bound):

mask = cv2.inRange(hsv, lower\_bound, upper\_bound)

return mask

# Inicializácia kamery

cap = cv2.VideoCapture(0)

# Inicializácia robotického ramena

# Pripojenie ramena k PC pomocou USB portu

robot\_arm = MyCobot("COM3", 115200) # Príklad pripojenia cez USB port OS Windows

# Funkcie pre manipuláciu s uchopovačom

def close\_gripper():

robot\_arm.set\_gripper\_state(1, 50)

def open\_gripper():

robot\_arm.set\_gripper\_state(0, 50)

# Funkcia pre pohyb ramena a manipuláciu s objektmi

def move\_robot\_to\_sort(color):

print(color)

try:

open\_gripper()

# Nastavenie polohy pre zdvihnutie objektu

pickup\_angles = pociatok # Zmeniť podľa potreby

robot\_arm.send\_angles(pickup\_angles, 80)

time.sleep(2)

close\_gripper()

time.sleep(2)

print('stop')

# Zvýšenie výšky pre premiestnenie objektu

pickup\_angles[1] = pickup\_angles[1] + 30

robot\_arm.send\_angles(pickup\_angles, 80)

time.sleep(2)

target\_angles = pickup\_angles.copy()

pickup\_angles[1] = pickup\_angles[1] - 30

# Nastavenie cieľovej polohy podľa farby

if color == 'red':

target\_angles[0] = 50

elif color == 'green':

target\_angles[0] = 80

elif color == 'blue':

target\_angles[0] = 110

# Presun objektu do cieľovej polohy

robot\_arm.send\_angles(target\_angles, 80)

time.sleep(2)

# Pustenie objektu

open\_gripper()

time.sleep(2)

target\_angles[0] = 15

robot\_arm.send\_angles(target\_angles, 80)

time.sleep(2)

# Návrat do počiatočnej polohy

robot\_arm.send\_angles(pociatok, 80)

time.sleep(2)

except Exception as e:

print(f"An error occurred in move\_robot\_to\_sort: {e}")

try:

robot\_arm.send\_angles(pociatok, 80)

time.sleep(5)

print("som na zaciatku !")

while True:

ret, origframe = cap.read()

print(origframe.shape) # Print image shape

cv2.imshow("original", origframe)

frame = origframe[120:350, 100:200]

if not ret:

break

hsv = cv2.cvtColor(frame, cv2.COLOR\_BGR2HSV)

lower\_red = np.array([0, 120, 70])

upper\_red = np.array([10, 255, 255])

mask\_red = detect\_color(hsv, lower\_red, upper\_red)

lower\_green = np.array([36, 100, 100])

upper\_green = np.array([86, 255, 255])

mask\_green = detect\_color(hsv, lower\_green, upper\_green)

lower\_blue = np.array([94, 80, 2])

upper\_blue = np.array([126, 255, 255])

mask\_blue = detect\_color(hsv, lower\_blue, upper\_blue)

contours\_red, \_ = cv2.findContours(mask\_red, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

contours\_green, \_ = cv2.findContours(mask\_green, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

contours\_blue, \_ = cv2.findContours(mask\_blue, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

for contour in contours\_red:

if cv2.contourArea(contour) > 500:

x, y, w, h = cv2.boundingRect(contour)

cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 0, 255), 2)

move\_robot\_to\_sort('red')

for contour in contours\_green:

if cv2.contourArea(contour) > 500:

x, y, w, h = cv2.boundingRect(contour)

cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)

move\_robot\_to\_sort('green')

for contour in contours\_blue:

if cv2.contourArea(contour) > 500:

x, y, w, h = cv2.boundingRect(contour)

cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 0), 2)

move\_robot\_to\_sort('blue')

cv2.imshow('Frame', frame)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

except KeyboardInterrupt:

print("Program interrupted by user.")

robot\_arm.release\_all\_servos()

finally:

cap.release()

cv2.destroyAllWindows()

robot\_arm.release\_all\_servos()

# Vysvetlenie funkcionality kódu

### Import knižnic

import cv2

import numpy as np

from pymycobot.mycobot import MyCobot # Import knižnice pre robotické rameno

import time

### Inicializácia - Nastavenie počiatočnej polohy robota

pociatok = [15.55,(-17.26),(-141.92),79.98,16.17,(-23.73)]

### ****Funkcia pre detekciu farby****:

# Funkcia pre detekciu farby

def detect\_color(hsv, lower\_bound, upper\_bound):

mask = cv2.inRange(hsv, lower\_bound, upper\_bound)

return mask

### ****Inicializácia kamery****:

cap = cv2.VideoCapture(0)

### ****Inicializácia robotického ramena****:

robot\_arm = MyCobot("COM3", 115200) # Príklad pripojenia cez USB port OS Windows

## Funkcie pre manipuláciu s uchopovačom - Gripperom

### ****Zatvorenie uchopovača****:

def close\_gripper():

robot\_arm.set\_gripper\_state(1, 50)

### ****Otvorenie uchopovača****:

def open\_gripper():

robot\_arm.set\_gripper\_state(0, 50)

## Funkcia pre pohyb ramena a manipuláciu s objektmi

def move\_robot\_to\_sort(color):

print(color)

try:

open\_gripper()

# Nastavenie polohy pre zdvihnutie objektu

pickup\_angles = pociatok # Zmeniť podľa potreby

robot\_arm.send\_angles(pickup\_angles, 80)

time.sleep(2)

close\_gripper()

time.sleep(2)

print('stop')

# Zvýšenie výšky pre premiestnenie objektu

pickup\_angles[1] = pickup\_angles[1] + 30

robot\_arm.send\_angles(pickup\_angles, 80)

time.sleep(2)

target\_angles = pickup\_angles.copy()

pickup\_angles[1] = pickup\_angles[1] - 30

# Nastavenie cieľovej polohy podľa farby

if color == 'red':

target\_angles[0] = 50

elif color == 'green':

target\_angles[0] = 80

elif color == 'blue':

target\_angles[0] = 110

# Presun objektu do cieľovej polohy

robot\_arm.send\_angles(target\_angles, 80)

time.sleep(2)

# Pustenie objektu

open\_gripper()

time.sleep(2)

target\_angles[0] = 15

robot\_arm.send\_angles(target\_angles, 80)

time.sleep(2)

# Návrat do počiatočnej polohy

robot\_arm.send\_angles(pociatok, 80)

time.sleep(2)

except Exception as e:

print(f"An error occurred in move\_robot\_to\_sort: {e}")

## Hlavný cyklus programu

try:

robot\_arm.send\_angles(pociatok, 80)

time.sleep(5)

print("som na zaciatku !")

while True:

ret, origframe = cap.read()

print(origframe.shape) # Print image shape

cv2.imshow("original", origframe)

frame = origframe[120:350, 100:200]

if not ret:

break

hsv = cv2.cvtColor(frame, cv2.COLOR\_BGR2HSV)

lower\_red = np.array([0, 120, 70])

upper\_red = np.array([10, 255, 255])

mask\_red = detect\_color(hsv, lower\_red, upper\_red)

lower\_green = np.array([36, 100, 100])

upper\_green = np.array([86, 255, 255])

mask\_green = detect\_color(hsv, lower\_green, upper\_green)

lower\_blue = np.array([94, 80, 2])

upper\_blue = np.array([126, 255, 255])

mask\_blue = detect\_color(hsv, lower\_blue, upper\_blue)

contours\_red, \_ = cv2.findContours(mask\_red, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

contours\_green, \_ = cv2.findContours(mask\_green, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

contours\_blue, \_ = cv2.findContours(mask\_blue, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

for contour in contours\_red:

if cv2.contourArea(contour) > 500:

x, y, w, h = cv2.boundingRect(contour)

cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 0, 255), 2)

move\_robot\_to\_sort('red')

for contour in contours\_green:

if cv2.contourArea(contour) > 500:

x, y, w, h = cv2.boundingRect(contour)

cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)

move\_robot\_to\_sort('green')

for contour in contours\_blue:

if cv2.contourArea(contour) > 500:

x, y, w, h = cv2.boundingRect(contour)

cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 0), 2)

move\_robot\_to\_sort('blue')

cv2.imshow('Frame', frame)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

except KeyboardInterrupt:

print("Program interrupted by user.")

robot\_arm.release\_all\_servos()

finally:

cap.release()

cv2.destroyAllWindows()

robot\_arm.release\_all\_servos()

# Popis funkcionality kódu za sebou

1. **Náčítanie rámu z kamery.** ret, origframe = cap.read()
2. ****Konverzia do HSV farebného priestoru**:** hsv = cv2.cvtColor(frame, cv2.COLOR\_BGR2HSV)
3. ****Detekcia jednotlivých farieb pomocou masiek**:**

lower\_red = np.array([0, 120, 70])

upper\_red = np.array([10, 255, 255])

mask\_red = detect\_color(hsv, lower\_red, upper\_red)

lower\_green = np.array([36, 100, 100])

upper\_green = np.array([86, 255, 255])

mask\_green = detect\_color(hsv, lower\_green, upper\_green)

lower\_blue = np.array([94, 80, 2])

upper\_blue = np.array([126, 255, 255])

mask\_blue = detect\_color(hsv, lower\_blue, upper\_blue)

1. **Nájdenie kontúr pre každú farbu a vykreslenie obrysov**:

contours\_red, \_ = cv2.findContours(mask\_red, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

contours\_green, \_ = cv2.findContours(mask\_green, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

contours\_blue, \_ = cv2.findContours(mask\_blue, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

for contour in contours\_red:

if cv2.contourArea(contour) > 500:

x, y, w, h = cv2.boundingRect(contour)

cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 0, 255), 2)

move\_robot\_to\_sort('red')

for contour in contours\_green:

if cv2.contourArea(contour) > 500:

x, y, w, h = cv2.boundingRect(contour)

cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)

move\_robot\_to\_sort('green')

for contour in contours\_blue:

if cv2.contourArea(contour) > 500:

x, y, w, h = cv2.boundingRect(contour)

cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 0), 2)

move\_robot\_to\_sort('blue')

1. **Zobrazenie orezaneho rámu kamery ktorý detekuje objekty**

cv2.imshow('Frame', frame)

1. Keď kamera nájde objekt a priradi ku nemu príslušnu farbu, spusti sa funkcia def move\_robot\_to\_sort(color): - otvorí sa gripper ktorý uchopí tento objekt a premiestni ho do polohy vopred určeneho kontajneru a otvorí gripper
2. ****Ukončenie programu pri stlačení klávesy 'q'**:**

if cv2.waitKey(1) & 0xFF == ord('q'):

Break

1. **Ukončenie programu a uvoľnenie zdrojov**

except KeyboardInterrupt:

print("Program interrupted by user.")

robot\_arm.release\_all\_servos()

finally:

cap.release()

cv2.destroyAllWindows()

robot\_arm.release\_all\_servos()

# Záver

Táto práca demonštruje úspešnú integráciu počítačového videnia a robotiky za účelom automatizovanej detekcie a triedenia objektov na základe ich farby. Použitím OpenCV pre spracovanie obrazu a MyCobot pre manipuláciu s objektmi sme vytvorili systém, ktorý je schopný rozpoznať farebné objekty, zdvihnúť ich a presunúť do príslušných kontajnerov.

Seznam použité literatury

[1]https://docs.elephantrobotics.com/docs/gitbook-en/7-ApplicationBasePython/7.2\_angle.html