computervision-final-prep / lab / Lab 09 (Tracking)-20251128 / **tracker_utils.py**

limhpone lab 9-obj tracking 267755b · 2 hours ago

75 lines (67 loc) · 2.68 KB

Code Blame Raw

```python
import cv2
import numpy as np

class MultiObjectTracker:
    def __init__(self, tracker_type="CSRT", use_legacy=True):
        self.tracker_type = tracker_type.upper()
        self.use_legacy = use_legacy and hasattr(cv2, "legacy")
        self.trackers = None
        self.factory = self._get_factory(self.tracker_type)

    def _get_factory(self, tracker_type):
        if self.use_legacy:
            FACTORIES = {
                "KCF":   cv2.legacy.TrackerKCF_create,
                "MIL":   cv2.legacy.TrackerMIL_create,
            }
        else:
            FACTORIES = {
                "KCF":   getattr(cv2, "TrackerKCF_create", None),
                "MIL":   getattr(cv2, "TrackerMIL_create", None),
            }
        factory = FACTORIES.get(tracker_type)
        if factory is None:
            raise ValueError(f"Tracker type '{tracker_type}' is not available in this OpenCV
        return factory

    def _create_multitracker(self):
        if self.use_legacy:
            return cv2.legacy.MultiTracker_create()
        return cv2.MultiTracker_create()

    def reset(self):
        self.trackers = None
```

```
34
35      def init_from_boxes(self, frame_or_roi, boxes):
36          self.trackers = self._create_multitracker()
37          for box in boxes:
38              tr = self.factory()
39              self.trackers.add(tr, frame_or_roi, tuple(map(float, box)))
40
41      def add(self, frame_or_roi, box):
42          if self.trackers is None:
43              self.trackers = self._create_multitracker()
44          tr = self.factory()
45          self.trackers.add(tr, frame_or_roi, tuple(map(float, box)))
46
47      def update(self, frame_or_roi):
48          if self.trackers is None:
49              return False, []
50          ok, boxes = self.trackers.update(frame_or_roi)
51          return ok, boxes
52
53      @staticmethod
54      def nms(boxes, iou_thr=0.3):
55          if not boxes:
56              return []
57          b = np.array(boxes, dtype=np.float32)
58          x1, y1 = b[:, 0], b[:, 1]
59          x2, y2 = b[:, 0] + b[:, 2], b[:, 1] + b[:, 3]
60          areas = (x2 - x1) * (y2 - y1)
61          order = areas.argsort()[::-1]
62          keep = []
63          while order.size > 0:
64              i = order[0]
65              keep.append(i)
66              xx1 = np.maximum(x1[i], x1[order[1:]])
67              yy1 = np.maximum(y1[i], y1[order[1:]])
68              xx2 = np.minimum(x2[i], x2[order[1:]])
69              yy2 = np.minimum(y2[i], y2[order[1:]])
70              iw = np.maximum(0.0, xx2 - xx1)
71              ih = np.maximum(0.0, yy2 - yy1)
72              inter = iw * ih
73              iou = inter / (areas[i] + areas[order[1:]] - inter + 1e-6)
74              order = order[1:][iou < iou_thr]
75          return [tuple(map(int, b[i])) for i in keep]
```