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

Лабораторная работа №4

По дисциплине: «ОИвИС»

Тема: «Трекинг множественных объектов»

Выполнил:
Студент 4 курса
Группы ИИ-23
Ежевский Е.Р.
Проверила:
Андренко К.В.

Брест 2025

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

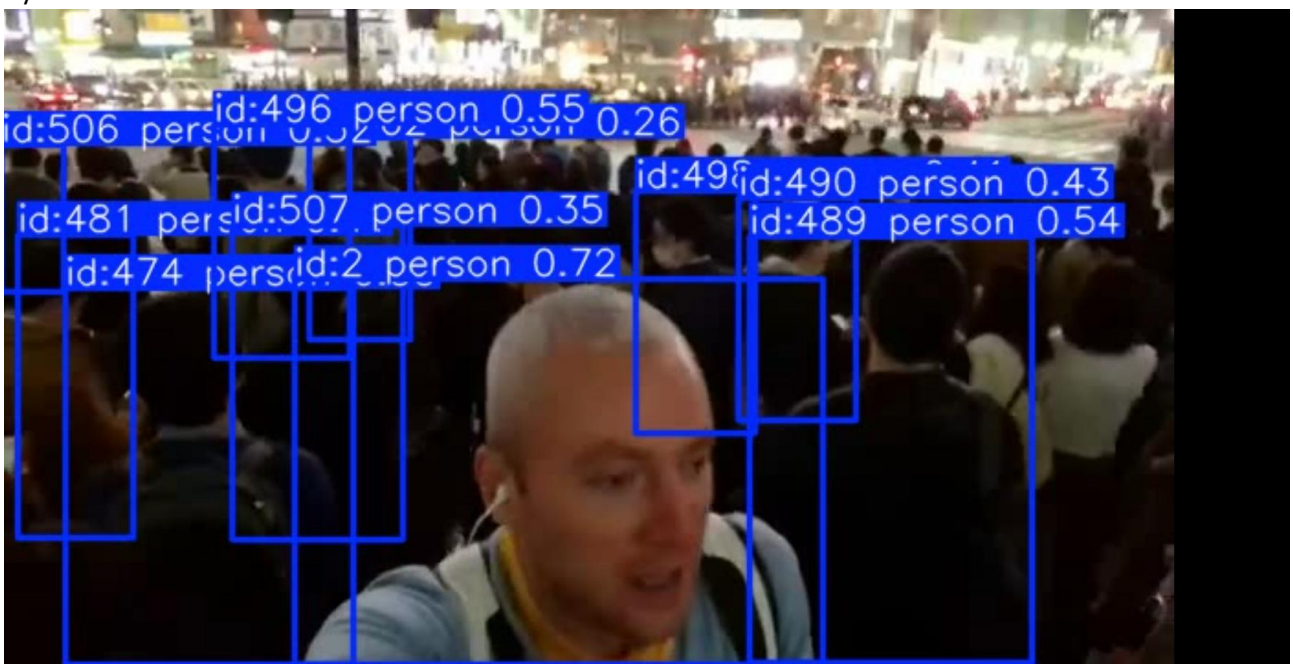
Общее задание

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.Вариант:

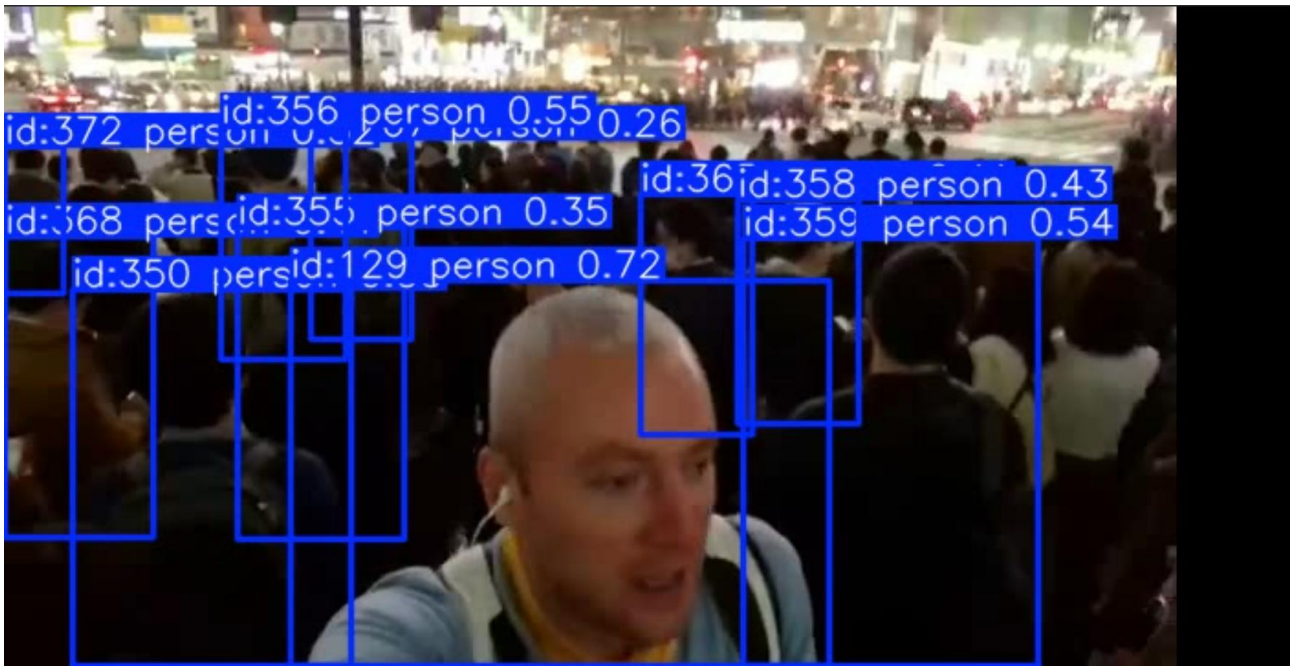
Код программы:

```
from ultralytics import YOLO

model = YOLO("runs/detect/train2/weights/best.pt")

model.track(
   source="video.mp4",
   conf=0.25,
   iou=0.45,
   tracker="botsort.yaml",
   show=False,
   save=True,
   name="botsort"
)

model.track(
   source="video.mp4",
   conf=0.25,
   iou=0.45,
   tracker="bytetrack.yaml",
   show=False,
   save=True,
   name="bytetrack"
)
```

Bytesort:



Botsort:



При проведении экспериментов с трекерами ByteTrack и BoT-SORT была проанализирована роль различных настраиваемых параметров в обеспечении стабильности и точности сопровождения объектов на основе модели-детектора, разработанной ранее. Влияние каждого параметра наблюдалось отдельно, что позволило выявить специфические особенности поведения каждого алгоритма.

Для трекера ByteTrack основным фактором, определяющим баланс между полнотой обнаружения и числом ложных срабатываний, выступил параметр track_thresh – порог уверенности детекции. Снижение порога способствовало удержанию удалённых или частично перекрытых объектов, однако одновременно увеличивало количество ошибочных треков, формирующихся на основе шумовых сигналов. Повышение порога, напротив, снижало уровень ложных треков, но чаще приводило к потере объектов. Параметр match_thresh, задающий порог пересечения по IoU для сопоставления детекций с существующими треками, также существенно влиял на стабильность идентификаторов: низкие значения увеличивали частоту смены ID, тогда как высокие – повышали устойчивость, но ухудшали реакцию на быстрое движение. Увеличение размера track_buffer помогало сохранять трек при кратковременных

пропаданиях детекции, однако в некоторых случаях это приводило к продолжению «фантомных» треков после исчезновения объекта.

В отличие от ByteTrack, алгоритм BoT-SORT продемонстрировал высокую чувствительность к качеству входных детекций. Изменение порога det_thresh вызывало значительные колебания в работе трекера: при заниженных значениях резко возрастало количество ложных траекторий, а при завышенных – участились случаи потери объектов даже при кратковременном снижении уверенности детектора. Ключевую роль играл параметр conf_thres, регулирующий передачу детекций в модуль ReID. Наилучшие результаты достигались в диапазоне 0.4–0.5: снижение порога ухудшало качество извлекаемых визуальных признаков, а повышение ограничивало объём данных, доступных для повторной идентификации.

Активация ReID-модуля через параметр with_reid существенно повысила устойчивость сопровождения: BoT-SORT практически перестал путать объекты при пересечении траекторий. Без его использования алгоритм приближался по поведению к классическому SORT с частыми сменами идентификаторов. Параметр gamma, влияющий на настройки фильтра Калмана, определял степень сглаживания путей объектов: увеличение значения делало траектории более плавными, но снижало способность реагировать на резкие изменения направления, что иногда приводило к заметным смещениям треков.

Сравнение двух методов показало, что ByteTrack отличается большей устойчивостью к изменениям параметров и сохраняет работоспособность даже при значительном снижении пороговых значений. BoT-SORT, напротив, требует более точной настройки и сильно зависит от качества работы детектора. В целом ByteTrack проявил себя как более надёжное решение в динамичных и сложных сценах, тогда как BoT-SORT оказался предпочтительнее для задач, требующих точного и устойчивого сопровождения отдельных объектов, особенно при использовании механизма повторной идентификации.

Вывод: исследовал применение алгоритмов трекинга на базе обученной сети-детектора объектов