Zadatak - dan 2

Prikažite na videu:

- Redni broj frame-a
- Broj objekata po klasi za svaki frame (barem 2 klase)
- Broj svih objekata

Priprema paketa

```
from IPython.display import clear_output
```

```
!pip3 install torch torchvision torchaudio --index-url https://download.pytorch.
clear_output(wait=False)
```

```
!pip3 install ultralytics
clear_output()
```

```
import torch
from ultralytics import YOLO
from PIL import Image
import cv2
import matplotlib.pyplot as plt
import colorsys
from IPython.display import HTML
from base64 import b64encode
import os
import shutil
```

```
if torch.cuda.is_available():
    # Get the default CUDA device
    device = torch.device('cuda')
    print(f'Device name: {torch.cuda.get_device_name(device)}')
    print(f'Total memory available: {torch.cuda.get_device_properties(device).tc
    print(f'CUDA version: {torch.version.cuda}')
else:
    print('CUDA is not available on this system.')
```

```
 Device name: Tesla T4
 Total memory available: 15.84 GB
 CUDA version: 11.8
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
 Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/cola
```

✓ 0s completed at 12:31 PM

```
    Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
    Requirement already satisfied: requests[socks] in /usr/local/lib/python3.1
    Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packa
    Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-pack
    Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10
    Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/
    Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/pyt
    Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python
    Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib
    Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/d
    Requirement already satisfied: PySocks!=1.5.7,>=1.5.6 in /usr/local/lib/py
```

```python
import gdown

# Download the video file
file_id = '1Fd9Yq2LqEclNjxfLr1RCxFV1lWAI4Af0'
url = f'https://drive.google.com/uc?id={file_id}'
output = '/content/test_vid.mp4'  # Provide the desired output path and filename
gdown.download(url, output, quiet=False)
```

```
    Downloading...
    From: https://drive.google.com/uc?id=1Fd9Yq2LqEclNjxfLr1RCxFV1lWAI4Af0
    To: /content/test_vid.mp4
    100%|██████████| 15.2M/15.2M [00:00<00:00, 127MB/s]
    '/content/test_vid.mp4'
```

```python
class Colors:
    # Ultralytics color palette https://ultralytics.com/
    def __init__(self):
        # hex = matplotlib.colors.TABLEAU_COLORS.values()
        hexs = ('FF3838', 'FF9D97', 'FF701F', 'FFB21D', 'CFD231', '48F90A', '92C
                '2C99A8', '00C2FF', '344593', '6473FF', '0018EC', '8438FF', '520
        self.palette = [self.hex2rgb(f'#{c}') for c in hexs]
        self.n = len(self.palette)

    def __call__(self, i, bgr=False):
        c = self.palette[int(i) % self.n]
        return (c[2], c[1], c[0]) if bgr else c

    @staticmethod
    def hex2rgb(h):  # rgb order (PIL)
        return tuple(int(h[1 + i:1 + i + 2], 16) for i in (0, 2, 4))

colors = Colors()
```

YouTube video download

Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
    Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/col
    Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab
    Requirement already satisfied: pytube in /usr/local/lib/python3.10/dist-pac
```

```
from pytube import YouTube
import os

def downloadYouTube(videourl, path, resolution="720p"):

    yt = YouTube(videourl)
    yt = yt.streams.filter(progressive=True, file_extension='mp4', resolution=re
    if not os.path.exists(path):
        os.makedirs(path)
    yt.download(path, filename="smallcat.mp4")

#TODO stavite odgovarajući YouTube link te želejni filename
downloadYouTube('https://www.youtube.com/watch?v=W86cTIoMv2U', '/content', '720p
```

```
# extract seconds worth of video frames of it
from IPython.display import clear_output
!yes Y | ffmpeg -ss 00:00:33 -i "/content/smallcat.mp4" -t 00:00:56 -c copy catc
clear_output()
```

```
!yes Y | ffmpeg -i "/content/catcropped.avi" cropped_test.mp4
clear_output()
```

Model

```
# load model
model = YOLO("yolov8n.pt")
# set model parameters
model.conf = 0.25  # NMS confidence threshold
model.iou = 0.45  # NMS IoU threshold
model.agnostic = False  # NMS class-agnostic
model.multi_label = False  # NMS multiple labels per box
model.max_det = 1000  # maximum number of detections per image

classes = model.names
list(classes.values())[:10]
```

```
    Downloading https://github.com/ultralytics/assets/releases/download/v0.0.0/
    100%|██████████| 6.23M/6.23M [00:00<00:00, 151MB/s]
    ['person',
     'bicycle',
     'car',
     'motorcycle',
     'airplane',
     'bus',
     'train',
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

Pomoćna dokumentacija

cv2.putText()

Video inference

```
red = (255, 0, 0)
green = (0, 255, 0)
blue = (0, 0, 255)
yellow = (255, 255, 0)
orange = (0, 165, 255)
purple = (178, 102, 155)
```

```
classes = model.names
list(classes.values()) [15:30]
```

```
    ['cat',
     'dog',
     'horse',
     'sheep',
     'cow',
     'elephant',
     'bear',
     'zebra',
     'giraffe',
     'backpack',
     'umbrella',
     'handbag',
     'tie',
     'suitcase',
     'frisbee']
```

```
def draw_boxes(image, results):
  # TODO: variables init
  countercat = 0
  countertree = 0

  for result in results[0].boxes.data.to("cpu"):
  # unpacking model output
    x1, y1, x2, y2 = int(result[0]), int(result[1]), int(result[2]), int(result|
    conf = result[4]
    class_id = int(result[5])
    class_name = classes[class_id]

    # TODO: count by class
    for class id in classes:
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
      elif (class name == 'tree'):
          countertree +=1

        #drawing
        cv2.rectangle(image, (x1,y1), (x2, y2), colors(class_id), 2)
        label = f'{class_name} {conf:.2f}'
        (w, h), _ = cv2.getTextSize(label, cv2.FONT_HERSHEY_SIMPLEX, 0.5, 2)
        cv2.rectangle(image, (x1, y1-h-15), (x1+w, y1), colors(class_id), -1)
        cv2.putText(image, label, (x1,y1-10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255,25

      # TODO: drawing counts

        cv2.putText(image, countercat, (x1, y1), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255

      return image


    # labels dir
    labels_dir = 'labels'
    if not os.path.exists(labels_dir):
      os.makedirs(labels_dir)
    else:
      shutil.rmtree(labels_dir)

    # Load the video
    # TODO: put valid path
    cap = cv2.VideoCapture('/content/cropped_test.mp4')

    # Get the video properties
    fps = cap.get(cv2.CAP_PROP_FPS)
    width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
    height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

    # Define the video writer
    fourcc = cv2.VideoWriter_fourcc(*'MJPG')
    out = cv2.VideoWriter('output.avi', fourcc, fps, (width, height))

    # Loop through the frames of the video
    frame_n = 0

    while cap.isOpened():
        ret, frame = cap.read()

        if ret:
          frame_n += 1
          # TODO: put frame number on frame
         #.se
          # Process the frame here
          result = model(frame, verbose=False)
          frame = draw_boxes(frame, result)
          out.write(frame)
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
        #cv2.imshow(frame)

        # Wait for the user to press a key (optional)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    else:
        break

# Release the video capture object and close all windows
cap.release()
cv2.destroyAllWindows()
```

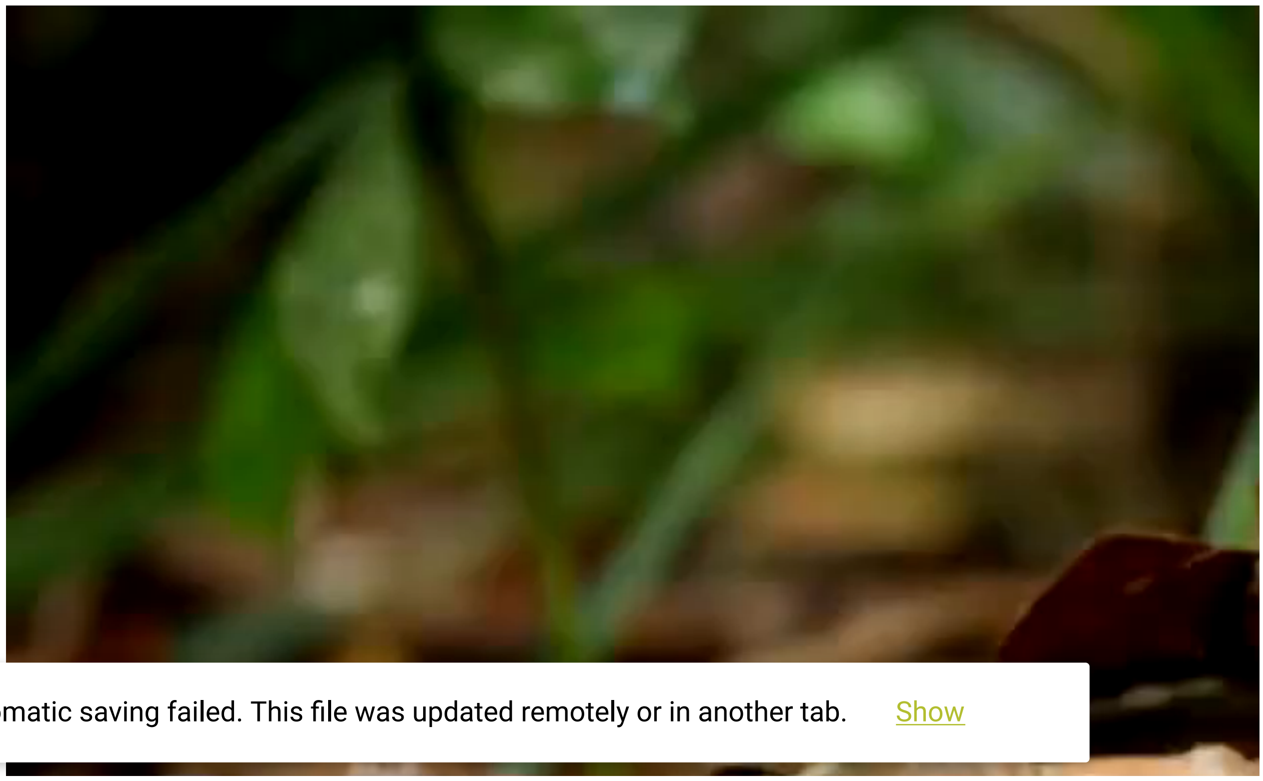
Double-click (or enter) to edit

Prikaz rezultata

```
# TODO: put valid path
!yes Y | ffmpeg -i "/content/output.avi" final.mp4
clear_output()
```

```
mp4 = open('/content/final.mp4','rb').read()
data_url = "data:video/mp4;base64," + b64encode(mp4).decode()
```

```
HTML("""
<video controls>
      <source src="%s" type="video/mp4">
</video>
""" % data_url)
```



Automatic saving failed. This file was updated remotely or in another tab. Show diff

Zadatak 2 - detekcija određenog područja

Alat za pregled koordinata na slici: pixspy

- Dohvatite prvi frame pomoću ćelije koda ispod te pomoću pixspy provjerite željene točke za pravokutnik (x1, y1), (x2, y2)
- Spremite točke u **globalnu** varijablu npr. rectangle = (x1, y1, x2, y2)
- Iscrtajte pravokutnik na svakom učitanom frame-u videa
- Odredite centar objekta i iscrtajte ga funckijom cv2.circle(frame, (x,y), radius, color, thickness)
- Pripremite funkciju koja će provjeravati nalazi li se centar objekta unutar pravokutnika
- Provjerite detektiraju li se objekti zaista samo u području pravokutnika

```
# TODO: put valid path
cap = cv2.VideoCapture("path_to_vid")
ret, frame = cap.read()
if ret:
    cv2.imwrite("fframe.jpg", frame)
cap.release()
cv2.destroyAllWindows()
```

```
rectangle = (0,0,0,0)
```

```
# TODO: Additionaly functions (optional)
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
def draw_boxes(image, results):
  # TODO:variables initialization


  for result in results[0].boxes.data.to("cpu"):
  # unpacking model output
    x1, y1, x2, y2 = int(result[0]), int(result[1]), int(result[2]), int(r
    conf = result[4]
    class_id = int(result[5])
    class_name = classes[class_id]

    # TODO: check if is in area


    #drawing
    cv2.rectangle(image, (x1,y1), (x2, y2), colors(class_id), 2)
    label = f'{class_name} {conf:.2f}'
    (w, h), _ = cv2.getTextSize(label, cv2.FONT_HERSHEY_SIMPLEX, 0.5, 2)
    cv2.rectangle(image, (x1, y1-h-15), (x1+w, y1), colors(class_id), -1)
    cv2.putText(image, label, (x1,y1-10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (

  # TODO: drawing counts


  return image


# Rectangle coordinates as a global variable
# rect = (600,395,1230,969)

# labels dir
labels_dir = 'labels'
if not os.path.exists(labels_dir):
  os.makedirs(labels_dir)
else:
  shutil.rmtree(labels_dir)

# Load the video
# TODO: put valid path
cap = cv2.VideoCapture('/content/test_vid.mp4')

# Get the video properties
fps = cap.get(cv2.CAP_PROP_FPS)
width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

# Define the video writer
fourcc = cv2.VideoWriter_fourcc(*'MJPG')
out = cv2.VideoWriter('output_area.avi', fourcc, fps, (width, height))

# Loop through the frames of the video
frame_n = 0
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
        if ret:
          frame_n += 1
          # TODO: draw rectangle on each frame


          # Process the frame here
          result = model(frame, verbose=False)
          frame = draw_boxes(frame, result)
          out.write(frame)

          # save results
          # result[0].save_txt(f'{labels_dir}/{frame_n}.txt')
          # Show the frame
          # cv2_imshow(frame)

          # Wait for the user to press a key (optional)
          if cv2.waitKey(1) & 0xFF == ord('q'):
              break
        else:
          break

    # Release the video capture object and close all windows
    cap.release()
    cv2.destroyAllWindows()


    # TODO: put valid path
    !yes Y | ffmpeg -i "/content/output_area.avi" final_area.mp4
    clear_output()


    # TODO: put valid path
    mp4 = open('/content/final_area.mp4','rb').read()
    data_url = "data:video/mp4;base64," + b64encode(mp4).decode()


    HTML("""
    <video controls>
          <source src="%s" type="video/mp4">
    </video>
    """ % data_url)
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff