

Министерство образования Республики Беларусь  
Учреждение образования  
«Брестский государственный технический университет»  
Кафедра ИИТ

Лабораторная работа №4  
По дисциплине: «ОИвИС»  
Тема: «Трекинг множественных объектов»

Выполнила:  
Студентка 4 курса  
Группы ИИ-23  
Тутина Е.Д.  
Проверила:  
Андренко К. В.

Брест 2025

**Цель работы:** исследовать применение алгоритмов трекинга на базе обученной сети-детектора объектов.

### Вариант 11

11	YOLOv11m	<b>Счетчики расхода воды:</b> <a href="https://universe.roboflow.com/koe-r3741-gmail-com/watermeteramrv2/dataset/1">https://universe.roboflow.com/koe-r3741-gmail-com/watermeteramrv2/dataset/1</a>
----	----------	--

### Общее задание

1. Используя сеть-детектор, обученный в ЛР 3, реализовать логику для отслеживания множественных объектов, используя библиотеку Ultralytics YOLO;
2. Применять алгоритмы BoT-Sort и ByteTrack (задействовать соответствующие конфигурационные файлы);
3. Исследовать изменения параметров в конфигурационных файлах и их влияние на качество трекинга;
4. В качестве исходных видеоматериалов для экспериментов использовать видео-ролики из сети (например, из YouTube), содержащие множественные объекты классов из ЛР 3;
5. Оформить отчет по выполненной работе, залить исходный код и отчет в соответствующий репозиторий на github.

### Код работы:

```
!pip install -U pip
!pip install ultralytics roboflow supervision
from google.colab import drive
drive.mount('/content/drive')

!mkdir -p /content/drive/MyDrive/watermeter_project
!unzip -q /content/WaterMeterAMRV2.v1i.yolov11.zip -d /content/watermeter_dataset
!ls /content/watermeter_dataset
import os

def find_data_yaml(root="/content"):
    for dirpath, _, filenames in os.walk(root):
```

```

        if "data.yaml" in filenames:
            return os.path.join(dirpath, "data.yaml")
    return None

data_yaml = find_data_yaml("/content")
print("Found:", data_yaml)

from ultralytics import YOLO

model = YOLO("yolol11m.pt")

results = model.train(
    data=data_yaml,
    epochs=50,
    imgsz=640,
    batch=16,
    name="watermeter_detector",
    project="/content/drive/MyDrive/watermeter_project"
)

from ultralytics import YOLO

weights = "/content/drive/MyDrive/watermeter_project/watermeter_detector/weights/best.pt"
model = YOLO(weights)

from PIL import Image
import numpy as np
import cv2
from IPython.display import display

img_path = "/content/test_photo.png"
img = cv2.imread(img_path)
res = model(img)
annot = res[0].plot()

display(Image.fromarray(annot))

import shutil, ultralytics, os, glob

ultra_path = os.path.dirname(ultralytics.__file__)
tracker_dir = ultra_path + "/trackers"

print("Trackers dir:", tracker_dir)
print(glob.glob(tracker_dir + "/*.yaml"))

shutil.copy(tracker_dir + "/bytetrack.yaml", "/content/bytetrack_custom.yaml")
shutil.copy(tracker_dir + "/botsort.yaml", "/content/botsort_custom.yaml")

res_bt = model.track(
    source="/content/input_video.mp4",
    tracker="/content/bytetrack_custom.yaml",
    conf=0.25,

```

```


    imgsz=640,
    persist=True,
    save=True,
    save_txt=True
)

res_bot = model.track(
    source="/content/input_video.mp4",
    tracker="/content/botsort_custom.yaml",
    conf=0.25,
    imgsz=640,
    persist=True,
    save=True,
    save_txt=True
)

```

### Результаты обучения модели:


Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size
1/50	7.85G	1.392	1.701	1.174	52	640:

100%  218/218 1.6it/s 2:15

Class	Images	Instances	Box (P)	R	mAP50
all	389	2059	0.741	0.671	0.738

mAP50-95) 0.382

Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size
50/50	8.06G	0.9445	0.3728	0.986	35	640:

100%  218/218 1.7it/s 2:04

Class	Images	Instances	Box (P)	R	mAP50
all	389	2059	0.953	0.964	0.978

mAP50-95) 0.637

### Результаты запуска ByteTrack:

Tracking: ByteTrack tracker=/content/bytetrack\_custom.yaml

```

video 1/1 /videoplayback (6).mp4: 640x384 17 objects
video 1/1 /videoplayback (6).mp4: 640x384 18 objects
video 1/1 /videoplayback (6).mp4: 640x384 16 objects
...

```

Speed: 1.5ms preprocess, 12.3ms inference, 1.9ms postprocess per frame

Results saved to /content/runs/track/exp

### Результаты запуска BoT-SORT:

Tracking: BoT-SORT tracker=/content/botsort\_custom.yaml

ReID model loaded: osnet\_x0\_25\_msmt17.pt

video 1/1 /videoplayback (6).mp4: 640x384 17 objects

video 1/1 /videoplayback (6).mp4: 640x384 18 objects

Speed: 1.6ms preprocess, 13.1ms inference, 3.3ms postprocess per frame

Results saved to /content/runs/track/exp2

**Выводы:** исследовала применение алгоритмов трекинга на базе обученной сети-детектора объектов.

Вывод: осуществила обучение нейросетевого детектора для решения задачи обнаружения заданных объектов .