face_detection_trace Documentation

发布 *v1.0*

yuanjh

2020 年 09 月 27 日

Contents:

级联分类器配置文件:model/haarcascade_frontalface_default.xml, 来自开源项目 opencv

测试视频:video/1,2,6.mp4 来自开源项目 Human-detection-and-Tracking

CHAPTER 1

src package

1.1 Submodules

1.2 src.face_detection module

class src.face_detection.**FaceDetection**

基类：object

人脸检测 FaceDetection 抽象类

detection(*frame*) → List[Tuple[str, str, str, str]]

图片中所有的人脸矩形框坐标

参数 **frame** – 图片,h,w,s 三维数组格式

返回 List[tuple], 每个 tuple 都是 (int,int,int,int) 形式的 opencv 的矩形坐标

class src.face_detection.**FaceDetectionCvCas**

基类：*src.face_detection.FaceDetection*

cascade_xml = '/home/docs/checkouts/readthedocs.org/user_builds/face-detection-trace/checkouts/late

detection(*frame*)

图片中所有的人脸矩形框坐标

参数 **frame** – 图片,h,w,s 三维数组格式

返回 List[tuple], 每个 tuple 都是 (int,int,int,int) 形式的 opencv 的矩形坐标

face_detector = <CascadeClassifier 0x7f7d14136b10>

class src.face_detection.FaceDetectionDlibFro

基类：*src.face_detection.FaceDetection*

detection(*frame*)

图片中所有的人脸矩形框坐标

参数 **frame** – 图片,h,w,s 三维数组格式

返回 List[tuple], 每个 tuple 都是 (int,int,int,int) 形式的 opencv 的矩形坐标

static dlib_box_to_cv(*rectangle: Tuple[int, int, int, int]*) → Tuple[int, int, int, int]

坐标系转换

将 dlib 的 rectangle 坐标转为 opencv 坐标系坐标

参数 **rectangle** – dlib 的 rectangle 坐标

:return:opencv 中的坐标系

face_detector = <_dlib_pybind11.fhog_object_detector object>

class src.face_detection.FaceDetectionFactory

基类：object

face detection 的工厂类

face_detection_construct = {'CV_CAS': <class 'src.face_detection.FaceDetectionCvCas'>, 'DLIB_FRO':

static get_detection(*detection_method*)

获取 detection_method 对应的 detection 实例

参数 **detection_method** – 人脸检测方法

返回 检测方法实例

class src.face_detection.FaceDetectionFrFoc

基类：*src.face_detection.FaceDetection*

detection(*frame*) → List[Tuple[str, str, str, str]]

图片中所有的人脸矩形框坐标

参数 **frame** – 图片,h,w,s 三维数组格式

返回 List[tuple], 每个 tuple 都是 (int,int,int,int) 形式的 opencv 的矩形坐标

static fl_to_cv_box(*rect: Tuple[int, int, int, int]*) → Tuple[int, int, int, int]

坐标系转换

将 face_recognition 的 face_locations 坐标转为 opencv 坐标系坐标

参数 **rect** – face_locations 中的坐标

:return:opencv 中的坐标系

1.3 src.face_detection_trace module

class src.face_detection_trace.**CapDetectionTrack**(*ipc_info*, *is_realtime*, *face_detector*, *face_encoding*, *detection_freq*, *persons*, *face_decector_lock*, *face_encoding_lock*)

基类：threading.Thread

检测追踪类

集成人脸检测和追踪

变量

- **is_start** (*bool*) – 是否已经开启
- *face_detector* (*object*) – 人脸检测器
- *face_encoding* (*object*) – 人脸编码器
- **__last_frame** (*object*) – 视频流的最新帧
- **frame_queue** (*queue*) – 视频流的视频帧队列
- **is_realtime** (*bool*) – 是否是实时模式
- **tracks** (*list*) – 追踪器列表, 一个视频会出现多个人, 自然也有多个追踪器
- **ipc_info** (*list*) – 视频源配置信息
- **persons** (*list*) – 已保存的人员和对应的人脸信息

static event_call_back(*type*, *ipc_name*, *track_id*, *img=None*, *box=None*, *person_name=None*)

追踪的人消失后的回调函数

参数

- **type** – 事件类型
- **ipc_name** – 摄像头名称
- **track_id** – 追踪器 id
- **img** – 图片帧
- **box** – 图片帧中的人脸位置
- **person_name** – 人名

is_save_stranger

name

A string used for identification purposes only.

It has no semantics. Multiple threads may be given the same name. The initial name is set by the constructor.

path

run()

启动视频解码, 人脸检测和人脸特征码提取线程

save_release_resouce()

资源保存和释放, 保存追踪器里关联的陌生人的人脸图片

video_imgs = None

class src.face_detection_trace.**DetectionTracksCtl**(*face_detector*, *face_encoding*)

基类: object

人脸检测, 识别控制器

变量

- *face_detector* (*object*) – 人脸识别器
- *face_encoding* (*object*) – 人脸编码器

static **background_subtraction**(*previous_frame*, *frame_resized_grayscale*, *min_area*)

通过视频帧变化比率, 判断是否需要启动检测线程

如果视频帧不发生变化, 说明没有人出现, 不需要进行人脸检测和追踪识别

参数

- **previous_frame** – 上一帧图片信息
- **frame_resized_grayscale** –
- **min_area** – 最小变化区域阈值

返回 图片变化率

start_all(*ipc_infos*, *camera_persons*)

启动所有视频流的人脸检测线程

参数

- **ipc_infos** – 视频流配置信息
- **camera_persons** – 各视频流对应对应人员信息

class src.face_detection_trace.**Person**(*person_name*, *img_files*, *ipc_name*, *is_new=False*, *new_face_frame_max=10*)

基类: object

人员, 可能是员工或者陌生人

变量

- **ipc_name** (*str*) – 摄像头名称
- **is_new** (*bool*) – 是否是新人

- **person_name** (*str*) – 人员姓名
- **frames_box_limit** (*list*) – 如果是新人, 建立 list, 保存新人头像
- **__encodings** (*List*) – frames_box_limit 里头像对应的头像特征码信息

encodings_valid()

返回有效的 encoding 信息

返回 encoding 中有效的头像编码

face_encoding = None

static get_camera_person_files(*cameras_dir*)

加载某个摄像头下某人的所有头像图片信息

参数 **cameras_dir** – 图片目录

返回 dict,key:camera_name, 摄像头名称,value:dict02,dict02:key:person_name, 姓名,value:img_list, 此人对应头像列表

static get_unknow_name() → str

生成陌生人名字

返回 陌生人名字

img_dir = ''

new_frame_box(*frame_box*)

向 frames_box_limit 中新增 frame_box 信息

参数 **frame_box** – FrameFox 信息, 包含图片和图片里头像位置信息

static new_unknow_person(*ipc_name*)

新增陌生人

:param ipc_name: 摄像头名称:return: 陌生人 person 实例

save()

保存 frames_box_limit 中的图片和头像位置信息

class src.face_detection_trace.Track(*ipc_name*, *tracker*, *img*, *box*, *encoding*, *persons*, *event_call_back*, *history=5*)

基类: `object`

跟踪器

内部包含了 opencv 追踪器, 或者说对 opencv 追踪器的二次封装

变量

- **ipc_name** (*str*) – 摄像头名称
- **__id** (*int*) – 跟踪器 id
- **tracker** (*object*) – 跟踪器,opencv 追踪器实例

- **face_img** (*list*) – 图片里的头像小图信息
- **img** (*list*) – 图片
- *encoding* (*list*) – 人脸头像对应特征码
- **__history** (*list*) – 临近的各帧是否包含此人
- **match_person** (*object*) – 跟踪器匹配的人
- *event_call_back* (*callable*) – 追踪的人消失之后的回调函数, 比如生成事件日志

alive()

追踪器是否处于活跃活跃状态

追踪器匹配的人最近几帧是否出现过

返回

find_person(*persons*, *tolerance=0.6*)

从入参的 persons 中匹配 track 追踪器追踪的人 (将 track 和 person 关联起来)

:param persons: 所有人员列表:param tolerance: 人员匹配阈值, 如果人员距离小于此阈值则认为是同一个人:return:

id

update(*img*)

更新追踪器图片

参数 **img** – 图片

返回

update_img(*img*, *box*, *encoding*)

更新追踪器信息

:param img: 图片:param box: 人脸位置:param encoding: 人脸特征编码

1.4 src.face_encoding module

class src.face_encoding.**FaceEncoding**

基类: object

人脸特征值提取 face encoding 抽象类

static encoding(*img*, *box*)

获取图片中头像的特征码

参数

- **img** – 图片,h,w,s 三维信息
- **box** – 头像坐标

static encoding_frame_box(*frame_box*) → List[Tuple[str, str, str, str]]

获取 frame_box 中头像和对应 box 位置的特征码

参数 **frame_box** – frame_box, 包含了图片和 box 信息

static encoding_img(*face_img*)

获取图片中第一个头像的特征码 (言外之意, 图片本来就是头像图片)

参数 **face_img** – 图片,h,w,s 三维信息

class src.face_encoding.FaceEncodingDlibReg

基类: *src.face_encoding.FaceEncoding*

static cv_box_to_dlib(*box*)

static encoding(*img*, *box*)

获取图片中头像的特征码

参数

- **img** – 图片,h,w,s 三维信息
- **box** – 头像坐标

static encoding_frame_box(*frame_box*)

获取 frame_box 中头像和对应 box 位置的特征码

参数 **frame_box** – frame_box, 包含了图片和 box 信息

static encoding_img(*face_img*)

获取图片中第一个头像的特征码 (言外之意, 图片本来就是头像图片)

参数 **face_img** – 图片,h,w,s 三维信息

face_detector = <_dlib_pybind11.fhog_object_detector object>

face_encoding = <_dlib_pybind11.face_recognition_model_v1 object>

shape = <_dlib_pybind11.shape_predictor object>

class src.face_encoding.FaceEncodingFactory

基类: object

face encoding 的工厂类

face_encoding_construct = {'DLIB_REG': <class 'src.face_encoding.FaceEncodingDlibReg'>, 'FR_FE': <c

static get_instance(*encoding_method*)

获取 encoding_method 对应的 encoding 实例

参数 **encoding_method** – 人脸特征码提取方法

返回 特征码提取方法实例

class src.face_encoding.FaceEncodingFrFe

基类: *src.face_encoding.FaceEncoding*

static encoding(*img*, *box*)

获取图片中头像的特征码

参数

- **img** – 图片,h,w,s 三维信息
- **box** – 头像坐标

static encoding_frame_box(*frame_box*)

获取 frame_box 中头像和对应 box 位置的特征码

参数 **frame_box** – frame_box, 包含了图片和 box 信息

static encoding_img(*face_img*)

获取图片中第一个头像的特征码 (言外之意, 图片本来就是头像图片)

参数 **face_img** – 图片,h,w,s 三维信息

1.5 src.util module

class src.util.FrameBox(*img=None*, *box=None*)

基类: object

自定义对象, 包含图片帧和头像位置信息

变量

- **img** (*list*) – 图片,hws 三维数组格式
- **box** (*tuple*) – 头像位置, 长度为 4 的 list

name

使用 box 的头像坐标生成图片文件名称

返回 图片文件名称

static parse_file(*file_name*)

从文件名和文件内容恢复出 frame_box 信息

参数 **file_name** – 文件名

返回 tuple 元祖,tuple[0] 图片信息,hws 三维信息,tuple[1] 头像位置, 长度为 4 的 list

class src.util.LimitList(*maxsize=10*)

基类: object

长度受限制的 list

变量 **maxsize** (*int*) – list 最大长度

append(*item*)

向 list 新增元素

参数 `item` – 新增元素

返回 是否添加成功, 长度超过最大限度则返回 false

`pop()`

从 list 里弹出元素

返回 返回弹出的元素, 如果 list 为空则返回 None

`class src.util.Util`

基类: `object`

工具类

主要包括坐标系转换, 画框, 图片特定区域切割, 以及文件夹遍历等工具类

坐标系转换: 由于调用了不同的工具包, 不同包的坐标标识方法是不同的, 为了保持内部变量含义统一性, 程序内部均采用 opencv 坐标系

`static cut_frame_box(frame, box: Tuple[int, int, int, int])`

从图片中截取出矩形区域

参数

- `frame` – 图片,h,w,s 格式的 3 维数组
- `box` – 矩形框

返回 矩形框内的图片,h,w,s 格式的 3 维数组

`static cv_to_fl_box(rect: Tuple[int, int, int, int])` → Tuple[int, int, int, int]

坐标系转换

将 opencv 坐标转为 face_recognition 的 face_locations 坐标

参数 `rect` – opencv 坐标

:return:face_recognition.face_locations 中的坐标

`static draw_boxes(frame, box: Tuple[int, int, int, int])`

矩形框绘制

`static get_dirs_files(dir: str)` → Dict[str, List[str]]

获取 dir 下的所有文件列表

参数 `dir` – 目录路径

:return:dict,key:person name,value:list,person face img

`static get_file_path_split(filename: str)` → Tuple[str, str, str]

将文件全路径拆分为文件目录路径, 文件名, 文件扩展名

参数 `filename` – 文件全路径

返回 tuple, 文件目录路径, 文件名, 文件扩展名

1.6 Module contents

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python 模块索引

S

索引

F

G

I

L

N

P

R

S

T

U

V