Multi-object tracking на основе библиотеки Norfair

Сичкар Георгий | СПбГУ | МатМех | Программная инженерия 3 курс

Введение

Доклад посвящен задаче multi-object-tracking, а также битблиотеке с открытым исходным кодом **norfair**. Это поверхностное погружение в предметную область трекинга объектов на видео. План доклада примерно такой:

- базовые определения и терминология;
- гайд по использованию **norfair**;
- примеры использования.

Трекинг

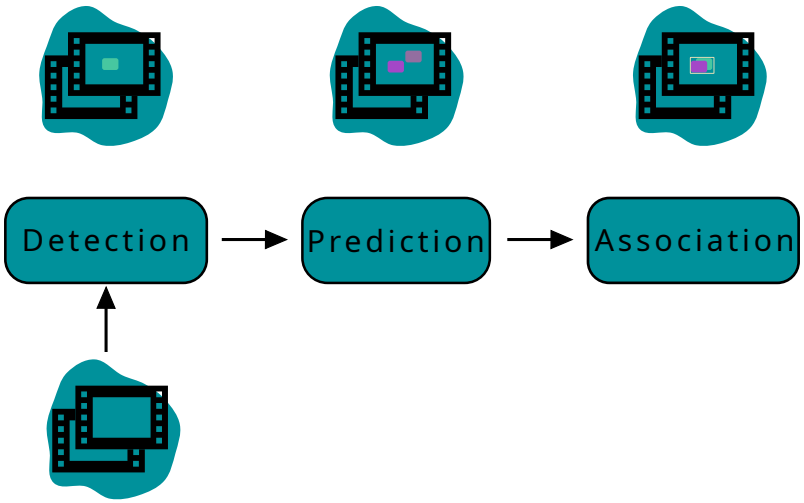
Object tracking —это задача отслеживания объекта на протяжении всего видео.

- Single object tracking (SOT) —обнаружение объекта не требуется (видеонаблюдение, видеотрансляции)
- Multi-object tracking (MOT) —одним из этапов является детектирование объектов (робототехника, автономное движение)

Существует довольно большое множество областей, в которых используется MOT:

- спортивный анализ —отслеживания мяча в теннисе, волейболе, футболе (Hawk-Eye Innovations);
- видеонаблюдение —отслеживание необычной активности;(HIKVISION);
- робототехника —сложные взаимодействия (с человеком).

Этапы трекинга



Зачастую алгоритмы состоят из трех частей. Первый этап —это детекция объекта на кадре. Второй этап заключается в предсказании перемещения объекта на основе уже имеющихся данных. Ну если более понятным языком, то мы в системе храним какой-то пул объектов, которые распознали ранее, и по данным о их перемещениях в прошлом пытаемся угадать куда они попадут в будущем. После создания гипотезы "где могут находится объекты"на третьем шаге происходит сопоставление объектов которые мы распознали с предсказанными перемещениями.

Алгоритмы и их реализации с открытым исходным кодом

Существует множество алгоритмов для решения задачи MOT. Здесь приведен список нескольких довольно популярных вариантов:

- SORT —использует Kalman-фильтр и Венгерский алгоритм
- DeepSORT —расширение SORT (дополнительно использует сверточную нейронную сеть, для определения идентичности объекта)
- ByteTrack
- FairMOT
- ...

Есть множество библиотек с открытым исходным кодом, которые реализуют MOT, опираясь на вышеописанные алгоритмы. Ниже перечислены самые популярные решения:

- OpenCV (популярная библиотека для computer vision);
- AlphPose (используется для трэкинга людей);
- dblib (только SOT);
- Ultralytics YOLO (алгоритм трэкинга —ByteTrack, YOLO для детекции);
- Norfair;
- ...

Norfair

Norfair — это настраиваемая облегченная библиотека Python для отслеживания нескольких объектов в реальном времени.

Преимущества:

- Пользовательский детектор.
- Алгоритм трэкинга —SORT.
- Обработка видео —OpenCV.
- Лицензия —BSD 3-Clause.

Базовая система трекинга с norfair

```
from norfair import Tracker, Video

video = Video(input_path="my_video.mp4")
tracker = Tracker()
detector = MyDetector()

for frame in video:
    detections = detector(frame)
    tracked_objects = tracker.update(detections=detections)
    draw_tracked_objects(frame, tracked_objects)
    video.write(frame)
```

- `Video()` —используется для представления пользовательского видео в программе (в частности для раскадровки).
- `Tracker()` —основной класс для трекинга объектов на видео, инкапсулирующий в себе этапы "предсказания"и "ассоциации".
- `MyDetector()` —класс, отвечающий за распознавание объектов на кадре. (Этот класс надо реализовать самому, он не предоставляется библиотекой)

Реализация MyDetector

Чтобы адаптировать сторонний детектор (какой-нибудь YOLO...), надо реализовать интерфейс `IDetector`.

```
from norfair import Detection

class IDetector(Protocol):
    @abstractmethod
    def __call__(self, frame: np.ndarray) -> Iterable[Detection]:
        pass

class MyDetector(IDetector):
    def __call__(self, frame: np.ndarray) -> Iterable[Detection]:
        object_points: numpy.ndarray
        ...
        return [Detection(point) for point in object_points]
```

Основные классы Norfair

- `Detection` —класс-контейнер, описывающий одну детекцию. Детекция отличается от объекта трекинга тем, что существует только на одном кадре видео. Алгоритм трекинга соотносит детекцию с предсказанным положением объекта и если они совпали, то присваивает детекцию объекту.
- `TrackedObject` —объекты трэкинга. Этот класс имеет "приватный конструктор"(экземпляры создаются и модифицируются классом `Tracker`)
- `Tracker` —основной класс, реализующий работу трекинга объектов.

Detection

```
from norfair import Detection

detection = Detection(
    points: np.ndarray,
    scores: np.ndarray
)
```

- `points` —набор точек, описывающих объект трекинга
- `scores` —уверенность детектора в том, что он правильно распознал объект

TrackedObject

```
from norfair import TrackedObject

tracked_objects: list[TrackedObject]
tracked_objects = tracker.update(...)
```

- `estimate` —где будут точки объекта в текущем кадре
- `id` —имя объекта
- `age` —сколько фрэймов пережил
- `last_detection` —последнее связывание с Detection

Tracker

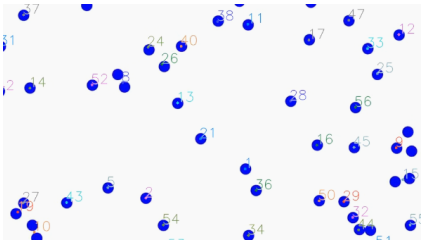
```
from norfair import Tracker

tracker = Tracker(
    distance_function: str | Callable,
    distance_threshold: float,
    hit_counter_max: int,
    detection_threshold: float,
    filter_factory: FilterFactory
)
```

- `distance_function` —используемая трекером для определения расстояния между `TrackedObject` и `Detection`;
- `distance_threshold` —максимальное расстояние, которое может считаться совпадением между `TrackedObject` и `Detection`;
- `hit_counter_max` —сколько подряд случившихся совпадений могут влиять на срок жизни `TrackedObject`;
- `detection_threshold` —пороговое значение (если `score` у точки ниже, то точка отбрасывается);

- `filter_factory` —может быть использована для изменения параметров фильтра (KalmanFilter);

Примеры

Полное видео с машинками.



Полное видео с точками.