Extra-úloha VIR

Voronov Serhii

8.11.2021

1 Uvod

Aktuálně pracuji v Testbedu společnosti Czech Institute of Informatics, Robotics and Cybernetics (CIIRC CTU), kde zabyvam se vyvojem elektroniky a firmwaru na industriální roboty. Jedna ze sub-úloh byla vývoj softwaru který by spolupracoval s robotem Kukou, kamerou a dopravníkem. Takže use-case je takový: na dopravníku jednou různé balíčky se součástky (třeba jako v krabicích z LEGO kde jsou N malých balíku různé velikosti), kamera detekuje velikost balíku a robot musí chytnout balík, pochopit jaký balík to je (je jich asi 5 druhu)a dat ho do správné krabice.

Zadaní bylo splněno ve spolupráci s kolegou Martinez Lema-David Sebastian, vyhledávačem google.com, tutoriálem z youtube kanálu sentdex a asi několika hrnky kávy.

Figure 1: Set-up na testovaní.

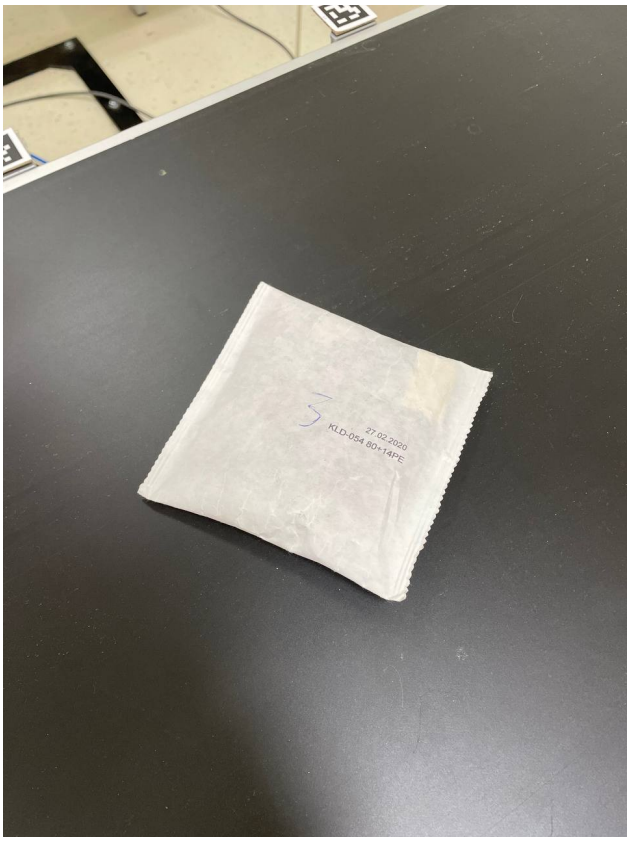


Figure 2: Příklad balíku

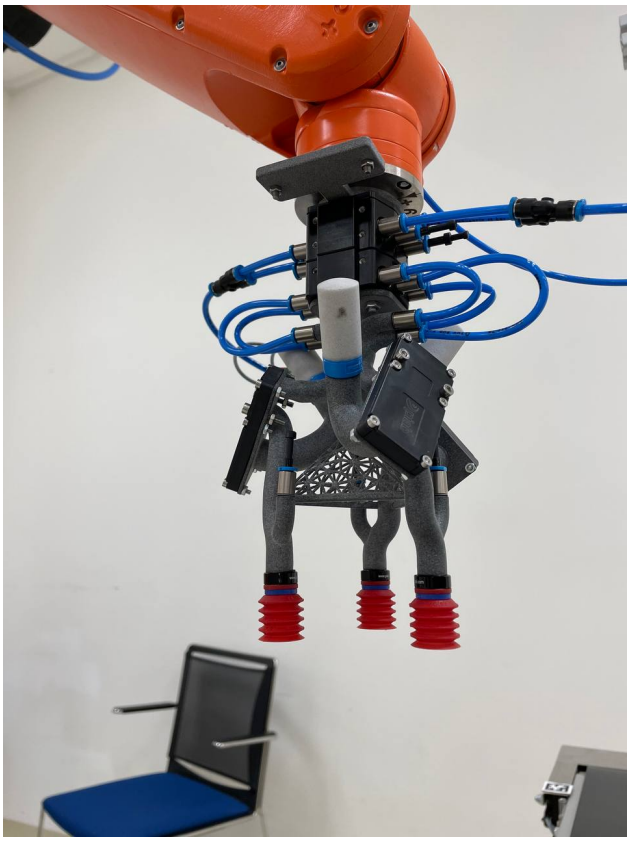


Figure 3: Gripper

Figure 4: Kamera nad dopravníkem

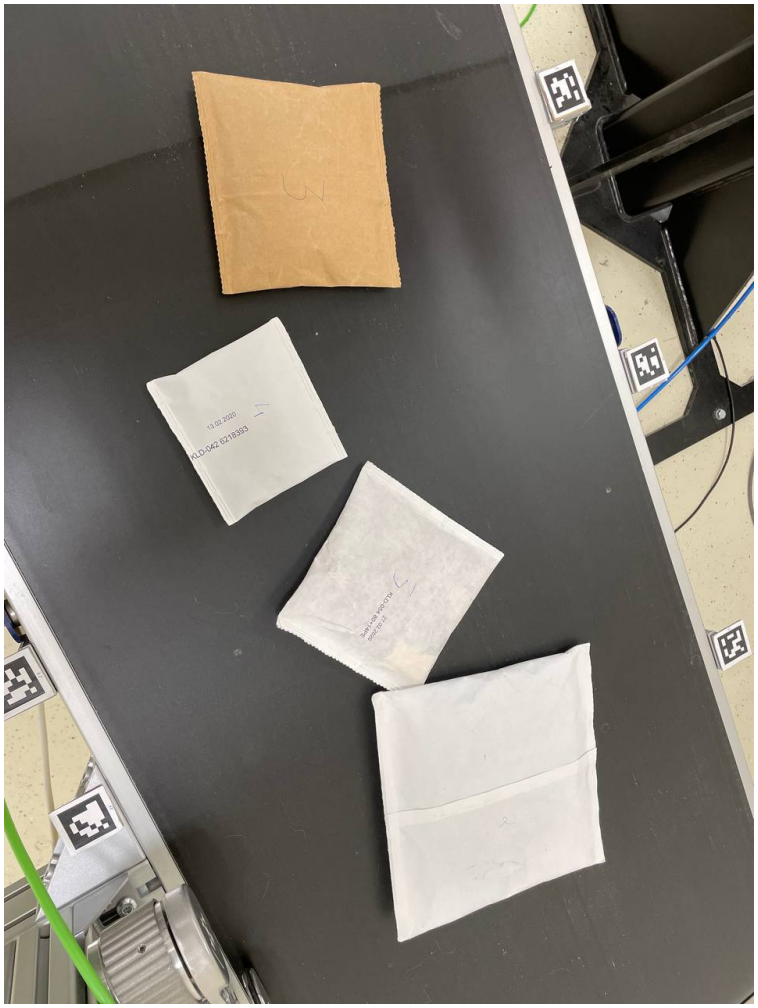


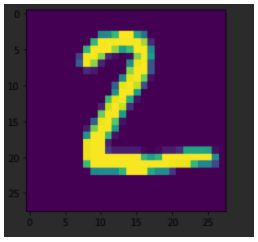
Figure 5: Balíky na dopravníků

2 Zadaní

Osobně já mám rad zadaní, které jsou spojené z reálnou praktikou. V procesu vývoje jsme setkali s potřebou odšuměni vstupního obrázku, což bych dal jako úlohu na extra body VIR

3 Příprava dat

Bohužel aktuálně nemáme přístup do set-upu robotu (je využít na jinem projektu), takže jako příklad použijeme populární datasets.mnist
Příklad vstupního obrázku:

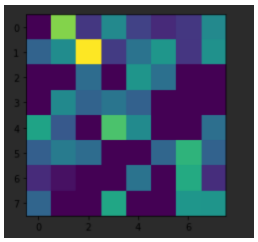


Udělám simulaci sumu:

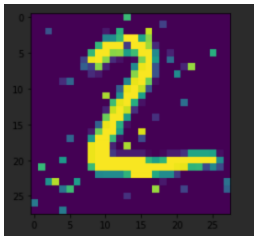
```
import random

def add_noise(img, random_chance=5):
    noisy = []
    for row in img:
        new_row = []
        for pix in row:
            if random.choice(range(100)) <= random_chance:
                new_val = random.uniform(0, 1)
                new_row.append(new_val)
            else:
                new_row.append(pix)
        noisy.append(new_row)
    return np.array(noisy)
noisy = add_noise(x_test[x_test_indx])
plt.imshow(noisy)
```

Šum:



Simulace zašuměného obrázku který dáme jako vystup z kamery:



4 Řešeni

Pro řešeni použiju standartní řešeni na odšuměni obrázku – autoencoders. V našem případě to vypadá takhle:

```
encoder_input = keras.Input(shape=(28,28,1),name="img")
x = keras.layers.Conv2D(28,(3,3), activation='relu', padding='same')(encoder_input)
x = keras.layers.MaxPooling2D((2,2), padding='same')(x)
x = keras.layers.Conv2D(12,(3,3), activation='relu', padding='same')(x)
x = keras.layers.MaxPooling2D((2,2), padding='same')(x)
x = keras.layers.Flatten()(x)
encoder_out = keras.layers.Dense(64,activation="relu")(x)
encoder = keras.Model(encoder_input,encoder_out,name="encoder")

decoder_input = keras.layers.Dense(588,activation="relu")(encoder_out)
x = keras.layers.Reshape((7,7,12))(decoder_input)
x = keras.layers.Conv2D(12,(3,3), activation='relu', padding='same')(x)
x = keras.layers.UpSampling2D((2,2))(x)
x = keras.layers.Conv2D(28,(3,3), activation='relu', padding='same')(x)
x= keras.layers.UpSampling2D((2,2))(x)
decoder_out = keras.layers.Conv2D(1,(3,3), activation='relu', padding='same')(x)
```

5 Dosazeny výsledek