An Embedded Toolset for Human Activity Monitoring in Critical Environments

Marco Di Benedetto[a] (marco.dibenedetto@isti.cnr.it), Fabio Carrara[a] (fabio.carrara@isti.cnr.it), Luca Ciampi[a] (luca.ciampi@isti.cnr.it), Fabrizio Falchi[a] (fabrizio.falchi@isti.cnr.it), Claudio Gennaro[a] (claudio.gennaro@isti.cnr.it), Giuseppe Amato[a] (giuseppe.amato@isti.cnr.it)

[a] Institute of Information Science and Technologies of the National Research Council of Italy (ISTI-CNR), Pisa, Italy

Corresponding Author:
Marco Di Benedetto
Institute of Information Science and Technologies of the National Research Council of Italy (ISTI-CNR), Pisa, Italy
Email: marco.dibenedetto@isti.cnr.it

An Embedded Toolset for Human Activity Monitoring in Critical Environments

Marco Di Benedetto [a,1,*], Fabio Carrara[a,1], Luca Ciampi[a,1], Fabrizio Falchi[a], Claudio Gennaro[a], Giuseppe Amato[a]

[a] *Institute of Information Science and Technologies of the National Research Council of Italy (ISTI-CNR), Pisa, Italy*

Abstract

In many working and recreational activities, there are scenarios where both individual and collective safety have to be constantly checked and properly signaled, as occurring in dangerous workplaces or during pandemic events like the recent COVID-19 disease. From wearing personal protective equipment to filling physical spaces with an adequate number of people, it is clear that a possibly automatic solution would help to check compliance with the established rules. Based on an off-the-shelf compact and low-cost hardware, we present a deployed real use-case embedded system capable of perceiving people's behavior and aggregations and supervising the appliance of a set of rules relying on a configurable plug-in framework. Working on indoor and outdoor environments, we show that our implementation of counting people aggregations, measuring their reciprocal physical distances, and checking the proper usage of protective equipment is an effective yet open framework for monitoring human activities in critical conditions.

Keywords: Deep Learning, Computer Vision, Machine Learning, Personal Protective Equipment, Counting, Homography, Embedded System

*Corresponding author.

Email addresses: marco.dibenedetto@isti.cnr.it (Marco Di Benedetto), fabio.carrara@isti.cnr.it (Fabio Carrara), luca.ciampi@isti.cnr.it (Luca Ciampi), fabrizio.falchi@isti.cnr.it (Fabrizio Falchi), claudio.gennaro@isti.cnr.it (Claudio Gennaro), giuseppe.amato@isti.cnr.it (Giuseppe Amato)

[1] Authors contribute equally.

1. Introduction

As occurs during a severe health emergency event, there exist scenarios in which ensuring compliance to a set of guidelines becomes crucial to secure a safe living environment in which human activities can be conducted. As evidenced during the recent COVID-19 pandemic, wearing medical masks, avoiding the creation of large gatherings in confined places, and keeping a certain physical distance among people were the most common rules every government applied in their jurisdiction territories. However, human supervision could not always guarantee this task, especially in crowded scenes where checking usage of personal protection equipment or enforcing strict social behavior has to be continuously assessed to preserve global health.

In the past decades, Computer Vision applications have shown astonishing results in several daily life tasks. Automatic image analysis aimed at classifying, locating, and counting objects, as well as estimating the distance between different instances of objects, are typical examples of applications of Computer Vision technology, which can be a valuable tool to automatically monitor human activities in critical environments through images captured by networked cameras.

This work presents an embedded modular Computer Vision-based and AI-assisted system that can carry out several tasks to help monitor individual and collective human safety rules. We strive for a real-time but low-cost system, thus complying with the compute- and storage-limited resources availability typical of off-the-shelves embedded devices, where images are captured and processed directly onboard. Our solution consists of multiple modules relying on well-researched neural network components, each responsible for specific functionalities that the user can easily enable and configure. In particular, by exploiting one of these modules or combining some of them, our framework makes available many capabilities. They range from the ability to estimate the so-called social distance (i.e., the physical distance among pedestrians) to the estimation of the number of people present in the monitored scene, as well as the

possibility to localize and classify personal protective equipment (PPE) worn by people (such as helmets, high-visibility clothing, and face masks) that the World Health Organizations has recommended as one of the primary tools to curb the spread of the disease, like, for example, the recent COVID-19 pandemic.

To validate our solution, we test all the functionalities that our framework makes available, exploiting two novel datasets that we collected and annotated on purpose and representing another contribution of our work. Specifically, we gathered the first dataset of images captured by a smart camera located in a public square in the city of Pisa, Italy, that represents a typical scenario for which it is crucial to check compliance with the safety rules, such as the maintaining of the social distance or the monitoring of the occupancy area. Moreover, we collected and annotated a second dataset comprising images containing pedestrians with and without PPE, such as helmets, high-visibility vests, and face masks. The peculiarity of this dataset is that part of the images are gathered from the GTA V video game and automatically annotated by the graphical engine. Experiments show that our system can effectively carry out all the functionalities that the user can set up, providing a valuable asset to monitor compliance with safety rules automatically.

To summarize, the main contributions of this work are the following:

- We introduce an expandable and flexible Computer Vision-based and AI-assisted embedded system, deployed in a real use-case scenario, capable of automatically monitoring human activities in critical environments, where individual and collective safety must be constantly checked. We base our solution on modules responsible for specific tasks that the user can easily configure and add to the whole system, providing many functionalities such as estimating the number of pedestrians present in the scene, measuring the social distances among people, and detecting PPE worn by individuals.

- We collect and annotate two novel datasets that we exploit to validate our framework. The first one, named *CrowdVisorPisa*, is gathered from

a camera in a public square of the city of Pisa, Italy, and represents a typical scenario for which it can be essential to monitor compliance with the safety rules, such as the observation of social distance. The second is instead a collection of images, partially synthetic, representing pedestrians with and without wearable PPE.

- We conduct experiments evaluating all the modules and the functionalities, which our framework makes available *in an embedded and deployed off-the-shelf device*, showing that our solution may be a valid aid to monitor and handle critical environments drastically reducing human supervision.

We organize the rest of this paper as follows. We review similar works in Section 2, and we introduce our modular framework and its plug-ins in Section 3. In Section 4, we describe the exploited datasets along with the adopted training procedures. In Section 5, we show our experiments, also discussing and analyzing the obtained results. Finally, we conclude the paper with Section 6, suggesting some insights on future directions.

2. Related Work

Due to the COVID-19 pandemic, many Computer Vision-based works have been recently published to help monitor human activities analyzing images, especially on the specific task of evaluating the social distance between people. For example, the Inter-Homines framework, presented in Fabbri et al. (2020), evaluates in real-time the contagion risk in a monitored area by analyzing video streams. The system includes occlusion correction, homography transformation, and people anonymization. People are located in the space exploiting the CenterNet (Zhou et al., 2019) object detector, and interpersonal distances are then calculated. Results are evaluated on the JTA dataset (Fabbri et al., 2018) (i.e., in a virtual world). In Saponara et al. (2021), the YOLO9000 detector (Redmon & Farhadi, 2017) has been exploited to detect people; centroids of

the found bounding boxes are then computed to evaluate the distance between them. Similarly, in Ahmed et al. (2021b), a platform for social distance tracking in top perspective video frames based on YOLOv3 (Redmon & Farhadi, 2018) was presented. Here too, centroids of the bounding boxes are used to estimate distances. A subset of the same authors also presented in Ahmed et al. (2021a) a social distance framework based on the Faster-RCNN detector (Ren et al., 2017). On the other hand, the authors in Punn et al. (2020) exploited YOLOv3 (Redmon & Farhadi, 2018) to detect humans and Deepsort (Wojke et al., 2017) to track people. They conducted experiments on the Oxford town center surveillance footages (Benfold & Reid, 2011). The usage of Faster R-CNN (Ren et al., 2017) and YOLOv4 (Bochkovskiy et al., 2020) to detect pedestrians are discussed in Yang et al. (2021) to monitor social distancing and density. Monitoring of workers to detect social distancing violation that uses Mobilenet-V2 (Sandler et al., 2018) to detect people is introduced in Khandelwal et al. (2020).

Another task recently tackled in literature, again related to the COVID-19 pandemic, is face mask detection. For example, the authors in Kong et al. (2021) presented an edge computing-based mask identification framework (ECMask). It consists of three main stages: video restoration, face detection (inspired by FaceBoxes (Zhang et al., 2017b)), and mask identification (based on Mobilenet-V2 (Sandler et al., 2018)). Deep learning models were trained and evaluated on the Bus Drive Monitoring Dataset, which unfortunately is not publicly available. Authors in Eyiokur et al. (2021) developed a deep learning-based computer vision system able to perform face mask detection but also face-hand interaction detection. A more comprehensive literature review of applications of artificial intelligence in battling against COVID-19 is given in N. (2021) including social distancing and face mask detection.

Differently from most other works, in this paper, we present a *modular* and *expandable* Computer Vision-based embedded system that can fulfill *multiple* tasks to help monitor compliance of individual and collective human safety rules in critical scenarios, like the one caused by the COVID-19 pandemic. The

main peculiarities are that it runs directly on a low-cost computing device. The user can easily enable and configure the available functionalities ranging from computing social distances, estimating the number of people present in the scene, or detecting PPEs, by combining more modules and building more complex tasks.

3. Modular Framework

The general purpose of our monitoring system is to be embeddable on low-cost devices and, above all, to be expandable to different features in demanding situations. To this end, we designed a framework able to orchestrate a set of internal and user-defined *plugins*, each dedicated to a single task. Specifying inputs and outputs makes it possible to create a dependency graph. Each sub-module represents a node, and each pair of matching input-output represents an edge. In this way, given the desired output, a *topological sort* is executed to minimize and linearize the sequential execution of computations. Although an easier solution may exist in the context of the plugins, our methodology is relatively simple to implement and allows low-cost systems to execute any complex compute graphs in a sequential and semantically-correct way.

An overview of our modular framework is depicted in Figure 1. Video frames are taken at regular intervals from one or more cameras and processed locally. Multiple video streams can be multiplexed and handled by a single system instance. Current modules include a) Pedestrian Detector, b) Density-based Pedestrian Counter, c) Instance-based Pedestrian Counter, d) Pedestrian Tracker, e) PPE Detector, and f) Interpersonal Distance Measurer; Figure 2 exemplifies the results of the analyses performed by each module, whereas their detailed description is reported in the following sections. All the modules are toggleable; the Instance-based Pedestrian Counter, Pedestrian Tracker, Interpersonal Distance Measurer, and PPE Detector modules depend on the output of the Pedestrian Detector module and require it to be active. Results of the active modules are combined and provided in JSON format to be consumed by

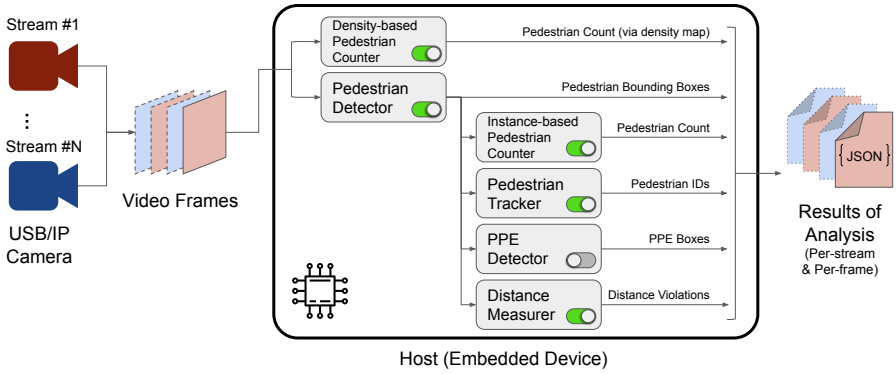

Figure 1: Overview of our modular framework. Multiple video streams can be multiplexed and handled by a single instance. Results are generated in JSON format and can be routed to downstream services. Our system is *flexible* and *expandable*, as modules can be activated or deactivated depending on the user's needs, and novel functionalities can be introduced with additional custom modules.

downstream services. Note that video frames are analyzed onboard and never stored; this enables privacy-aware solutions where captured images never leave the edge devices.

3.1. Detecting Pedestrians

The pedestrian detector is the system's main component on which almost all other plug-ins rely. Its primary purpose is to localize and classify pedestrian instances from input images. These detections constitute the main data that will be exploited, in different ways, by the other nodes of the system.

We base our pedestrian detector on *Faster R-CNN* Ren et al. (2017), a popular state-of-the-art CNN-based object detection system. It operates as a two-stage algorithm, exploiting two different modules during the different phases of its detection pipeline. In the first stage, a CNN acts as a backbone by extracting input image features. Starting from this features' space, the Region Proposal Network (RPN) is responsible for generating the region proposals that might contain objects, slicing pre-defined region boxes (called anchors), and ranking them, suggesting the ones most likely containing objects. The second

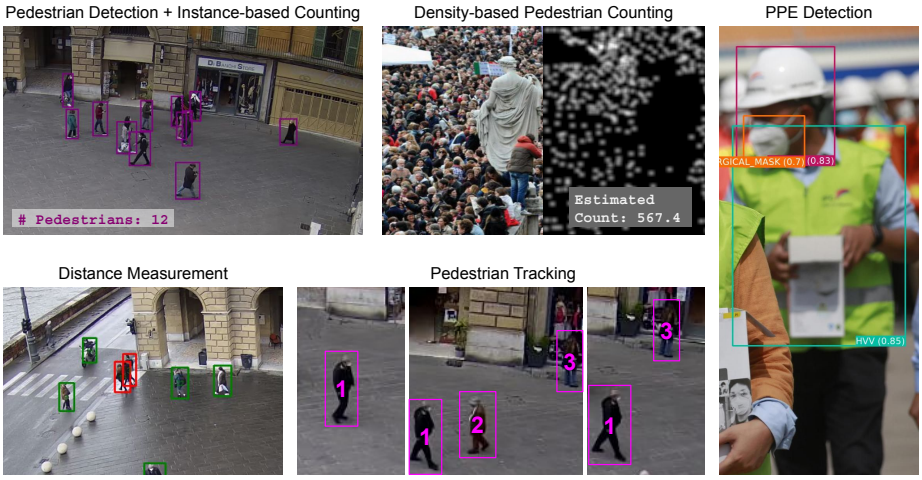

Figure 2: Visualization of output examples of the modules currently available in our system. The outputs of each module are the following. **Pedestrian Detection & Instance-based Counting**: list of pedestrian bounding boxes and respective count. **Pedestrian Tracking**: numeric ID assigned to detected pedestrians persisting through frames. **Density-Based Pedestrian Counting**: estimated number of pedestrians (and, optionally, the density map). **Distance Measurement**: IDs of groups of pedestrians violating a predefined distance. **PPE Detection**: list of PPE bounding boxes detected per pedestrian.

module is the Fast R-CNN detector (Girshick, 2015) that classifies and localizes the objects inside the proposed regions, outputting class scores together with bounding boxes coordinates.

We preferred a two-stage detector to single-shot detectors, as we could easily extract features from the region proposal stage and make them directly available for subsequent processing. We used the extracted features when tracking pedestrians, and they could be used in other future modules, such as cross-camera pedestrian re-identification. Moreover, Faster R-CNN is widely adopted and usually guarantees a state-of-the-art detection performance. However, the modularity of our system does not limit us to Faster R-CNN, and another object detector can be easily adopted in the future, just replacing the related module.

We specialize the Faster R-CNN detector to localize pedestrian instances, performing a supervised domain adaption that exploits several pedestrian datasets.

We detail all these strategies in Section 4.

3.2. Tracking Pedestrians

Object tracking can be an essential tool to increase the robustness to spurious detections and achieve temporal consistency in video analysis. To this end, we implement and apply an object tracker over pedestrian detection to reidentify people among consecutive video frames. This step is beneficial for assessing temporal rules, such as raising alarms after the same pedestrian occupies a forbidden area for more than a predefined amount of time.

The implementation of the tracker follows the formulation of DeepSort (Wojke et al., 2017). It is based on SORT (Bewley et al., 2016), a simple causal tracking algorithm for 2D objects in which targets are represented by the position and area of the bounding box and their speed of variation. The state of each target (also known as *tracklet*) is updated with available detections using a Kalman filter framework. DeepSort builds upon SORT by adding a matching scheme between predicted and actual targets based on feature vectors that describe the appearance of tracked objects; tracklets can be confirmed if the cosine score between feature vectors of the predicted and actual target is above a programmable threshold. We refer the reader to Wojke et al. (2017) for an evaluation of DeepSORT compared to other tracking methods.

Feature vectors in DeepSort must be provided by extracting representations from detected regions with an additional pre-trained network. In our implementation, we avoid this step by reusing the feature vectors of detected regions that the object detection network has already extracted; in particular, we perform a Region of Interest (RoI) average pooling of the features extracted by the CNN backbone using only the regions provided by the pedestrian detection module.

3.3. Crowd Counting

In some scenarios where individual and collective safety has to be constantly monitored, like people aggregations during the recent COVID-19 pandemic, estimating the number of people present in a region of interest is crucial to

monitor the occupancy area. By measuring and limiting the number of people who can visit a location at any one time, it is possible to drastically reduce the likelihood of setting up people gatherings and, consequently, minimize human virus transmission. Our solution relies on a dedicated plug-in that can work in two different modalities that the user can conveniently pick out, depending on the considered scenario. The first one, named *Counting by Instances*, is better suited for not particularly crowded environments and relies on the pedestrian detector described in Section 3.1. The second, named *Counting by Density Estimation*, is instead a more holistic approach more appropriate for highly crowded scenarios; it aims at computing a mapping between the features of the captured image and its pedestrian density maps, skipping the detection of the single instances. The estimated number of people present in the controlled area can then be obtained by integrating this density map. In the following paragraphs, we describe in detail both modalities.

Counting by Instances. This counting modality depends entirely on the pedestrian detector. Specifically, the pedestrian detection module provides the input, i.e., the localized pedestrian instances. The counting by instances plug-in is only responsible for counting them. As already mentioned, this approach has some limitations in highly crowded scenarios since, in this case, people instances are heavily occluded and not easily identifiable.

Counting by Density Estimation. This modality tackles the counting task as a supervised regression problem from the image features to an associated density map, following the seminal work Lempitsky & Zisserman (2010), avoiding the detection of individual object instances. As mentioned above, this approach is desirable in highly congested scenarios, where the instances of the objects are not completely visible due to occlusions. The input of this module consists directly of the captured image, so this node does not depend on the pedestrian detector module.

In this scenario, the most widely used labels needed for the supervised training are the dotted annotations, obtained by putting a single dot on each object

instance in each image. Formally, we assume to have a set of N training images $I_1, I_2, ..., I_N$. We also assume that each image I_i is labeled with a set of 2D points $\mathbf{P}_i = P_1, ..., P_{K(i)}$, where $K(i)$ is the total number of annotated objects (in our case pedestrians). For a training image I_i, we define the ground truth density map as

$$\forall p \in I_i, \quad H_i(p) = \sum_{P \in \mathbf{P}_i} \delta(p - P). \tag{1}$$

Here, p denotes a pixel, while a point identifying a pedestrian is represented as a delta function. Converting it into a continuous density function with Gaussian kernel G_σ we obtain

$$\forall p \in I_i, \quad F_i(p) = \sum_{P \in \mathbf{P}_i} \delta(p - P) * G_\sigma. \tag{2}$$

The sum of the density map is equivalent to the total number of pedestrians. It is worth noting that the Gaussian spread parameter σ depends on the size of each pedestrian in the image, considering the perspective transformation. However, it is almost impossible to obtain the occluded object's size manually in a high-density environment. So this parameter is a dataset-specific quantity empirically estimated. Then, given a set of training images together with their ground truth densities, we aim to learn a transformation of the feature representation of the image that approximates the density function at each pixel to minimize the sum of the mismatches between the ground truth and the estimated density functions (the *loss* function).

We build our density map estimator upon the Congested Scene Recognition Network (CSRNet) (Li et al., 2018), a CNN-based algorithm that can understand highly congested scenes and that has been successfully adopted in many crowded scenarios. It comprises two major components. For the image features extraction, it exploits a modified version of the well-known VGG-16 network (Simonyan & Zisserman, 2015), where the final classification part, i.e., the final fully-connected layers, is removed. The output size of this front-end network is 1/8 of the original input size. Following other works (Yu & Koltun, 2016; Chen et al., 2018, 2017), a back-end composed of dilated convolutional layers

are stacked upon this front-end to extract deeper information of saliency and, at the same time, maintain the output resolution. Using dilated convolutions, we can deliver larger reception fields while replacing pooling operations (e.g., the max pool operation) that are often responsible for losing quality in the density generation procedure. We refer the reader to Li et al. (2018) for a detailed comparison of CSRNet against other state-of-the-art methods.

3.4. Measuring Social Distances

A critical condition that must be kept under control in dynamic environments where an infection is ongoing is represented by the physical distance among individuals. In the case of air-borne diseases, it is thus very common to issue rules to avoid people gatherings in confined places and keep a specific reciprocal separation to contrast the spread of pathogen agents. Although crowd counting is effective in monitoring aggregations, measuring distances among people becomes critical during pandemic events. Assuming that individuals mostly hang out on the same planar floor, we decided to measure their actual distance by applying a simple pre-calibrating step to the fixed monitoring camera, using a proper geometrical transformation that places detected items on a common system of reference. Our solution is to pre-compute a mapping between real points in the scene whose relative position is known and their projection on the acquired frame image. This process is well known in Computer Vision and consists of finding a *homography*, i.e., a perspective transformation that projects on two different points of view a set of 3D points lying on the same plane.

More in detail, a d-dimensional homography represents the linear operation

$$x' = Hx \quad x, x' \in \mathbb{R}^d,$$

and is expanded in homogeneous 3D coordinates as an approximation of the projection operation of a pinhole camera, represented by a 4×4 matrix with the translation and projective components added. In the case of coplanar points, we can consider the input z component as a constant and thus eliminate it from

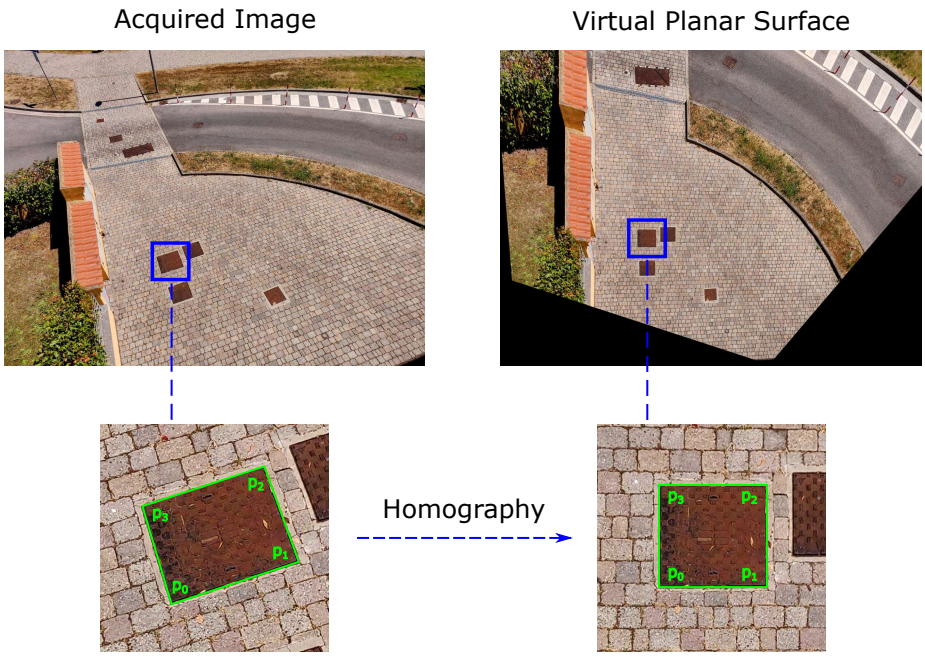

Figure 3: An example of homography. A set of four points in the acquired image is mapped through homography to its projection on a Euclidean metric space laying on a virtual planar surface.

the above formulation:

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

Thus, the eight h_{ij} unknown can be found if we can relate two sets of four 2D points through homography. To calculate its elements, we minimize the matching error of the two sets after perspective projection (i.e., dividing by the homogeneous component to take point distance to the camera into account, see Parent (2012)) to bring coordinates from homogeneous to euclidean space:

$$\begin{aligned} w' &= h_{31}x + h_{32}y + h_{33}, \\ x' &= \frac{h_{11}x + h_{12}y + h_{13}}{w'}, \\ y' &= \frac{h_{21}x + h_{22}y + h_{23}}{w'}. \end{aligned}$$

We can set up the projected image of four coplanar (but not collinear) points to a virtual flat surface and then use the found transformation matrix to approximate the relative location of any point laying on the same plane, as shown in Figure 3. This calibration step is kept as straightforward as possible for an untrained operator, as it is easy to select four points of a known-sized quadrangle (e.g., a standardized manhole). As perturbations may occur with point selection, we also allow to select more than four coplanar points whose relative position is known and then use a *random sample consensus* (RANSAC (Fischler & Bolles, 1987)) algorithm to iterate through a random selection of four points for a first approximation of the plane equation, which is eventually refitted to minimize the error.

3.5. Detecting Personal Protective Equipment

A simple intervention for protecting health and well-being is wearing Personal Protective Equipment (PPE). It is particularly true in dangerous working environments, such as wearing harnesses and helmets on construction sites. Still, it also became evident in light of the recent COVID-19 pandemic, where wearing

face masks can prevent infections. Therefore, we implement a module dedicated to detecting worn PPE, essential for ensuring compliance with regulations that imply personal protection.

Our solution for PPE detection follows the same methodology already adopted for pedestrian detection: specifically, we adopt the same detector architecture based on Faster-RCNN (see Section 3.1 for details). The PPE detector network, differently from the pedestrian detector, takes a rather small image depicting a pedestrian as input. It is trained to distinguish and detect several classes of worn PPE, i.e., surgical/face masks, helmets, and high-visibility vests. The pedestrian detector module provides the input of this module: once detected, the patch of the video frame depicting a pedestrian is given as input to the PPE detector that provides bounding boxes of PPE if the pedestrian wears them.

Conceptually, the detection of PPE could be tackled by the pedestrian detector module by adding the PPE classes to the base detection model. However, we empirically noticed that merging the PPE detection with the pedestrian detection module leads to performance degradation in both tasks. Using separate modules provides a more flexible solution in which the input image resolution of both detectors can be adjusted separately and better adapted to the monitored scenario. For example, when wide areas are monitored, PPE detection is invoked several times, depending on how many pedestrians are detected, on small patches of pedestrians. In this case, the PPE detection network can be configured to work on lower resolution inputs to maintain an affordable computational cost.

4. Datasets and Architecture Adaptions

A key point in producing a verification system that can generalize on a broad spectrum of working conditions is to generate a training set based on an adequately large amount of environmental conditions. In our case, this means accessing a massive amount of images involving people under different scenarios. Manually annotating new images collections is expensive and requires a notable

human effort. Instead, a recently promising approach is to gather data from virtual world environments that resemble the characteristics of the real-world scenarios and where the labels can be acquired with an automated process. Thus, in this work, we build vast training datasets, considering both real-world and synthetic images from public datasets when available, and collecting others when needed, covering a multitude of different scenarios and contexts. Hereafter, we describe these data, dividing them according to the module for which they are employed. Furthermore, we describe the exploited training procedures, highlighting the changes we made to the architectures to adapt them to our specific scenarios.

4.1. Datasets for Pedestrian Detection

We use many popular publicly available pedestrian detection datasets to train the pedestrian detector module. Furthermore, we introduce a novel dataset, named *CrowdVisorPisa*, that we also employ for evaluating our solution. In the following, we detail all the exploited datasets.

Virtual Pedestrian Dataset (ViPeD) (Amato et al., 2019; Ciampi et al., 2020). The Virtual Pedestrian Dataset is a *synthetic* collection of images generated exploiting the highly photo-realistic graphical engine of the video game Grand Theft Auto V (GTA V) by Rockstar North. It comprises about 500K images belonging to 512 different urban environments (256 for training and 256 for testing) characterized by various weather conditions, illumination, perspectives, viewpoints, and density of people. Labels are *automatically* provided by the game engine and consist of bounding boxes precisely localizing the pedestrians present in the scenes. More details on the generation of ViPeD can be found in Amato et al. (2019); Ciampi et al. (2020).

MOT17Det (Milan et al., 2016) and MOT20Det (Dendorfer et al., 2019). The MOT17Det and MOT20Det datasets are two collections of images (5,316 and 8,931, respectively), annotated with bounding boxes, taken from multiple sequences describing crowded scenarios having different characteristics, like view-

points, weather conditions, and camera motions. The authors provided training and test subsets, but they released only the ground-truth labels for the former. The main difference between MOT20Det and MOT17Det is that the first contains more crowded scenarios.

CityPersons (Zhang et al., 2017a). The CityPersons dataset consists of a set of stereo video sequences recorded from a moving car in streets from different cities in Germany and neighboring countries. In particular, the authors provide 5,000 frames from 27 cities labeled with bounding boxes and split across train/validation/test subsets.

CrowdHuman (Shao et al., 2018). CrowdHuman is a benchmark dataset for pedestrian detection. It comprises 15,000, 4,370, and 5,000 images for training, validation, and testing, respectively, describing diverse, crowded scenarios, with an average number of persons in an image of 22.6. The authors annotated each human instance with a head bounding box, a human visible-region bounding box, and a full-body bounding box.

PRW (Zheng et al., 2017). The PRW dataset contains 11,816 frames where 932 different pedestrian identities are annotated with their bounding boxes. The authors provide the training and the test splits.

CUHK-SYSU (Xiao et al., 2017). The CUHK-SYSU is a large-scale benchmark dataset containing 18,184 images, 8,432 different identities, and 96,143 pedestrian bounding boxes. It is divided into training and test subsets.

CrowdVisorPisa (ours). The CrowdVisorPisa dataset[2] is a novel collection of images that we collected and annotated on purpose for this work. In particular, we stored 15 different sequences gathered from a webcam located in a public square of the city of Pisa, Italy, each of which comprises ten images captured with a time interval of 1 second. We manually labeled all frames, localizing

[2] We provide the CrowdVisorPisa and CrowdVisorPPE datasets upon request.

Figure 4: A sample from our novel *CrowdVisorPisa* dataset, together with bounding box annotations localizing pedestrians.

the pedestrian instances with bounding boxes. Furthermore, we also annotated each sequence taking track of the different pedestrian entities entering or exiting the scene. We divided the dataset into train and test splits, considering 10 and 5 different sequences, respectively. The former split is exploited to train the pedestrian detector module, while the latter is used to evaluate the performance of some modules of our framework. It is worth noting that, due to camera positioning not modifiable for local restrictions, this dataset represents a particularly challenging scenario as people instances are small and sometimes difficult to localize. A sample of the dataset is shown in Figure 4.

4.2. Datasets for PPE Detection

CrowdVisorPPE (ours). We collect and annotate a novel dataset (named *CrowdVisorPPE*[2]) to train and evaluate our PPE detection module. It comprises 54,017 images representing pedestrians with and without wearable PPE. Roughly half of the dataset comprises synthetic images procedurally generated using the GTA V video game engine as in Di Benedetto et al. (2020), whereas the other half comprises real-world photographic images taken from the Web and manu-

		# PPE instances		
Train Split	# img	Helmet	HVV	Mask
GTA V (**V**)	28,078	9,575	21,374	0
Web (**R**)	21,820	10,673	10,686	1,630
Test Split				
Web (**R**)	4,119	2,163	2,017	271

Table 1: Details of the *CrowdVisorPPE* dataset. **V** = virtual/synthetic data; **R** = real/photographic data.

ally annotated. The PPE classes of interest, i.e., helmets, high-visibility vests (HVVs), and face masks, are annotated with bounding boxes. The real-world subset is the only source of face mask instances since they are not available for rendering in GTA V. We hold out a subset of real images as the test split, whereas synthetic images and the remaining real ones form the training split. We show the dataset details in Table 1 and some samples in Figure 5.

4.3. Datasets for Crowd Counting by Density Estimation

To train the module responsible for crowd counting by density estimation, we exploit many publicly available datasets, detailed in the following.

GTA5 Crowd Counting (GCC) (Wang et al., 2019). The GCC dataset is a large-scale and diverse synthetic crowd counting dataset, gathered from the video-game Grand Theft Auto V (GTA5) and automatically annotated. It consists of 15,212 images, with a resolution of 1080×1920, containing 7,625,843 persons in 400 different scenarios with various locations, weather conditions, and crowd densities. Compared with the existing datasets, GCC is a more large-scale crowd counting dataset in both the number of images and persons.

ShanghaiTech (Zhang et al., 2016). The ShanghaiTech dataset is a large-scale crowd dataset of nearly 1,200 manually dot-annotated images with a total of

(a) GTA V (Virtual) (b) Web (Real)

Figure 5: Samples from our novel *CrowdVisorPPE* dataset. PPE classes are color coded: helmet, high-visibility vest, face mask.

330,165 people with centers of their heads. This dataset consists of two parts: part A, containing 482 images crawled randomly from the Internet, and part B, composed of 716 images taken from the busy streets of metropolitan areas in Shanghai. The crowd density varies significantly between the two subsets, making this dataset more challenging. The two parts are divided into training and testing subsets: 300 images of part A are used for training, and the remaining 182 images for testing, while 400 images of part B are for training and 316 for testing.

UCF-QNRF (Idrees et al., 2018). The UCF-QNRF dataset is a collection of images gathered from three sources: Flickr, Web Search, and the Hajj footage. The authors performed the entire annotation process in two stages, the first one for the labeling and the second one for the verification, for a total of 2,000

human-hours spent through to its completion. This dataset comprises 1,535 images with more than 1 million dot-annotations on the centers of the pedestrian's heads, divided into training and test subsets.

NWPU-Crowd (Wang et al., 2021). The NWPU dataset is a large-scale congested crowd counting and localization dataset consisting of 5,109 images, in a total of 2,133,375 annotated heads with points and boxes. Compared with other real-world datasets, it has the most extensive density range. Another peculiarity of this dataset is that it also comprises some negative samples like high-density crowd images to assess the robustness of models.

4.4. Adaptation for Pedestrian Detection

To make the pedestrian detector able to run efficiently directly on computational- and resource-limited devices, we employ, as the backbone of Faster R-CNN, the *ResNet50* architecture, a lighter version of the popular *ResNet101* (He et al., 2016). We start considering the detector pre-trained on the *COCO* dataset (Lin et al., 2014), a large collection of images depicting complex everyday scenes of ordinary objects in their natural context, categorized into 80 different classes. In our case, we have to localize and identify objects belonging to just one class (i.e., pedestrian). To this end, we further simplify the model by reducing the number of the final fully convolutional layers responsible for classifying the detected objects, making our detector lighter. We call *Light* this modified version of the pedestrian detector module to distinguish it from the *Full* original one, having instead the *ResNet101* backbone and a larger number of fully connected layers.

Intending to specialize the detector in finding the specific pedestrian object category, we adopt a supervised *domain adaptation* strategy, exploiting the datasets described in Section 4.1 and fine-tuning the network to this specific task. Following Ciampi et al. (2020), we employ the *Balanced Gradient Contribution* (BGC) (Ros et al., 2016a,b) strategy, where, during the training phase, we mix the synthetic data, taken from *ViPeD*, and the real-world images

gathered from the remaining datasets. In this way, as already demonstrated in Ciampi et al. (2020), we boost the performance of the detector compared to a model relying only on real-world data, taking advantage of the great variability and size of *ViPeD*, and, at the same time, mitigating the existing domain shift between these synthetic data and the real-world ones. In particular, during the training phase, we exploit batches composed of 2/3 of synthetic images and 1/3 of real-world data, thus considering statistics from both domains throughout the entire procedure and where the real-world data acts as a regularization term over the synthetic data training loss. We refer the reader to Ciampi et al. (2020) for details performance comparisons against other state-of-the-art methods.

4.5. Adaptation for PPE Detection

As in the pedestrian detector, we adopt the Faster R-CNN model with the *ResNet50* backbone as the PPE detector architecture. The methodology used to obtain the trained PPE detector follows Di Benedetto et al. (2020) and reaches a comparable detection performance: we start from a detector pre-trained on COCO with a new detection head that matches the number of the PPE classes, and then we use a mixture of synthetic and real images of pedestrians with PPE when training the model, to finally testing it on real data only. We refer the reader to Di Benedetto et al. (2020) for comparisons against other detection models and the model trained only on real-world data. The only difference concerning the object detector and Di Benedetto et al. (2020) is that we perform PPE detection only on pre-segmented patches containing a single pedestrian instead of searching for PPE in the entire video frame. This simplifies the task for the model and enables us to save computational budget by processing smaller images.

4.6. Adaptation for Crowd Counting by Density Estimation

To train the density-based pedestrian counter module, we adopt a supervised domain adaptation strategy consisting of training the network with the synthetic data and then fine-tuning it exploiting the real-world images, that has already

been proved to be effective in Amato et al. (2019) and Ciampi et al. (2020), providing a performance boost compared to models trained only on real-world data. In particular, we set the initial weights of the network layers with values coming from a Gaussian distribution with 0.01 standard deviation. Then, we train the network exploiting the GCC dataset, and, finally, we fine-tune it using the real-world data.

5. Experiments and Results

We evaluate all the modules making up our framework, considering different scenarios and exploiting appropriate metrics depending on the task. For all the experiments, we consider the *Light* version of our object detector module since it has shown similar performance compared with the *Full* version, and it is more appropriate in combination with low-cost and computational-limited hardware.

Being our target a deployable monitoring system, we selected the NVIDIA Jetson TX2 embedded device as the hardware host. It comprises two 64-bit CPUs with two and four cores each, an NVIDIA Pascal GPU with 256 CUDA cores, 8 GB of RAM shared between the system and the graphics accelerator, and a 32 GB solid-state storage volume. The operating system is NVIDIA's Linux4Tegra (L4T) distribution based on Ubuntu. At the time of writing, the cost of the device was less than USD 500. We installed Python 3.8 with OpenCV 4.5 and the deep learning framework PyTorch 1.8. As detailed in Table 2, memory usage is kept within 5 GB of both system and GPU RAM. An external USB camera completes the whole installation.

5.1. Counting by Instances

In this setting, we test and evaluate the counting by instance functionality. We consider our *CrowdVisorPisa* dataset and, in particular, the five sequences belonging to the test subset, performing two different sets of experiments over it; the first one involves only the pedestrian detector module, and the second instead takes also into account the tracker module. More in detail, in the first

PD	DC	PPE	SysRAM	GpuRAM
Light	✗	✗	2.36	0.55
	✓	✗	2.44	0.86
	✗	✓	2.35	2.10
	✓	✓	2.44	2.20
Full	✗	✗	2.51	0.62
	✓	✗	2.52	0.94
	✗	✓	2.51	2.20
	✓	✓	2.51	2.30

Table 2: System and GPU Memory Usage in GB. **PD** = pedestrian detector model type; **DC** = whether the density counter module is active; **PPE** = whether the PPE detector module is active. The modular framework is assumed to always use the object detector in its *Light* or *Full* models, along with the enabled distance measure plug-in that consumes a fixed and negligible (less than 1 MB) amount of memory. Video stream size is 1173×880 RGB pixels. System memory is calculated with `/usr/bin/time -f "%M"`, GPU memory with `torch.cuda.max_memory_allocated()`.

case, we evaluate the effectiveness of our framework to estimate the number of people present in the single frames. On the other hand, in the second scenario, we also consider the temporal relation existing between consecutive images, tracking the found pedestrian instances over time.

We report in Figure 6 the obtained results concerning the first scenario. Each row of the figure represents a different sequence. The first column shows the number of people that our detector module can localize for each frame comprising a sequence, varying the detection threshold. On the other hand, in the second column, we illustrate the errors in terms of counting. We also report, for each sequence, the best Mean Absolute Error (MAE), i.e., the mean of the sum of the absolute errors, obtained with a specific threshold. As can be seen, we get a MAE close to 1 or 2, depending on the considered scenario, demonstrating that the module provides a reliable estimation of the number of pedestrians present in the monitored scene. The optimal threshold may vary depending on the scenario and the desired behavior, e.g., the user may prefer under- or over-estimation in case of errors. Due to the empirical nature

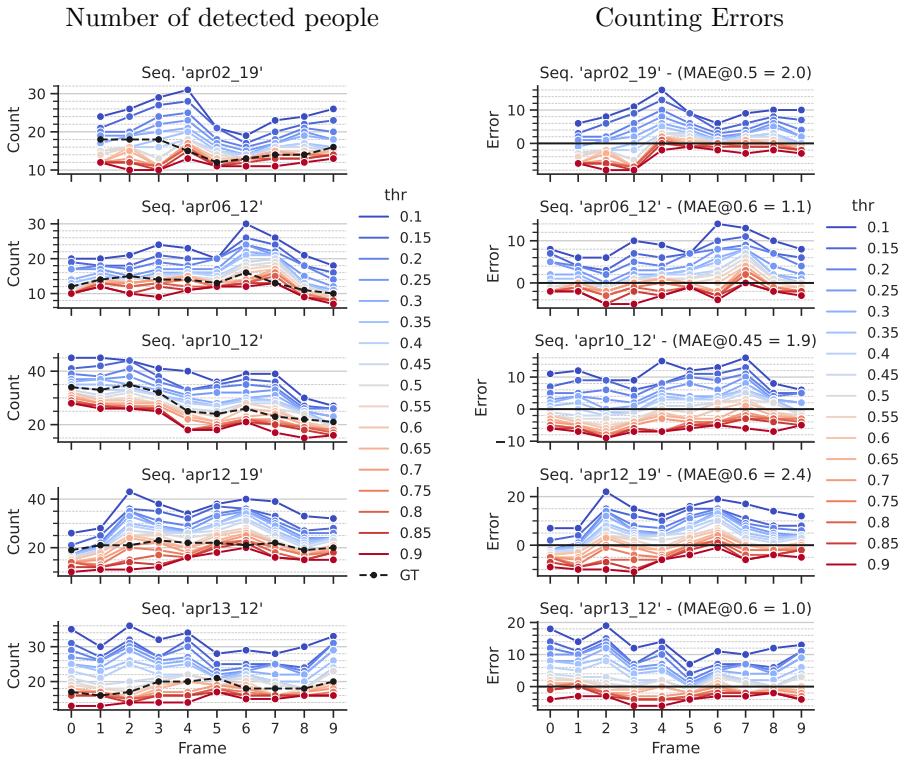

Figure 6: Evaluation of *counting by instances* functionality of our framework, considering the single still frames of the five test sequences of our *CrowdVisorPisa* dataset. In the first column, we report the number of people located by our detector, varying the detection thresholds. The black line (GT) indicates the actual number of pedestrians in the frame. The second column shows the counting errors and the best MAE obtained with a specific detection threshold.

of its choice, in our system, we provide the dynamical configuration of several parameters, including detection thresholds.

On the other hand, in Figure 7, we show the results concerning the second scenario. Each row of the figure corresponds to a different sequence. We report the results about the single frames making up a sequence for three different detection thresholds, one for each column. In particular, we indicate the pedestrians that enter and exit from the scene at each frame, exploiting the tracklets provided by the tracker module that represents the recognized identities of the people instances over time. Solid lines represent the actual number of people

present in a frame over time, and green/red candles represent the number of people entering/exiting the scene. Similarly, dashed lines represent the number of people predicted by our system, and yellow/blue candles represent the estimated number of people entering/exiting the scene as predicted by the tracker module. We notice that with a low (resp. high) threshold value, our system tends to overestimate (resp. underestimate) the total number of people present in a sequence, thus finding its optimal threshold values in the 0.5 - 0.6 range. We also note that false-positive detections tend to create spurious peaks in the people count. However, they often recovered in the immediate following frame.

5.2. Counting by Density Estimation

Given that the annotation procedure for labeling datasets having these characteristics is highly costly in terms of manual human effort, we exploited the test subsets of the already publicly available datasets described in Section 4.

We report in Table 3 the obtained results in terms of Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). It is worth noting that, as a result of the squaring operation, RMSE effectively penalizes large errors more heavily than small ones, thus more suitable when outliers are particularly undesirable. Furthermore, we also compute the Structural Similarity Index Measure (SSIM) (Wang et al., 2004) to measure the density map quality, which measures images' similarities under three aspects: brightness, contrast, and structure. The value of SSIM is in the $[0, 1]$ range: the larger it is, the less distortion of the image is measured. Finally, in Figure 8 we show some examples of the considered images, together with the ground truth and the predicted density maps.

5.3. Detecting Pedestrians and Personal Protection Equipment

We validate pedestrian and worn PPE detection, performed respectively by the Pedestrian Detector and the Personal Protection Equipment Detector modules. For the former, we focus on the five test sequences of our *CrowdVisorPisa* dataset, whereas for the latter, we consider the *CrowdVisorPPE* test subset.

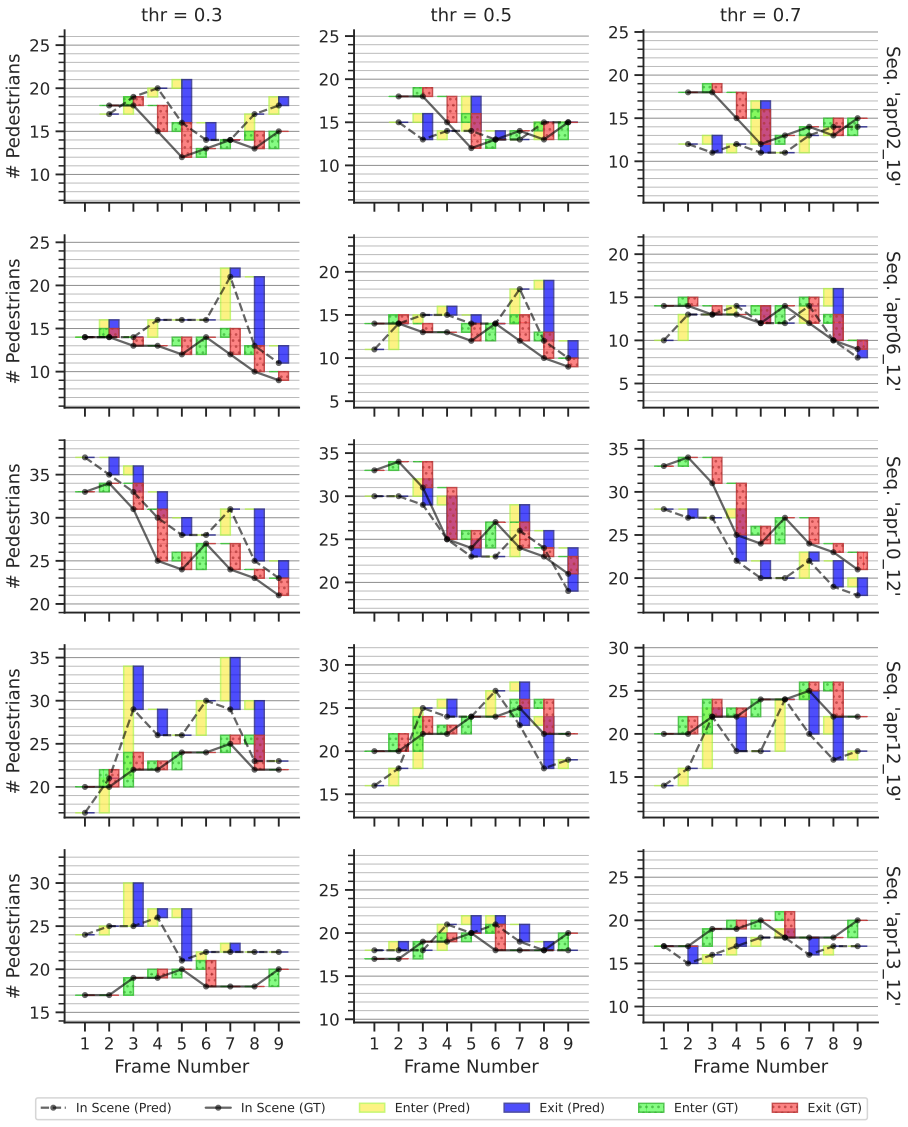

Figure 7: Evaluation of the counting by instances functionality of our framework, taking into account also the tracker module. We considered the five test sequences of our *CrowdVisorPisa* dataset reported one for each row. Columns represent the results obtained for three different detection thresholds. For each plot, we show the pedestrians that enter and exit from the scene, relying on the tracklets describing the recognized identities of the people instances over time.

PD	**PPE**	**DC**		
mAP ↑	mAP ↑	MAE ↓	RMSE ↓	SSIM ↑
0.836	0.606	92.28	365.4	0.79

Table 3: Evaluation metrics of the Pedestrian Detection (**PD**), Density Counter (**DC**), and **PPE** Detector modules, measured on the corresponding test sets. The mean Average Precision (mAP) measures the average precision of the detection when varying the score threshold in detection-based modules (**PD**, **PPE**). For **DC**, MAE and RMSE measure the counting error, while SSIM measures the quality of the predicted density map.

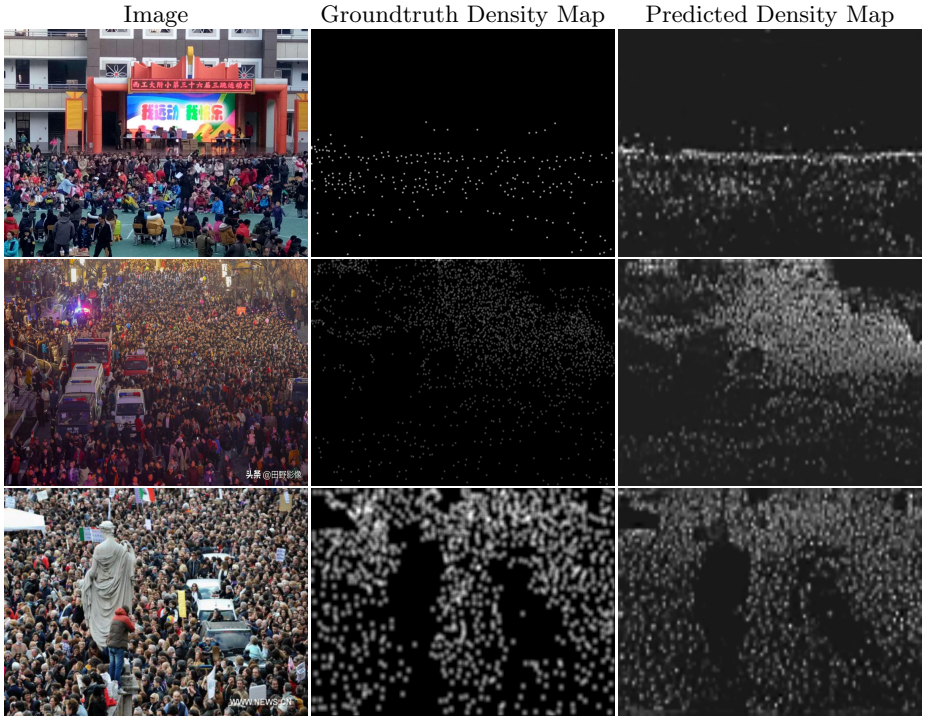

Figure 8: Some samples of the test subsets exploited for the evaluation of our *counting by density* estimation functionality, together with the ground truth and the predicted density maps. Integrating these density maps, i.e., summing up the pixel values, we can estimate the number of people present in the image.

As a performance metric, we adopt the mean Average Precision (mAP) with an IoU value >= 0.5, a popular metric in measuring the accuracy of object detectors that computes the average precision value for recall values spanning 0 to 1. We prefer the mAP over other threshold-dependent metrics, such as True Positive Rate, False Positive Rate, or F1-score. Indeed, threshold-dependent metrics are scenario- and application-specific; end-users may accept different levels of false positive or negative rates and may decide to tune thresholds differently depending on their needs. On the other hand, mAP provides a unique metric summarizing the performance at multiple operational points. We report the obtained results in Table 3, showing that our modules can reach a mAP of 0.836 and 0.606 for the pedestrian detection and the PPE detection tasks, respectively. Figure 9 shows some predictions of PPE detections on the CrowdVisorPPE test set.

5.4. Measuring Social Distances

To establish a correspondence between the acquired image and a planar *metric* surface onto which objects (i.e., pedestrians) positions can be evaluated, we used a known-sized manhole in the monitored scene to calculate the homography. The computed matrix is exploited to unwarp pixel position into real-world relative locations. The homography reprojection is a closed-form mathematical process, so no previous training is needed.

Our measuring results can be seen in Figure 10: in the examples shown, the precision of measurements is relative to both the initial calibration (i.e., manhole real size mapped to its projection on the input image) and the accuracy of the pedestrian bounding boxes (i.e., rectangles) predicted by the object detector. We measured the manhole with an upper bound precision of 1 cm and a pixel area error of about 3 cm, thus confining the overall measurements below the 10 cm error. For pedestrian positions, we used the midpoint of the lower edge of its predicted bounding box. As can be seen from the gridded unprojection of the examples, results are consistent within the above error gap.

Figure 9: Examples of predictions of the PPE Detection module on our CrowdVisorPPE test set. PPE classes are color coded (helmet, high-visibility vest, face mask), and the detection score is reported in parenthesis.

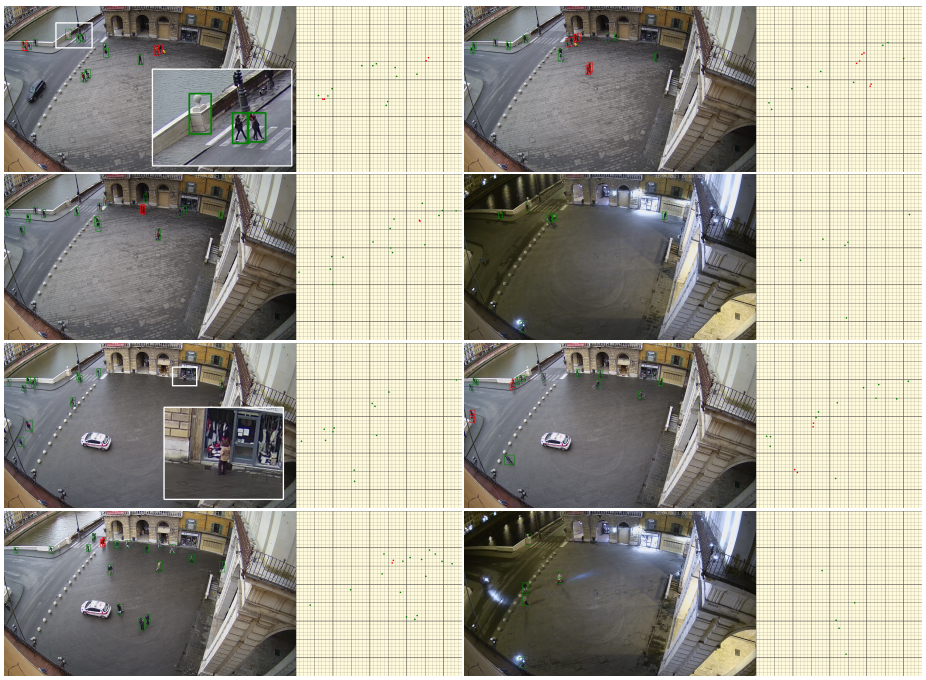

Figure 10: Examples of detections and distance warnings under different lighting conditions. Each of the eight images represents, on its left side, pedestrians detected and tracked in the example scenario, while showing on its right side their 2D projection on a virtual planar surface obtained through homography, with a reference 1-meter-spacing overlay grid. The green color means a safe placement, and the red indicates violations of the 1-meter physical distance rule. Some failure examples are outlined in white and zoomed.

6. Conclusion

In this work, we presented a modular framework based on Computer Vision and AI technologies, deployed in a real use-case scenario on a low-cost off-the-shelf embedded platform and aimed at monitoring human activities in critical conditions. Our goal was to provide a system having the peculiarity to be *expandable* in the future, simply adding new modules in charge of performing new functionalities that the user can easily enable (or disable) according to their needs. In this way, our framework turns out to be *flexible* since the tasks to be accomplished can change in time, and it is always possible to insert a new module responsible for performing it. As an effective setup, we implemented a set of visual-based modules for pedestrian detection, tracking, aggregation counting based on instances and density maps, social distancing calculations, and personal protection environment detection. Specifically, we trained artificial neural models with publicly available and, for the purpose of the physical device installation, custom datasets; at the same time, we applied a transfer learning approach to expand detection capabilities by using computer-generated training imagery. To test the effectiveness of our solution, we monitored a known place in Italy during the restrictions imposed from the COVID-19 pandemic, proving satisfactory accuracy in terms of detection, counting, and physical distance measurements.

6.1. Future Work

In the next iteration of this project, we plan to develop an algorithm that can automatically select the best counting modality between instancing and density map, which is currently chosen manually by the user. At the same time, we will try to integrate and expand modules with further visual analyses, like gesture/posture recognition and the assessment of appropriate PPE wearing. Finally, we will attempt to apply a transfer learning approach to predict physical distances among people by using an automatically labeled computer-generated training set based on a rendering engine simulation.

Acknowledgements

This work was partially supported by the H2020 projects AI4EU (GA 825619) and AI4Media (GA 951911), and by the Tuscany POR FSE 2014-2020 project AI-MAP (CNR4C program, CUP B15J19001040004). We would like to thank Visual Engines S.r.l. - https://www.visualengines.com - for supporting this research and giving us access to their data sources collected for the CrowdVisor project.

References

Ahmed, I., Ahmad, M., & Jeon, G. (2021a). Social distance monitoring framework using deep learning architecture to control infection transmission of COVID-19 pandemic. *Sustainable Cities and Society*, *69*, 102777. URL: https://doi.org/10.1016%2Fj.scs.2021.102777. doi:10.1016/j.scs.2021.102777.

Ahmed, I., Ahmad, M., Rodrigues, J. J., Jeon, G., & Din, S. (2021b). A deep learning-based social distance monitoring framework for COVID-19. *Sustainable Cities and Society*, *65*, 102571. URL: https://doi.org/10.1016%2Fj.scs.2020.102571. doi:10.1016/j.scs.2020.102571.

Amato, G., Ciampi, L., Falchi, F., Gennaro, C., & Messina, N. (2019). Learning pedestrian detection from virtual worlds. In E. Ricci, S. R. Bulò, C. Snoek, O. Lanz, S. Messelodi, & N. Sebe (Eds.), *Image Analysis and Processing - ICIAP 2019 - 20th International Conference, Trento, Italy, September 9-13, 2019, Proceedings, Part I* (pp. 302–312). Springer volume 11751 of *Lecture Notes in Computer Science*. URL: https://doi.org/10.1007/978-3-030-30642-7_27. doi:10.1007/978-3-030-30642-7_27.

Benfold, B., & Reid, I. (2011). Stable multi-target tracking in real-time surveillance video. In *CVPR 2011*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2011.5995667. doi:10.1109/cvpr.2011.5995667.

Bewley, A., Ge, Z., Ott, L., Ramos, F., & Upcroft, B. (2016). Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE. URL: https://doi.org/10.1109%2Ficip.2016.7533003. doi:10.1109/icip.2016.7533003.

Bochkovskiy, A., Wang, C., & Liao, H. M. (2020). Yolov4: Optimal speed and accuracy of object detection. *CoRR*, *abs/2004.10934*. URL: https://arxiv.org/abs/2004.10934. arXiv:2004.10934.

Chen, L., Papandreou, G., Schroff, F., & Adam, H. (2017). Rethinking atrous convolution for semantic image segmentation. *CoRR*, *abs/1706.05587*. URL: http://arxiv.org/abs/1706.05587. arXiv:1706.05587.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2018). DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*, 834–848. URL: https://doi.org/10.1109%2Ftpami.2017.2699184. doi:10.1109/tpami.2017.2699184.

Ciampi, L., Messina, N., Falchi, F., Gennaro, C., & Amato, G. (2020). Virtual to real adaptation of pedestrian detectors. *Sensors*, *20*, 5250. URL: https://doi.org/10.3390%2Fs20185250. doi:10.3390/s20185250.

Dendorfer, P., Rezatofighi, S. H., Milan, A., Shi, J., Cremers, D., Reid, I. D., Roth, S., Schindler, K., & Leal-Taixé, L. (2019). CVPR19 tracking and detection challenge: How crowded can it get? *CoRR*, *abs/1906.04567*. URL: http://arxiv.org/abs/1906.04567. arXiv:1906.04567.

Di Benedetto, M., Carrara, F., Meloni, E., Amato, G., Falchi, F., & Gennaro, C. (2020). Learning accurate personal protective equipment detection from virtual worlds. *Multimedia Tools and Applications*, *80*, 23241–23253. URL: https://doi.org/10.1007%2Fs11042-020-09597-9. doi:10.1007/s11042-020-09597-9.

Eyiokur, F. I., Ekenel, H. K., & Waibel, A. H. (2021). A computer vision system to help prevent the transmission of COVID-19. *CoRR*, *abs/2103.08773*. URL: https://arxiv.org/abs/2103.08773. arXiv:2103.08773.

Fabbri, M., Lanzi, F., Calderara, S., Palazzi, A., Vezzani, R., & Cucchiara, R. (2018). Learning to detect and track visible and occluded body joints in a virtual world. In *Computer Vision – ECCV 2018* (pp. 450–466). Springer International Publishing. URL: https://doi.org/10.1007%2F978-3-030-01225-0_27. doi:10.1007/978-3-030-01225-0_27.

Fabbri, M., Lanzi, F., Gasparini, R., Calderara, S., Baraldi, L., & Cucchiara, R. (2020). Inter-homines: Distance-based risk estimation for human safety. *CoRR*, *abs/2007.10243*. URL: https://arxiv.org/abs/2007.10243. arXiv:2007.10243.

Fischler, M. A., & Bolles, R. C. (1987). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. In *Readings in Computer Vision* (pp. 726–740). Elsevier. URL: https://doi.org/10.1016%2Fb978-0-08-051581-6.50070-2. doi:10.1016/b978-0-08-051581-6.50070-2.

Girshick, R. (2015). Fast r-CNN. In *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE. URL: https://doi.org/10.1109%2Ficcv.2015.169. doi:10.1109/iccv.2015.169.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2016.90. doi:10.1109/cvpr.2016.90.

Idrees, H., Tayyab, M., Athrey, K., Zhang, D., Al-Maadeed, S., Rajpoot, N., & Shah, M. (2018). Composition loss for counting, density map estimation and localization in dense crowds. In *Computer Vision – ECCV 2018* (pp. 544–559). Springer International Publishing. URL: https://doi.org/10.1007%2F978-3-030-01216-8_33. doi:10.1007/978-3-030-01216-8_33.

Khandelwal, P., Khandelwal, A., Agarwal, S., Thomas, D., Xavier, N., & Raghuraman, A. (2020). Using computer vision to enhance safety of workforce in manufacturing in a post COVID world. *CoRR*, *abs/2005.05287*. URL: https://arxiv.org/abs/2005.05287. arXiv:2005.05287.

Kong, X., Wang, K., Wang, S., Wang, X., Jiang, X., Guo, Y., Shen, G., Chen, X., & Ni, Q. (2021). Real-time mask identification for COVID-19: An edge-computing-based deep learning framework. *IEEE Internet of Things Journal*, *8*, 15929–15938. URL: https://doi.org/10.1109%2Fjiot.2021.3051844. doi:10.1109/jiot.2021.3051844.

Lempitsky, V. S., & Zisserman, A. (2010). Learning to count objects in images. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, & A. Culotta (Eds.), *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada* (pp. 1324–1332). Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2010/hash/fe73f687e5bc5280214e0486b273a5f9-Abstract.html.

Li, Y., Zhang, X., & Chen, D. (2018). CSRNet: Dilated convolutional neural networks for understanding the highly congested scenes. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2018.00120. doi:10.1109/cvpr.2018.00120.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In *Computer Vision – ECCV 2014* (pp. 740–755). Springer International Publishing. URL: https://doi.org/10.1007%2F978-3-319-10602-1_48. doi:10.1007/978-3-319-10602-1_48.

Milan, A., Leal-Taixé, L., Reid, I. D., Roth, S., & Schindler, K. (2016). MOT16:

A benchmark for multi-object tracking. *CoRR*, *abs/1603.00831*. URL: http://arxiv.org/abs/1603.00831. arXiv:1603.00831.

N., M.-H. T. (2021). Applications of artificial intelligence in battling against covid-19: A literature review. *Chaos, Solitons & Fractals*, *142*, 110338. URL: https://doi.org/10.1016%2Fj.chaos.2020.110338. doi:10.1016/j.chaos.2020.110338.

Parent, R. (2012). Chapter 2 - technical background. In R. Parent (Ed.), *Computer Animation (Third Edition)* (pp. 33–60). Boston: Morgan Kaufmann. (Third edition ed.). URL: https://www.sciencedirect.com/science/article/pii/B9780124158429000022. doi:https://doi.org/10.1016/B978-0-12-415842-9.00002-2.

Punn, N. S., Sonbhadra, S. K., & Agarwal, S. (2020). Monitoring COVID-19 social distancing with person detection and tracking via fine-tuned YOLO v3 and deepsort techniques. *CoRR*, *abs/2005.01385*. URL: https://arxiv.org/abs/2005.01385. arXiv:2005.01385.

Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, faster, stronger. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2017.690. doi:10.1109/cvpr.2017.690.

Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. *CoRR*, *abs/1804.02767*. URL: http://arxiv.org/abs/1804.02767. arXiv:1804.02767.

Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster r-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*, 1137–1149. URL: https://doi.org/10.1109%2Ftpami.2016.2577031. doi:10.1109/tpami.2016.2577031.

Ros, G., Sellart, L., Materzynska, J., Vazquez, D., & Lopez, A. M. (2016a). The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2016.352. doi:10.1109/cvpr.2016.352.

Ros, G., Stent, S., Alcantarilla, P. F., & Watanabe, T. (2016b). Training constrained deconvolutional networks for road scene semantic segmentation. *CoRR*, *abs/1604.01545*. URL: http://arxiv.org/abs/1604.01545. arXiv:1604.01545.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2018.00474. doi:10.1109/cvpr.2018.00474.

Saponara, S., Elhanashi, A., & Gagliardi, A. (2021). Implementing a real-time, AI-based, people detection and social distancing measuring system for covid-19. *Journal of Real-Time Image Processing*, *18*, 1937–1947. URL: https://doi.org/10.1007%2Fs11554-021-01070-6. doi:10.1007/s11554-021-01070-6.

Shao, S., Zhao, Z., Li, B., Xiao, T., Yu, G., Zhang, X., & Sun, J. (2018). Crowdhuman: A benchmark for detecting human in a crowd. *CoRR*, *abs/1805.00123*. URL: http://arxiv.org/abs/1805.00123. arXiv:1805.00123.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In Y. Bengio, & Y. LeCun (Eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. URL: http://arxiv.org/abs/1409.1556.

Wang, Q., Gao, J., Lin, W., & Li, X. (2021). NWPU-crowd: A large-scale benchmark for crowd counting and localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*, 2141–2149. URL: https://doi.org/10.1109%2Ftpami.2020.3013269. doi:10.1109/tpami.2020.3013269.

Wang, Q., Gao, J., Lin, W., & Yuan, Y. (2019). Learning from synthetic data for crowd counting in the wild. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2019.00839. doi:10.1109/cvpr.2019.00839.

Wang, Z., Bovik, A., Sheikh, H., & Simoncelli, E. (2004). Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, *13*, 600–612. URL: https://doi.org/10.1109%2Ftip.2003.819861. doi:10.1109/tip.2003.819861.

Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. In *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE. URL: https://doi.org/10.1109%2Ficip.2017.8296962. doi:10.1109/icip.2017.8296962.

Xiao, T., Li, S., Wang, B., Lin, L., & Wang, X. (2017). Joint detection and identification feature learning for person search. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2017.360. doi:10.1109/cvpr.2017.360.

Yang, D., Yurtsever, E., Renganathan, V., Redmill, K. A., & Ümit Özgüner (2021). A vision-based social distancing and critical density detection system for COVID-19. *Sensors*, *21*, 4608. URL: https://doi.org/10.3390%2Fs21134608. doi:10.3390/s21134608.

Yu, F., & Koltun, V. (2016). Multi-scale context aggregation by dilated convolutions. In Y. Bengio, & Y. LeCun (Eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4,*

2016, Conference Track Proceedings. URL: http://arxiv.org/abs/1511.07122.

Zhang, S., Benenson, R., & Schiele, B. (2017a). CityPersons: A diverse dataset for pedestrian detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2017.474. doi:10.1109/cvpr.2017.474.

Zhang, S., Zhu, X., Lei, Z., Shi, H., Wang, X., & Li, S. Z. (2017b). FaceBoxes: A CPU real-time face detector with high accuracy. In *2017 IEEE International Joint Conference on Biometrics (IJCB)*. IEEE. URL: https://doi.org/10.1109%2Fbtas.2017.8272675. doi:10.1109/btas.2017.8272675.

Zhang, Y., Zhou, D., Chen, S., Gao, S., & Ma, Y. (2016). Single-image crowd counting via multi-column convolutional neural network. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2016.70. doi:10.1109/cvpr.2016.70.

Zheng, L., Zhang, H., Sun, S., Chandraker, M., Yang, Y., & Tian, Q. (2017). Person re-identification in the wild. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2017.357. doi:10.1109/cvpr.2017.357.

Zhou, X., Wang, D., & Krähenbühl, P. (2019). Objects as points. *CoRR*, *abs/1904.07850*. URL: http://arxiv.org/abs/1904.07850. arXiv:1904.07850.