Fuentes Jiménez, D., Losada Gutiérrez, C., Casillas Pérez, D., Macías Guarasa, J., Pizarro, D., Martín López, R. & Luna, C. 2021, "Towards dense people detection with deep learning and depth images", Engineering Applications of Artificial Intelligence, vol. 106, art. no. 104484, pp. 1-26.

Available at https://doi.org/10.1016/j.engappai.2021.104484

Towards Dense People Detection with Deep Learning and Depth images

David Fuentes-Jimenez[a,c], Cristina Losada-Gutierrez[a], David Casillas-Perez[a,b], Javier Macias-Guarasa[a], Daniel Pizarro[a], Roberto Martin-Lopez[a], Carlos A. Luna[a]

[a]*Department of Electronics. University of Alcalá, Ctra. Madrid-Barcelona, km. 33600, 28805. Alcalá de Henares. SPAIN*
E-mails: d.fuentes@edu.uah.es (D. Fuentes-Jimenez), cristina.losada@uah.es (C. Losada-Gutierrez), david.casillas@edu.uah.es (D. Casillas-Perez), roberto.martin@edu.uah.es (R. Martin-Lopez), javier.maciasguarasa@uah.es (J. Macias-Guarasa), daniel.pizarro@uah.es (D. Pizarro), carlos.luna@uah.es (C.A. Luna).
[b]*Department of Signal Processing and Communications, Universidad Rey Juan Carlos, 28943 Fuenlabrada, SPAIN*
E-mails: david.casillas@urjc.es (D. Casillas-Perez)
[c]*Corresponding author. Tel: +34 918856592, fax: +34 918856591.*

Abstract

This paper describes a novel DNN-based system, named `PD3net`, that detects multiple people from a single depth image, in real time. The proposed neural network processes a depth image and outputs a likelihood map in image coordinates, where each detection corresponds to a Gaussian-shaped local distribution, centered at each person's head. This likelihood map encodes both the number of detected people as well as their position in the image, from which the 3D position can be computed. The proposed DNN includes spatially separated convolutions to increase performance, and runs in real-time with low budget GPUs. We use synthetic data for initially training the network, followed by fine tuning with a small amount of real data. This allows adapting the network to different scenarios without needing large and manually labeled image datasets. Due to that, the people detection system presented in this paper has numerous potential applications in different fields, such as capacity control, automatic video-surveillance, people or groups behavior analysis, healthcare or monitoring and assistance of elderly people in ambient assisted living environments. In addition, the use of depth information does not allow recognizing the identity of people in the scene, thus enabling their detection while preserving their privacy. The proposed DNN has been experimentally evaluated and compared with other state-of-the-art approaches, including both classical and DNN-based solutions, under a wide range of experimental conditions. The achieved results allows concluding that the proposed architecture and the training strategy are effective, and the network generalize to work with scenes different from those used during training. We also demonstrate that our proposal outperforms existing methods and can accurately detect people in scenes with significant occlusions.

Keywords: People detection, Depth camera information, Interest regions estimation, Feature extraction, Deep learning, Convolutional Neural Networks

1. Introduction

People detection and localization from cameras have lately received a great deal of attention from the scientific community, due to their multiple applications in different areas, such as security, automatic video surveillance (Villamizar et al., 2018; Moro et al., 2018; Susperregi et al., 2013), analysis of people or groups behavior, healthcare (Lee et al., 2013; Wang et al., 2016; Gavriilidis et al., 2018) or elderly people assistance in ambient assisted living environments (Bektüzün et al., 2013; Barabas et al., 2019). However, it remains an open problem, and presents several challenging tasks (Zhang et al., 2016b, 2018), especially in crowded scenes.

Early people detection methods used RGB images, mainly captured from *frontal* viewpoint. Methods such as (Ramanan et al., 2006; Chen et al., 2010; Jeong et al., 2013; Bak et al., 2017; Aguilar et al., 2017) used traditional computer vision algorithms, such as appearance models in (Ramanan et al., 2006), or classical face detection in (Chen et al., 2010). These approaches achieved good results in controlled environments, but struggle with the presence of partial occlusions, motion blur, and low resolution images.

Deep Neural Networks (DNN) have greatly improved the state-of-the-art in several critical computer vision applications, such as object detection (Redmon et al., 2015; Han et al., 2020), semantic segmentation (Cao et al., 2017; Li et al., 2020; Ruiz-Santaquiteria et al., 2020), classification (Shin et al., 2016) or activity recognition (Hayashi et al., 2015; Zhang et al., 2015). Similarly, DNN-based people detection methods using RGB data (Bochkovskiy et al., 2020; Du et al., 2019; G.Ghiasi et al., 2019; Zhang et al., 2020) have considerably improved over the classical algorithms. However, these DNN-based methods have also significant drawbacks, such as the large amount of labeled data needed for training, and the requirement of dedicated processing units to run and train the network. Besides, recent DNN-based methods still present low accuracy in highly cluttered scenes (see Figure 1 for an example).

People detection using depth images is a less popular topic in the literature, mainly because depth cameras are not as widely available as RGB ones. Nonetheless, using depth images has significant advantages: *1)* Depth images naturally disambiguate objects at different depths, which may help to process occlusions in crowded scenes; *2)* depth information is less complex than RGB information as it is not affected by appearance or lighting changes; and *3)* once detected in the image, the three dimensional positions of people are directly available using depth information, which is a desirable feature in many applications.

There exist several depth-based people detection methods in the literature, and some of them also include RGB information. Recent DNN-based approaches obtain the best detection results in this category. However, they bear important limitations. Some methods are specific to *zenithal* viewpoints (Del Pizzo et al., 2016; Luna et al., 2017; Fuentes-Jimenez et al., 2019b; Zhou et al., 2017a), which reduce occlusions and make

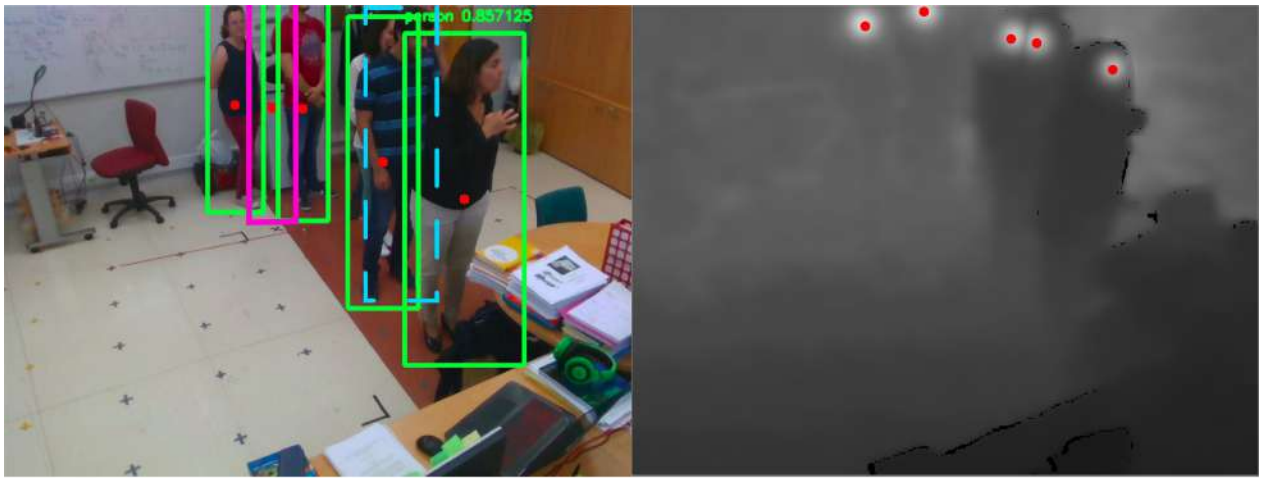


Figure 1: Performance example of our PD3net vs the DNN-based method YOLO16. The left image shows the people detection results obtained by YOLO16, where the green rectangles are its correct detections. The right image shows the results achieved by our PD3net, where the red dots are its correct detections. Note how YOLO16 generates a false positive (pink rectangle) as a microwave oven on top of a small fridge at the room back is detected as a person, and a false negative (cyan dotted rectangle) as the woman in a white shirt is partially occluded by the man in a blue and black striped shirt. On the contrary, our system is able to correctly detect all people in that scene with a strong occlusion.

the problem less ambiguous, with the drawback of limiting the camera field-of-view. Other strategies use a conventional *frontal* viewpoint (Girshick et al., 2013; Zhou et al., 2017a), which covers a wider range of applications. Girshick et al. (2013) and Zhou et al. (2017a) use a region proposal method to detect candidates, and a classifier to select the positive regions that correspond to a person. This strategy is not optimal, especially for densely populated scenes, and it is not efficient, as its complexity depends on the number of possible candidates detected in the image.

This paper proposes a new DNN-based approach, that we call PD3net, for detecting multiple people from a single depth image acquired using a camera in an elevated frontal position, as seen in Figure 1. The main contributions of the proposal are listed below:

- The network architecture is fully convolutional and very efficient by using spatially separable convolutions, so that it runs in real-time with low-cost GPU and CPU architectures.
- The neural network is initially trained end-to-end with synthetically generated depth images. After this initial training we fine-tune the network with a small number of annotated real images. Our experimental evaluation shows that this strategy leads to accurate detectors that generalize well in general scenes.
- The neural network recovers a dense likelihood map that effectively detects multiple people in crowded

scenes (see Figure 1).

- The experimental results outperform those of existing state-of-the-art proposals, including both classical and DNN-based approaches.
- The method works with different cameras and depth sensing technologies.
- The method does not have any restriction in the maximum number of detections per image.

The rest of the paper is structured as follows: Section 2 reviews the latest state-of-the-art methods focused on the field of people detection. Section 3 explains in detail our DNN-based proposal, describing its architecture and the training procedure. Section 4 shows the experimental setup developed to evaluate our approach. This section includes a thorough comparison with the main state-of-the-art methods over a wide range of publicly available datasets. Finally, Section 5 describes the main conclusions.

2. Previous works

People detection methods are classified in this section according to three main criteria: *a)* the type of information used, distinguishing between RGB, depth and its combination RGB+D; *b)* the type of algorithm used, distinguishing between classical and DNN-based strategies; and finally *c)* the camera's point of view. Table 1 shows the most relevant people detection methods in the state-of-the-art and their corresponding classification, according to the three criteria.

2.1. RGB methods

Classical approaches for people detection (Ramanan et al., 2006; Chen et al., 2010; Jeong et al., 2013; Bak et al., 2017; Aguilar et al., 2017) use conventional RGB images as input, usually taken with a *frontal* camera. Within these approaches, Ramanan et al. (2006) proposes a method based on appearance models, whereas Jeong et al. (2013) suggests an approach for people detection using interest point classification. Other alternatives for RGB people detection are based in face detection (Chen et al., 2010), image descriptors based on Brownian motion statistics (Bak et al., 2017) or HAAR-LBP and HOG cascade classifiers combined with Saliency Maps (Aguilar et al., 2017). Recently, we can find several RGB people detection methods based on DNNs, such as (Bochkovskiy et al., 2020; Du et al., 2019; G.Ghiasi et al., 2019; Wang & Zhao, 2017; Zhao et al., 2017; Tian et al., 2015). They outperform classical methods but still struggle in populated scenes with occlusions.

Table 1: Classification of the main people detector methods in the state-of-the-art. The *Alg* column shows the algorithmic strategy used (C=Classical, D=DNN). The *Input* column shows the input information (R=RGB, D=Depth, R+D=RGBD). The *View.* column show the camera viewpoint (F=Frontal, Z=Zenithal).

Reference	Alg.	Input	View	Description
Ramanan et al. (2006)	C	R	F	Tracking by model-building and detection
Chan et al. (2008)	C	R	F	Privacy-preserving system based on a mixture of dynamic textures motion model
Chen et al. (2010)	C	R	F	People counting system based on face detection
Dan et al. (2012)	C	R+D	Z	RGB and Depth fusion for people detection
Zhang et al. (2012)	C	D	Z	Unsupervised people counting via vertical Kinect Sensor
Stahlschmidt et al. (2013, 2014)	C	D	Z	Differences from the ground plane are used to develop regions of interest
Jeong et al. (2013)	C	R	F	People counting based in statical and moving detection points.
Zhu & Wong (2013)	C	R+D	Z	Adaboost algorithm built from weak classifiers for detecting people
Galăík & Gargalík (2013)	C	D	F	Real-Time People Detector with minimum-weighted bipartite graph matching
Tian et al. (2015)	D	R	Z	Optimization of pedestrian detection with semantic tasks
Liu et al. (2015)	C	R+D	Z	People detection taking different poses in cluttered and dynamic environments
Del Pizzo et al. (2016)	C	R+D	Z	Depth-RGB and both people detector with crossing-path points
Vera et al. (2016)	D	D	-	Cooperating network for people detection
Bak et al. (2017)	C	R	F	Brownian covariance descriptor
Aguilar et al. (2017)	C	R	F	Cascade classifier with salience map for pedestrian detection
Wang & Zhao (2017)	D	R	Z	Multi-layer regional-based convolutional for crowded scenes
Zhao et al. (2017)	D	R	F	Real-time detection based on physical radius-depth detector
Luna et al. (2017)	C	D	Z	ToF people detector based on depth information
Ren et al. (2017)	C	R+D	Z	Parallel deep feature extraction from RGB and Depth simultaneously
Zhou et al. (2017a)	C	R+D	Z	Depth-encoding scheme which enhances the information for classification
Hu et al. (2018)	C	R+D	Z	Uses the 3D Mean-Shift with depth constraints to multi-person detection
Verma et al. (2019)	C	R	Z	Fuzzy-based detector based on CCA components
Fuentes-Jimenez et al. (2019b)	D	D	Z	Encoder-decoder DNN blocks with refinement.

2.2. *Depth and RGB-D methods*

As mentioned above, using depth images has important advantages over RGB information when applied in people detection tasks, in terms of information complexity and ambiguity reduction. We also include in this category RGB-D methods that jointly use RGB and Depth information, as RGB cameras are usually included with commercial depth sensors. We distinguish here between methods that use a *zenithal* camera viewpoint and those using a *frontal* camera viewpoint.

We find both classical and DNN-based approaches that use a *zenithal* depth camera. These methods are highly accurate regardless of the approach, due to the advantages given by the camera viewpoint, that virtually eliminates occlusions in indoor scenarios. This category includes depth-only methods (Luna et al., 2017; Fuentes-Jimenez et al., 2019b; Zhang et al., 2012; Stahlschmidt et al., 2013, 2014), and RGB-D approaches (Dan et al., 2012; Del Pizzo et al., 2016; Liu et al., 2015; Zhou et al., 2017a; Ren et al., 2017; Hu et al., 2018). In this context, RGB-D methods do achieve better results than depth-only methods. Among the classical depth-based methods, the proposal by Zhang et al. (2012) is based on a maximum detector followed by a water-filling algorithm, while Stahlschmidt et al. (2013, 2014) filter depth images using the normalized Mexican Hat Wavelet. Both proposals reduce their detection rate when people are very close to each other, or when they cross their paths. Besides, false positives appear if there are body parts different from the head (such as hands) closer to the camera. To address these drawbacks, several proposals include a classification stage to discriminate people from other elements in the scene (Galčík & Gargalík, 2013; Vera et al., 2016; Zhu & Wong, 2013; Luna et al., 2017), thus reducing these false positives. DNN-based methods (Wang & Zhao, 2017; Fuentes-Jimenez et al., 2019b) are more accurate than the previous ones, and significantly reduce the amount of false positives. In particular, the strategy by Fuentes-Jimenez et al. (2019b) achieves a high degree of generalization, being accurate with data captured from different sensors and environments, and without the need of retraining.

The zenithal viewpoint greatly reduces the effect of occlusions, but brings some drawbacks, such as the reduction of the field of view, which is limited by the camera height. In addition to this, most applications mainly use frontal camera viewpoints, such as in video-surveillance scenarios. However, in these configurations, the performance decreases due to occlusions between people and the detection task becomes more complex, requiring more sophisticated algorithms capable of dealing with occlusions. The state-of-the-art in this category is dominated by DNN-based methods (Girshick et al., 2013; Zhou et al., 2017a), mainly due to their higher learning capacity and their robustness to occlusions. These methods improve the detection results achieved by classical algorithms (Galčík & Gargalík, 2013). However, both (Girshick et al., 2013) and (Zhou et al., 2017a) are based on region proposal methods and classifiers. These strategies make them

not efficient as they are not robust enough to occlusions in crowded scenes, generating a high amount of false negatives.

Finally, we want to point out that in most of the people detection proposals shown in Table 1, the experimentation has been done on limited or non-public datasets. Only 10 of the 23 papers shown in the table, use publicly available datasets that are referenced in the papers themselves (most of them being composed by RGB images for people detection). The rest of the papers evaluate their methods by means of video sequences recorded by the authors in controlled experiments, with no citation nor public availability, thus limiting the possibility of reproducing their findings

Among the proposals using publicly available data, the approach by Zhang et al. (2012) uses the water filling datasets detailed in the same paper; Liu et al. (2015) use the RGB images of poses recorded by Choi et al. (2012); and Vera et al. (2016) use the dataset in (Wang et al., 2014). Bak et al. (2017) carry out experiments in the RGB dataset recorded by both Dalal & Triggs (2005) and Enzweiler et al. (2010); while Wang & Zhao (2017) use the RGB images taken in Stewart et al. (2016); and Zhao et al. (2017) use the RGB datasets detailed in (Zhang et al., 2016a; Choi et al., 2012). Zhou et al. (2017a) make a comparison with the three datasets described in (Tao et al., 2015), (Silberman et al., 2012), and (Spinello & Arras, 2011). Finally, Fuentes-Jimenez et al. (2019b) and Luna et al. (2017) develop their experiments in the GOTPD1 dataset (Fuentes-Jimenez et al., 2019b).

3. PD3net People Detector

3.1. Problem Formulation

This paper proposes PD3net, a CNN-based multiple people detector from depth images. PD3net receives a depth image as input and produces a likelihood map as output, with the same size as the input image, see Figure 2, and where each detection corresponds to a Gaussian-shaped local distribution, centered at the person's head position in the image.

3.2. Architecture of the PD3net Network

Figure 3 shows the PD3net architecture, which is inspired in the architecture initially proposed by Fuentes-Jimenez et al. (2019b). It is composed of two blocks: *the main block (MB)* and *the hypothesis reinforcement block (HRB)*. Both blocks are built using encoder-decoder architectures, which are common in other computer vision tasks, such as semantic segmentation (Badrinarayanan et al., 2015; Romera et al., 2017), 3D reconstruction, registration (Fuentes-Jiménez et al., 2018; Golyanik et al., 2018), or deep-fake detection (Guera

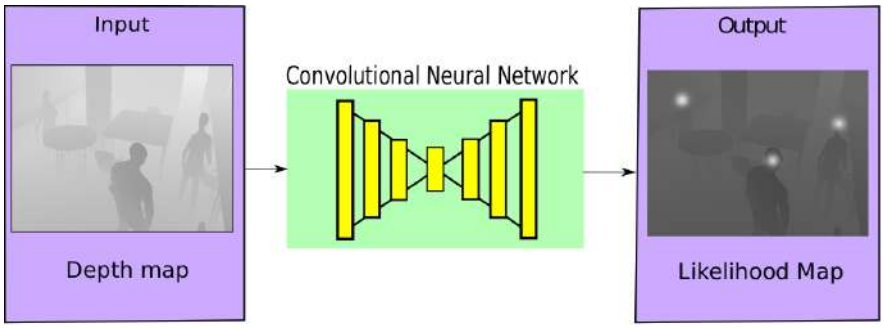


Figure 2: Depth input image and likelihood output map.

.

& Delp, 2018; Kim et al., 2018). The PD3net structure is based in a non-conventional convolutional neural network approximation as it does not use classical CNN schemes.

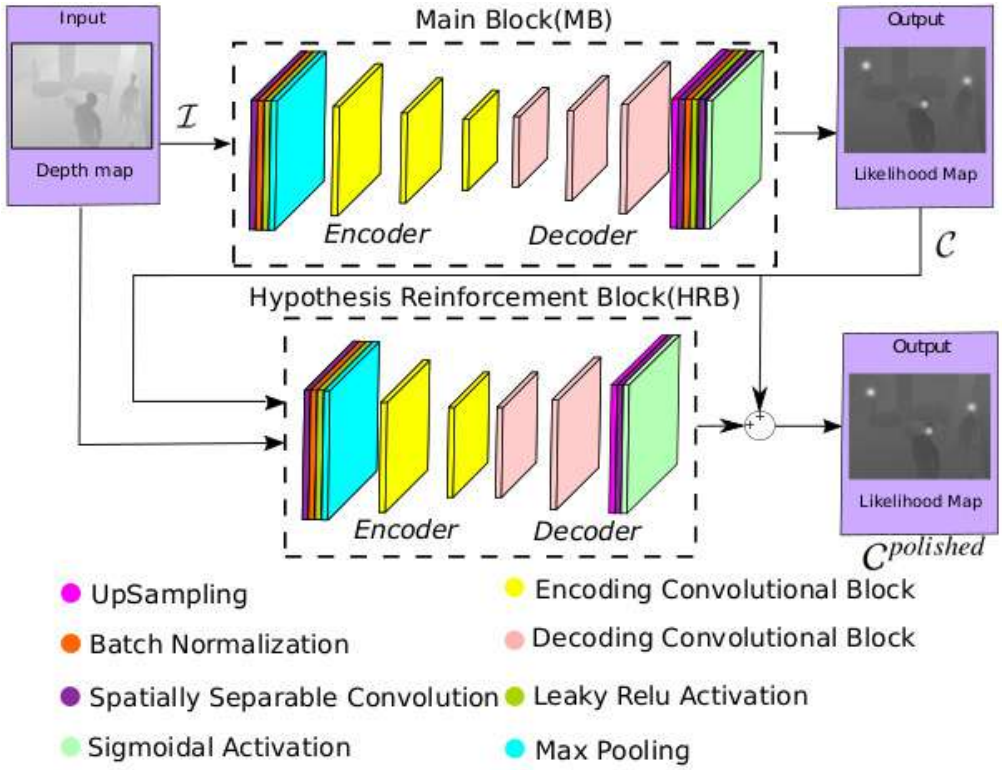
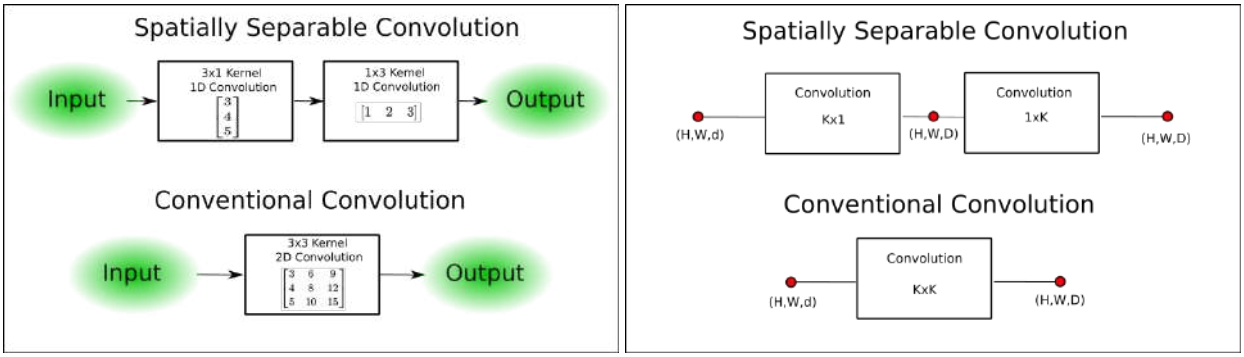


Figure 3: PD3net Architecture.

The input depth image $\mathcal{I}$, with a size of 240×320 in our case, is processed by the MB, which generates the initial likelihood map C (also with a size of 240×320), which is then concatenated with the input image $\mathcal{I}$, so that the input tensor of the HRB ($\mathcal{I}_2$) is composed by the concatenation of $\mathcal{I}$ and C, and has a size of $240 \times 320 \times 2$, as seen in Figure 3. The HRB output is added to C to produce the final likelihood map $C^{polished}$

(240×320) that contains the refined detections.



(a) Example comparison.

(b) General formulation

Figure 4: Differences between conventional and spatially separable convolutions.

The `PD3net` architecture is optimized to be fast and efficient in terms of parameters and operations (more details on its computational complexity can be found in Section 4.5). It employs the following elements:

- **Spatially Separable Convolutional Layers**: These layers increase the processing speed in comparison with regular convolutional layers, and are included in popular architectures, such as the *Inception V3* (Szegedy et al., 2015). Spatially separable convolutions use 1D filters to compose a "2D equivalent" convolution (an example can be seen in Figure 4a). However, spatially separable convolutions reduce the number of operations and parameters only under certain conditions that we discuss next.

  The number of parameters $nparam_{conv}$ and mathematical operations $nops_{conv}$ of a regular 2D convolutional layer are given by:

$$\begin{cases} nparam_{conv} = KKdD = K^2dD \\ nops_{conv} = HWD(K^2d + (K^2 - 1)d) = HWD((2K^2 - 1)d) \end{cases} \tag{1}$$

  where K is the kernel size, d is the input depth size, D is the output depth size, H is the input height, and W is the input width (see Figure 4b).

  In spatially separable convolutions composed of two 1D filters of $1 \times K$ and $K \times 1$, the number of parameters $nparam_{sep}$ and mathematical operations $nops_{sep}$ are:

$$\begin{cases} nparam_{sep} = KdD + KD^2 \\ nops_{sep} = HWD(Kd + (K - 1)d) + HWD(KD + (K - 1)D) = HWD((2K - 1)(d + D). \end{cases} \tag{2}$$

  Using equations (1) and (2), the following inequalities give the necessary conditions for reducing the number of operations and parameters when using spatially separable convolutions instead of conven-

tional convolutions:

$$nparam_{conv} > nparam_{sep} \;\Rightarrow\; d > \frac{1}{(K-1)}D$$
$$nops_{conv} > nops_{sep} \;\Rightarrow\; d > \frac{(2K-1)}{2K(K-1)}D \tag{3}$$

where we suppose a stride of 1 and a constant size of the image compensated with the padding. We apply the conditions of equation (3) to the three specific cases used in our neural network architecture, where $K \in \{3, 5, 7\}$. Table 2 shows, for each case, the conditions under which a spatially separable convolution improves a conventional convolution in terms of parameters and operations, and also states the most restrictive condition.

Improvement in	$K = 3$	$K = 5$	$K = 7$
Parameters $d > \frac{1}{(K-1)}D$	$d > 0.5D$	$d > 0.25D$	$d > 0.16D$
Operations $d > \frac{(2K-1)}{2K(K-1)}D$	$d > 0.416D$	$d > 0.225D$	$d > 0.154D$
Most restrictive	$d > 0.5D$	$d > 0.25D$	$d > 0.16D$

Table 2: Conditions that the use of factorized convolutions must meet to be faster and more parameter efficient than conventional convolutions for the proposed CNN.

- **Leaky ReLU activations:** We use *Leaky ReLU* activations to improve the generalization capabilities of the network and the training convergence, as proposed by Xu et al. (2015). *Leaky ReLU* activations help to solve the "dying ReLU" problem, where the zero activation zone of the ReLU slows down and destabilizes the training process. Instead of the zero activation zone, the *Leaky ReLU* has a small negative slope that mitigates this problem.

- **Residual blocks:** We use residual blocks with a structure similar to that used by He et al. (2016). The main difference is the inclusion of the spatially separable convolutions inside the residual blocks. We define two basic residual blocks: the *Encoding Convolutional Block* (`ECBs`), and the *Decoding Convolutional Block* (`DCBs`). The `ECBs` and `DCBs` blocks are similar in structure, as shown in Figure 5. They are formed by two unbalanced branches, where the first one has three convolutional layers, and the second one has one convolutional layer. The output of the blocks is composed of the added normalized output of the two branches. The main difference between the `ECBs` and `DCBs` lies in the convolutional part: the `ECB` uses a *factorized* convolution, and the `DCB` uses a *resized* convolution. In Figure 5 the parameters a, b and c define the number of filters in the three-layered branches. The number of filters of the third convolution in the bottom section must be equal to the number of filters in the top section (c parameter).

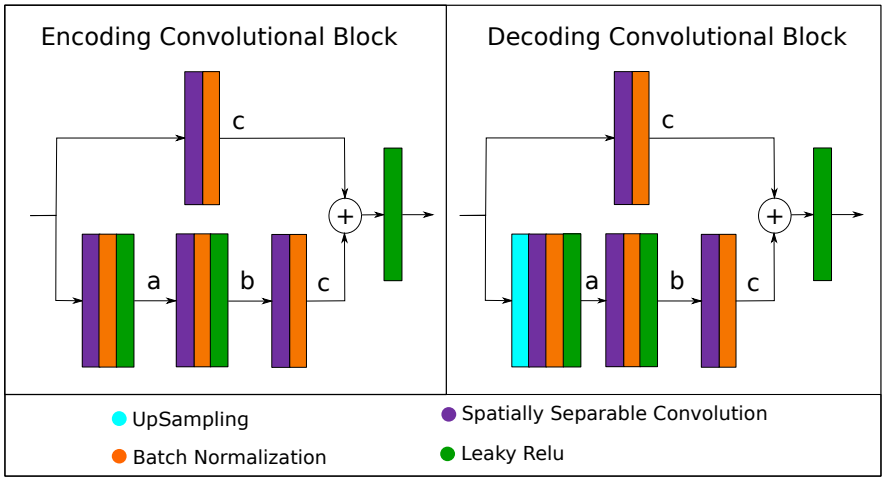


Figure 5: Architecture of the ECBs and DCBs .

- **Resized convolutions:** We use the resized convolution as an approximation for the de-convolution with nearest neighbor interpolation. This is done to avoid the possible checkerboard artifacts that could appear in our output likelihood maps if we use the transposed convolution approximation, as discussed in (Odena et al., 2016). Additionally, we prefer the nearest neighbor interpolation instead of bilinear or bicubic interpolation, because they led to problems in the interpolation of high-frequency image features, as also explained by Odena et al. (2016).

It is worth noting that, although the PD3net encoder-decoder structure is similar to that of Fuentes-Jimenez et al. (2019b), there are important differences that lead to relevant improvement in its performance and efficiency. In particular, the use of Spatially Separable Convolutional Layers increases the processing speed while reducing the number of operations and parameters. Moreover, PD3net includes Leaky ReLU activations to improve the generalization capability of the network, which facilitates adaptation to different contexts after training using synthetic data. Finally, the PD3net proposal uses Resized Convolutions as an approximation for the deconvolution with nearest neighbor interpolation, improving the interpolation for high frequency components.

3.2.1. Main Block (MB) Architecture

Table 3 describes the *Main Block* (MB) architecture, showing the parameters and dimensions of each layer. The main block follows an encoder-decoder neural structure. The encoder is made up of one regular convolutional layer (*i.e.* composed of a 2D convolution, Batch Normalization, Leaky ReLU activation, and a max pooling layer), followed by three ECB blocks. The decoder is composed of three DCB blocks, followed by one de-convolution (upsampling + convolutional layer) and one convolutional layer for adapting the output

Table 3: Detailed architecture of the main block.

Main Block (MB)		
Layer	**Output size**	**Parameters**
Input	$240 \times 320 \times 1$	-
Convolution	$120 \times 160 \times 64$	kernel=(7, 7) / strides=(2, 2)
BN	-	
Activation	Leaky ReLU	
Max Pooling	$40 \times 53 \times 64$	size=(3, 3)
ECB	$40 \times 53 \times 256$	kernel=(3, 3) / strides=(1, 1) (a=64, b=64, c=256)
ECB	$20 \times 27 \times 512$	kernel=(3, 3) / strides=(2, 2) (a=128, b=128, c=512)
ECB	$10 \times 14 \times 1024$	kernel=(3, 3) / strides=(2, 2) (a=256, b=256, c=1024)
DCB	$10 \times 14 \times 256$	kernel=(3, 3) / strides=(1, 1) (a=1024, b=1024, c=256)
DCB	$20 \times 28 \times 128$	kernel=(3, 3) / strides=(2, 2) (a=512, b=512, c=128)
DCB	$40 \times 56 \times 64$	kernel=(3, 3) / strides=(2, 2) (a=256, b=256, c=64)
Cropping	$40 \times 54 \times 64$	cropping=[(0, 0) (1, 1)]
Up Sampling	$120 \times 162 \times 64$	size=(3, 3)
Convolution	$240 \times 324 \times 64$	kernel=(7, 7) / strides=(2, 2)
Cropping	$240 \times 320 \times 64$	cropping=[(0, 0) (2, 2)]
BN	-	
Activation	Leaky ReLU	
Convolution	$240 \times 320 \times 1$	kernel=(3, 3) / strides=(1, 1)
Activation	Sigmoidal	
Output	$240 \times 320 \times 1$	-

depth. As opposed to the encoder, the decoder consists of DCB blocks that increase the data size and decreases depth, to obtain an output with the same size as the $\mathcal{I}$ input depth image.

Regarding the use of spatially separable convolutions in the MB layers, the first layer is not spatially separable since it would not reduce its computational complexity. In fact, introducing a spatially separable convolution in the first layer would imply a number of operations 8.71 times higher than a conventional convolution. As we can see in the conditions previously shown in Table 2, with a kernel $K = 7$ the input depth d would have to be at least 0.16 times the output depth D to lead to an improvement in terms of parameters and operations, with respect to a traditional convolution operation. The number of required operations in this case for a traditional convolution is 97, in contrast to the separable one that requires 845, which finally derives in 8.71 times the operations of a conventional convolutional layer.

3.2.2. Hypothesis Reinforcement Block (HRB) Architecture

The Hypothesis Reinforcement Block (HRB) is a reduced version of the MB. It receives $\mathcal{I}_2$ as input, which is composed by the concatenation of $\mathcal{I}$ and C as described above, with a size of $240 \times 320 \times 2$. The output of the HRB is added to C, the MB output, to obtain the refined likelihood map $C^{polished}$. The HRB follows the same structure as the MB and uses an encoder-decoder architecture. It uses two ECB and two DCB blocks for the encoder and decoder sections, respectively. Table 4 summarizes all the layers and blocks used in the HRB.

3.3. Training procedure

The network training process consists of two stages. In stage 1, we use a photo realistic synthetic and publicly available database named GESDPD (Fuentes-Jimenez et al., 2019a). It was created using the Blender graphics simulator (Blender Online Community, 2018) to initially train our network end-to-end.

The GESDPD generation procedure started by defining a 3D environment representing a rectangular room with some generic elements (round and rectangular tables, and a cylindrical column). The room dimensions are $8.56m \times 5.02m$, and it has a height of $3.84m$. A top view of the room with three sample users is shown in Figure 6.

GESDPD is composed of 22000 depth and RGB synthetic images, that simulate to have been taken within the artificial room with a depth and RGB cameras located in an elevated front position, at a height of 3 meters.

GESDPD labeled maps are manually generated with the same dimensions as the input images (320×240). For labeling the people positions, we have placed Gaussian functions around the centroid of the head of each

Table 4: Detailed architecture of the hypothesis reinforcement block.

Hypothesis Reinforcement Block (HRB)		
Layer	**Output size**	**Parameters**
Input	$240 \times 320 \times 2$	-
Convolution	$120 \times 160 \times 64$	kernel=(7, 7) / strides=(2, 2)
BN	-	
Activation	Leaky ReLU	
Max Pooling	$40 \times 53 \times 64$	size=(3, 3)
ECB	$40 \times 53 \times 256$	kernel=(3, 3) / strides=(1, 1) (a=64, b=64, c=256)
ECB	$20 \times 27 \times 512$	kernel=(3, 3) / strides=(2, 2) (a=128, b=128, c=512)
DCB	$40 \times 54 \times 128$	kernel=(3, 3) / strides=(2, 2) (a=512, b=512, c=128)
DCB	$80 \times 108 \times 64$	kernel=(3, 3) / strides=(2, 2) (a=256, b=256, c=64)
Up Sampling	$240 \times 324 \times 64$	size=(3, 3)
Cropping	$240 \times 320 \times 64$	cropping=[(0, 0) (2, 2)]
Convolution	$240 \times 320 \times 64$	kernel=(3, 3) / strides=(1, 1)
BN	-	
Activation	Leaky ReLU	
Convolution	$240 \times 320 \times 1$	kernel=(3, 3) / strides=(1, 1)
Activation	Sigmoidal	
Output	$240 \times 320 \times 1$	-

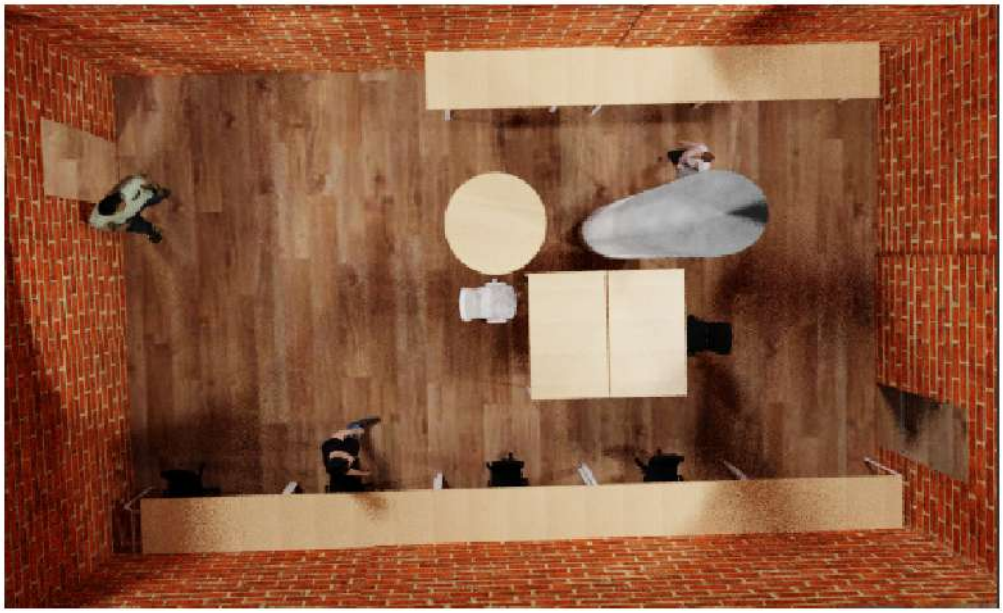


Figure 6: Virtual indoor environment used to create GESDPD.

person in the scene, so that the centroid corresponds to the 2D position of the center of the head and has a normalized value of one.

The synthetic images show a room with different persons walking in different directions, specifically 3 men and 1 woman. The set of labeled people is composed of 20800 samples. The camera image resolution is 320×240 and its perspective is not stationary, as it moves and rotates around the room along the database, which avoids having a constant background. This characteristic allows the network to learn the most important features for the person detection task, while using a dataset that is automatically generated and labeled. We split the synthetic database in training and validation subsets composed by the 67% and the 33% percent of the available data, respectively.

The two first rows of Figure 7 show the synthetically generated room. As described above, we randomly place virtual depth cameras so that each generated frame has a varying background. This strategy prevents the CNN from dealing with a constant background, thus allowing it to more easily generalize to different environments. An example of the synthetically generated depth images can be seen in the bottom row of Figure 7 using a color coded depth scheme.

In stage 2, we use a manually labeled real database composed by 3500 frames, recorded using a realsense D435 camera (Intel). We use this dataset to fine-tune and adapt the network weights to the real environment. This stage is necessary because the synthetic data does not include the image conditions that appear in a real scenario, such as motion blur, measurement noise or the influence of the ambient light in the depth

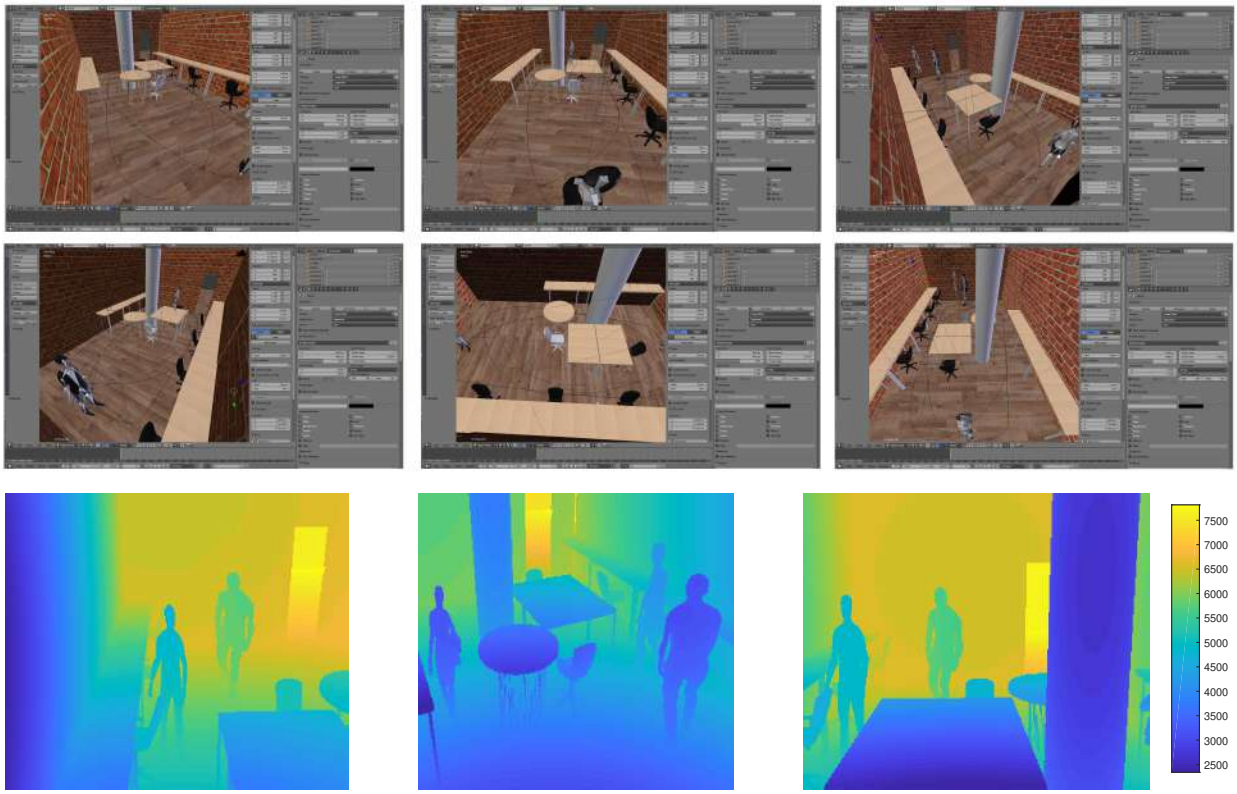


Figure 7: Sample images of the Blender simulated room with different perspectives (top two rows), and samples of synthetic depth images for training (belonging to the `GESDPD` dataset, bottom row).

measures. Regarding the labeling process, we again placed 2D Gaussian-like distributions centered at each person's head labeled positions. The Gaussian function is normalized, so that its maximum is equal to 1. The standard deviation of the Gaussian function is constant in all the cases, because using a variable standard deviation (*e.g.* varying it with the depth value) proved to decrease the detection performance in preliminary experiments. The standard deviation value has been calculated using the average estimate of a human head diameter, which is $\mathcal{D} = 15$ pixels, from which we set $\sigma = \mathcal{D}/2.5 = 6$. With multiple people, the overall likelihood map for each given point is computed assigning the maximum value among all the maps created for each individual person at the given point. Finally, we truncate the likelihood, making it equal to 0 for all values below the threshold $1/255$. This truncation plays an important role in the loss function. An example of a zoomed labeled image with the overlap of two labeled positions is showed in Figure 8.

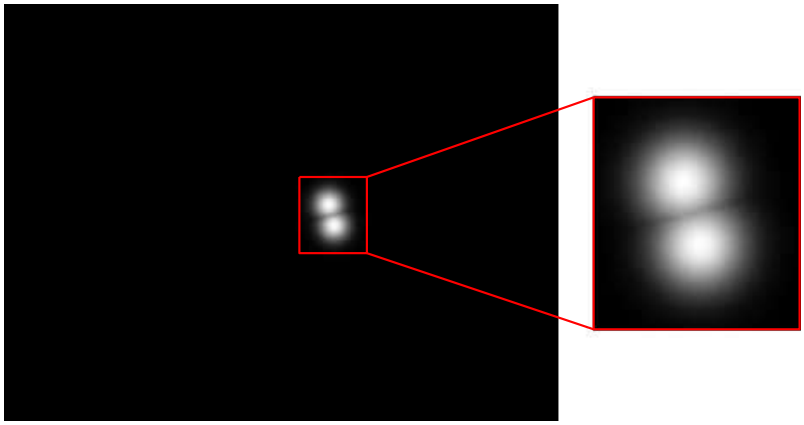


Figure 8: Gaussians Overlap Example.

With respect to the optimizer, we apply the *Swats* optimization strategy proposed by Keskar & Socher (2017). It involves a first training phase with the *Adam* solver (Kingma & Ba, 2014) corresponding to stage 1, and a second phase with *SGD+Momemtum* that corresponds to stage 2. This strategy improves the network generalization capabilities, as demonstrated in (Keskar & Socher, 2017). The initial *learning rate* of the first stage is 0.001, which is combined with the use of an early stopping callback to save the best possible model. In the second stage we use the *SGD+Momemtum* solver with a learning rate equal to 10^{-5}.

Our CNN uses a training set composed of input depth images $\mathcal{I}_i$ and ground truth likelihood maps $[C_i, C_i^{polished}]$, where $C_i = C_i^{polished}$. The proposed loss function minimizes the mean square error (MSE)

between the ground-truth likelihood maps $[C_i, C_i^{polished}]$ and their values estimated by the network, denoted by $[\hat{C}_i, \hat{C}_i^{polished}]$. We have to highlight that the proposed loss considers two groups of points in the likelihood maps. The first group corresponds to points where $C_i > 0$, and we mark them with the + subindex in any of the maps (*e.g.* C_{i_+} and $\hat{C}_{i_+}$). The second group corresponds to those points where $C_i = 0$, and we mark them with the subindex 0 (*e.g.* C_{i_0} and $\hat{C}_{i_0}$). An example of the $C_i = 0$ and $C_i > 0$ points considered can be seen in Figure 9.

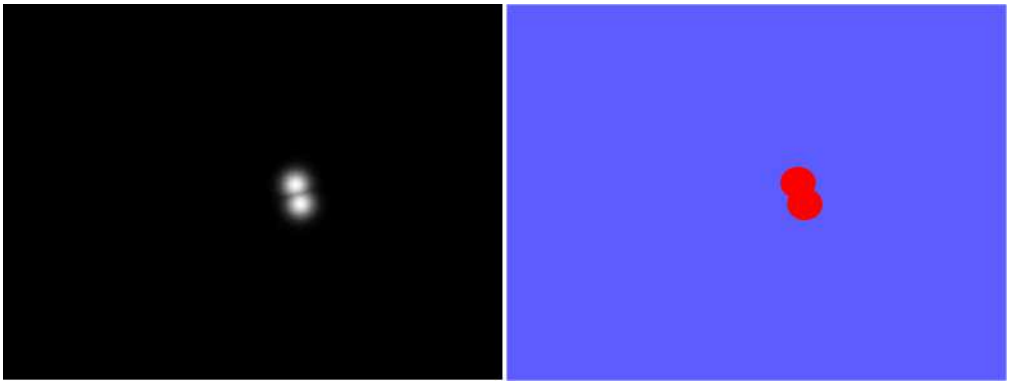


Figure 9: Example ground truth likelihood map of $C_i = 0$ and $C_i > 0$ points. In the left image, we can see an output likelihood map. In the right image we can observe the $C_i = 0$ and $C_i > 0$ points plotted in blue and red, respectively.

The proposed loss function is calculated as the addition of two terms, $\mathcal{L}_1$ and $\mathcal{L}_2$:

$$\mathcal{L} = \mathcal{L}_1 + \mathcal{L}_2, \tag{4}$$

where $\mathcal{L}_1$ corresponds to the loss at the `MB` output C, and $\mathcal{L}_2$ correspond to the loss at the `PD3net` output $C^{polished}$. The $\mathcal{L}_1$ term is the weighted sum of two MSE functions, involving the two points groups described above, denoted by the subindexes + and 0:

$$\mathcal{L}_1 = \lambda_1 \frac{1}{N} \sum_{i=1}^{N} \|\hat{C}_{i_+} - C_{i_+}\|^2 + \lambda_2 \frac{1}{N} \sum_{i=1}^{N} \|\hat{C}_{i_0} - C_{i_0}\|^2. \tag{5}$$

The hyperparameters λ_1 and λ_2 balance the relative importance of each term, and N is the batch size. The $\mathcal{L}_2$ term is similar $\mathcal{L}_1$ but involving the maps generated at the output, denoted by the superindex *polished*:

$$\mathcal{L}_2 = \lambda_1 \frac{1}{N} \sum_{i=1}^{N} \|\hat{C}_{i_+}^{polished} - C_{i_+}^{polished}\|^2 + \lambda_2 \frac{1}{N} \sum_{i=1}^{N} \|\hat{C}_{i_0}^{polished} - C_{i_0}^{polished}\|^2. \tag{6}$$

In our experiments, we empirically selected $\lambda_2 = 1$ and $\lambda_1 = 1.3$ after an initial experimental validation on independent data, for which we used the `GESDPD` validation subset. The term $\mathcal{L}_1$ forces the `MB` to produce the best possible initial map and $\mathcal{L}_2$ forces `HRB` to refine the `MB` output.

We train the network for 30 epochs with the *Adam* Optimizer in the first stage, and for 20 epochs with *SGD+Momemtum* in the second stage, choosing the best possible model obtained along the training process.

4. Experimental Work

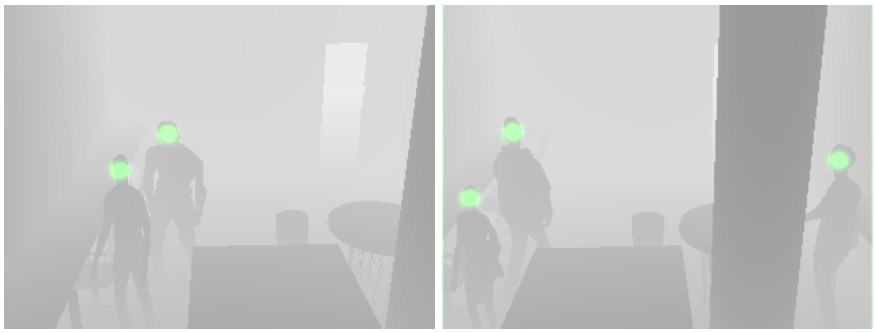
In this section we first describe the used datasets and their partition scheme (Subsection 4.1); and the thorough experimental setup developed (Subsection 4.2), including a description of the state-of-the-art algorithms we are comparing with, the evaluation metrics, and additional relevant issues to the experimental approach. After that, the results achieved and its comparison with a wide range of state-of-the-art proposals are included. The results are first presented for the synthetic data case in Section 4.3. This allows us to introduce and explain the effectiveness of a synthetic pre-training such as the one performed here, and which are the limitations that can be found in the real world. The results for each of the used realistic datasets are presented in Section 4.4, including a final comparison of the average results of all the databases, and a discussion about the use of a global training approach. Finally, the computational performance is briefly reviewed in Section 4.5.

4.1. Datasets & Data Partition

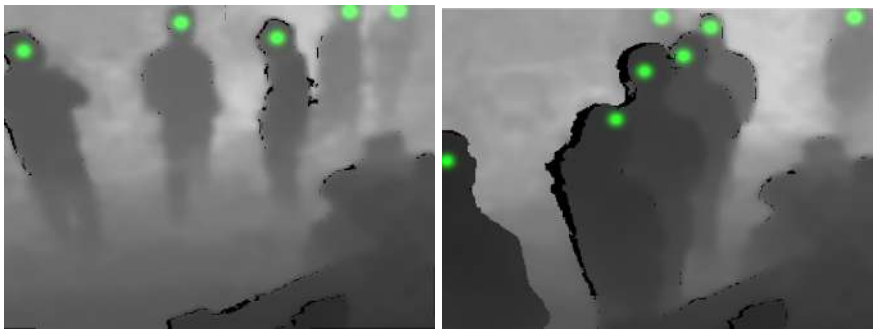
We have used five different databases which cover a wide range of the evaluation conditions presented in the state-of-the-art section. For each dataset, figures 10 11 and 12 provide some sample frames to give an idea on the style and quality of these different datasets, and their detailed description is included next:

1. **GESDPD**, described in (Fuentes-Jimenez et al., 2019a): The GEintra Synthetic Depth Person Detection Dataset (GESDPD) is a synthetic dataset which contains 22000 depth images, that simulate to have been taken with a sensor in an elevated front position, in an indoor working environment. As described in Section 3.3, these images have been generated using the Blender simulation software (Blender Online Community, 2018). The simulated scene shows a room with different people walking in different directions. The camera perspective is not stationary, as it rotates and moves along the dataset, which avoids a constant background that could be learned by CNN in the training, as it can be seen in Figure 7, which shows different perspectives of the synthetic room. Using different backgrounds around the synthetic room allows CNN to see the background as noise and focus the training in the people that come along the image, immunizing the network to the change of camera perspective and assembly conditions.

   The generated images have a resolution of 320×240 pixels codified as 16 bits unsigned integers. In the synthetic examples shown in Figure 7, the images correspond to three different perspectives, with the depth values represented using a color map. Additional samples of this dataset are included in Figure 10a using a grey level depth coding scheme, where the Gaussian-shaped ground truth information has been represented as green blobs.



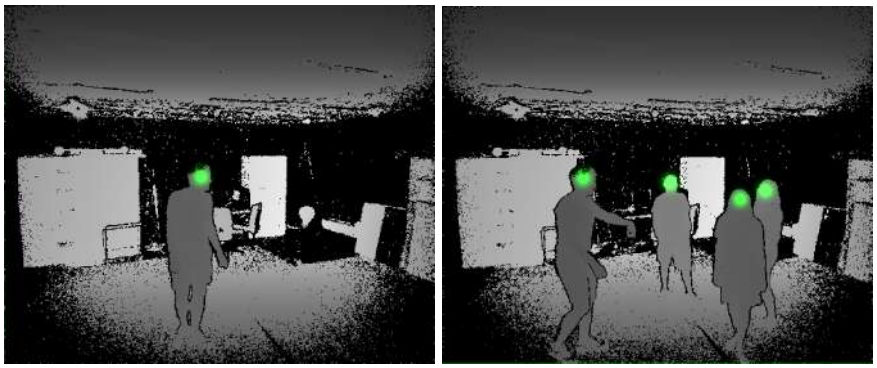
(a) Sample frames with labeled groundtruth from the GESDPD dataset (Fuentes-Jimenez et al., 2019a).



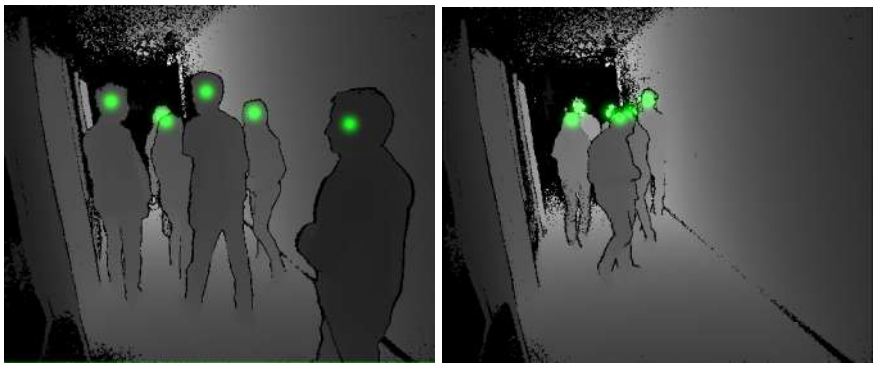
(b) Sample frames with labeled groundtruth from the GFPD dataset (Fuentes-Jimenez et al., 2020).

Figure 10: Sample frames from the GESDPD and GFPD (gray coded depth) with ground truth as green blobs.

2. **GFPD**, presented in (Fuentes-Jimenez et al., 2020): The Geintra Frontal Person Detection Dataset (`GFPD`) is a high-resolution dataset recorded with an realsense D435 camera Intel. The `GFPD` contains 5270 depth frames with 13827 annotated people instances. The camera has an active stereo depth sensor that provides depth maps with a resolution of 1280×720. The recordings of this dataset consider a great variety of conditions including different sensor heights (2200-2700 mm), different tilt angles (26-41 degrees), as well as different backgrounds and lighting conditions. Some samples of this dataset are shown in Figure 10b.
3. **EPFL**, presented in (Bagautdinov et al., 2015), that includes two different datasets:
   (a) **EPFL-LAB**: The first one (`EPFL-LAB`) contains around 1000 RGB-D frames with around 3000 annotated people instances. There are at most 4 people simultaneously present in the room. They are mostly facing the camera, presumably a scenario for which the Kinect software was fine-tuned. Some samples of this dataset can be visualized in Figure 11a.
   (b) **EPFL-CORRIDOR**: The second one (`EPFL-CORRIDOR`) was recorded in a more realistic environment, a corridor in a university building. It contains over 3000 frames with up to 8 individuals simultaneously present. It is a challenging dataset since there are important occlusions given the

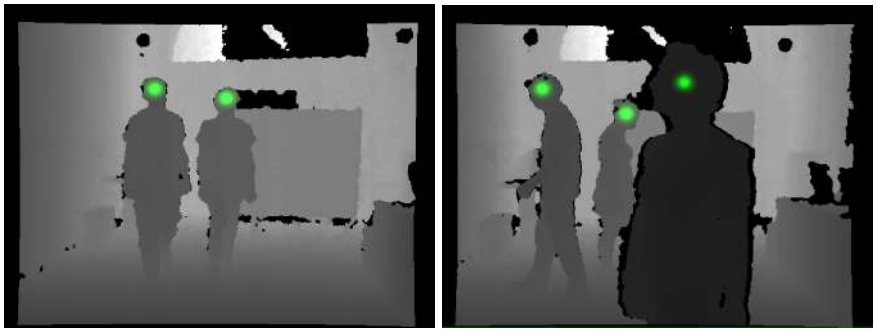


(a) Sample frames with labeled groundtruth from the EPFL-LAB dataset (Bagautdinov et al., 2015).



(b) Sample frames with labeled groundtruth from the EPFL-CORRIDOR dataset (Bagautdinov et al., 2015).

Figure 11: Sample frames from the EPFL-LAB, EPFL-CORRIDOR databases (gray coded depth).



(a) Sample frames with labeled groundtruth from the KTP dataset (Munaro & Menegatti, 2014).

(b) Sample frames with labeled groundtruth from the UNIHALL dataset (Spinello & Arras, 2011).

Figure 12: Sample frames from the KTP and UNIHALL databases (gray coded depth) with ground truth as green blobs..

camera position at one end of a narrow corridor, as can be seen in the sample frames of Figure 11b.

4. **KTP**, described in (Munaro & Menegatti, 2014): The Kinect Tracking Precision dataset (`KTP`) contains several sequences of at most 5 people walking in a small lab environment. They were recorded by a depth camera mounted on a robot platform. Here we only uses the subset that was recorded while the camera was static. The authors provide ground truth locations of the individuals, both on the image plane, and on the ground plane. Unfortunately, the quality of the ground truth for the ground plane is limited, due to the poor quality registration of the depth sensor location in the environment. In order to fix this, we manually specified points corresponding to individuals on the depth maps, then projected them on the ground plane, and took the average to get a single point representing the location of each person. This approach introduced a small localization bias as we only observe the outer surface of the person, but any motion capture system would have similar issues. Some samples of this dataset can be visualized in Figure 12a.
5. **UNIHALL**, presented in (Spinello & Arras, 2011; Luber et al., 2011), in which the authors report their results in a dataset containing about 4500 RGB-D images recorded in an University hall from three statically mounted Kinect cameras. Unfortunately, there is no ground plane ground truth available, thus we only report results in the image plane. To compare with their results, we follow the evaluation procedure described in (Spinello & Arras, 2011), that is, without penalizing approaches for not detecting occluded or hidden people. We also report our performance for the full dataset separately. Some samples of this dataset can be visualized in Figure 12b.

Regarding the data partition approach for the training and testing subsets, Table 5 summarizes the total number of people labeled in the ground truth in these frames (column `#PeopleFull`), and the partition statistics for the training subsets (column `#PeopleTrain`) and for the testing subsets (column `#PeopleTest`). Table 5 also provides the percentages corresponding to these subsets below the accumulated totals for the `#PeopleTrain` and `#PeopleTest` columns to give an idea of their relative sizes[1].

4.2. Experimental setup

4.2.1. Evaluated algorithms

We compared the performance of our proposal `PD3net` with up to ten different strategies described in the literature that use depth cameras with an elevated non-overhead position in a people detection task.

[1]Extended details for the interested reader can be found in Table A.10 in Appendix A.

Table 5: Data partition summary (full details in Table A.10).

	#PeopleFull	#PeopleTrain	#PeopleTest
GESDPD	54215	36063 (67%)	18152 (33%)
GFPD	13827	9490 (69%)	4337 (31%)
EPFL-LAB	2287	1385 (61%)	902 (39%)
EPFL-CORRIDOR	8032	3598 (45%)	4434 (55%)
KTP	6719	3299 (49%)	3420 (51%)
UNIHALL	2979	2054 (69%)	925 (31%)
ALL Datasets	**88059**	**55889 (63%)**	**32170 (37%)**

The algorithm selection procedure initially had to do with the nature of the algorithmic approximations used, as we wanted to include both classical methods (Dollár et al., 2009; Shotton et al., 2011; Munaro & Menegatti, 2014; Spinello & Arras, 2011; Bagautdinov et al., 2015), and those based on DNNs (Redmon & Farhadi, 2018; Zhou et al., 2017b; Uijlings et al., 2013). Within these, we tried to include the widest possible variability of the proposed solutions. The second criterion was the information they use. Priority has been given to the search for methods only on depth only data (Bagautdinov et al., 2015). Given the lack of this type of methods, we also considered those merging RGB and depth information (Zhou et al., 2017b; Spinello & Arras, 2011; Shotton et al., 2011; Munaro & Menegatti, 2014), and finally those using only RGB (Dollár et al., 2009; Zhou et al., 2017b) methods. Because of this second criterion and given the lack of depth-only methods, we decided to develop additional depth-only proposals, such as the YOLO-Depth method, in order to bring variability to this group. Given the limited number of state-of-the-art methods that address these tasks and share data or source code, we have not been very restrictive in adding methods to the evaluation. The finally selected proposals allow for a retrospective evaluation of how the resolution of the person detection task has progressed over time and depending on the information or nature of the methods.

Table 6 shows the main characteristics of all the evaluated methods, considering both classical and DNN approaches. It is relevant to note that we are comparing our depth-only proposal with others using different combinations of RGB and depth information, which imposes strong differences in the quality of the exploited data.

Table 6: Evaluated state-of-the-art methods for multiple people detection.

Solutions	Methods	Input information	References
Classic	ACF	RGB	Dollár et al. (2012)
	PCL-Munaro	RGB-D	Munaro & Menegatti (2014)
	Kinect2	RGB-D	Kinect-SDK (2014)
	Unihall	RGB-D	Spinello & Arras (2011)
	DPOM	D	Bagautdinov et al. (2015)
DNN	RCNN	RGB	Girshick et al. (2013)
	RGBCNN	RGB	Zhou et al. (2017b)
	RGBCECDCNN	RGB-D	Zhou et al. (2017b)
	YOLO v3	RGB	Redmon & Farhadi (2018)
	YOLO depth	D	New development

Below, we explain in detail the evaluated state-of-the-art methods that we use in the comparison:

- `ACF` (Dollár et al., 2012): That uses a RGB detector, based on AdaBoost and aggregated channel features (Dollár et al., 2009) to give an idea on how a state-of-the-art detector that does not use depth can perform on these sequences.

- `PCL-MUNARO` (Munaro & Menegatti, 2014): That uses a RGB-D detector based on modified HOG features on regions extracted by depth segmentation.

- `KINECT2` (Kinect-SDK, 2014): Based on the results obtained from the human pose estimation of the latest Kinect for Windows SDK (Kinect-SDK, 2014). It is not publicly known what specific algorithm is used. However in (Shotton et al., 2011), the authors report that their algorithm is at the core of the human pose estimation for the older version of the Kinect software. For undisclosed reasons, the framework supports tracking up to 6 people, with the working depth range limited to 4.5 meters. To ensure fairness, we keep these restrictions in mind when using the `EPFL-LAB` and `EPFL-CORRIDOR` datasets, that is, we do not penalize algorithms for not detecting more than 6 people or people who are further than 4.5 meters away.

- `UNIHALL` (Spinello & Arras, 2011): That uses a RGB-D detector based on HOG and HOD features. The code is not available and we, therefore, report only a single point on the precision-recall curves.

- `DPOM` (Bagautdinov et al., 2015): This method checks for a human presence on the ground plane using *Bayesian inference*. Before that, the first step is to detect the ground plane and remove it from the 3D

processed point cloud. After the ground plane elimination, all the remaining points will be clustered and segmented as possible person detections. The algorithm stops when it groups all the clustered regions in the whole depth image.

- `RCNN` (Girshick et al., 2013): That uses a region proposal based CNN, with an architecture that was originally used for region segmentation and classification. In addition to this architecture, it uses the region of interest method from (Zhou et al., 2017b).

- `RGBCNN` (Zhou et al., 2017b): That uses a RGB CNN-based object detector with region of interest selection method and selective search from (Uijlings et al., 2013).

- `RGBCECDCNN` (Zhou et al., 2017b): That uses a RGB and Depth combined CNN-based object detector with CECD channel encoding.

It is important to highlight that in the case of the RCNN based strategies, the results could be poor due to the fact that their relatively simple architecture was originally designed for a region segmentation and classification system, not specifically oriented to the person detection task.

In addition to these methods, and to allow for additional experimentation on our `GESDPD` dataset, we have also used the YOLO (*You Only Look Once*) object detector (Redmon & Farhadi, 2018) to be applied in the person detection task using depth and RGB data. Our use of the YOLO strategy comprises two approaches, using the YOLO system *as is*, and adapting it to more properly handle the depth information. These are the two developed systems on this line:

- `YOLO-V3` (Redmon & Farhadi, 2018): RGB object detector based on CNN and the bounding box based approximation. This implementation was based on the original Yolo V3 architecture and trained with the COCO 2017 dataset (Lin et al., 2014). The parameters used to configure the architecture were an input image size of 416×416, 9 anchors, and all the COCO classes. The evaluation is done on the RGB data available for the corresponding datasets.

- `YOLO-Depth`: Depth object detector based on the (Redmon & Farhadi, 2018) structure, modifying it so that it is able to use depth images as input instead of RGB images. This implementation is trained with the depth datasets used in this paper. Thus the size of the depth training size is lower that that of the RGB training used in the original YOLO, which can affect the results obtained by this approach. Nevertheless, this is still a valid approach to be considered for its potential to work using depth data. The parameters used to configure the architecture, were an input image size of 416×416, 9 anchors,

and only with the person class. No structural changes have been made to the architecture as compared to the original version.

4.2.2. Evaluation Metrics

To provide a detailed view of the performance on the evaluated algorithms, we have calculated the main standard metrics in a detection problem, namely *Precision*, *Recall*, and F_{1score}. The results are shown both in tables (to provide the precise values), and in bar graphs (to provide an easier visual comparison).

In the scoring process, the following convention has been adopted regarding occlusions: In the case that an occluded person is not detected, it does not generate a detection error if the heads of the users are occluded in a percentage higher than 50%. Therefore the occlusion limitation does not practically affect occlusions between the bodies of the users, but only considering the occlusion of the upper body part.

In the results tables, we also include confidence intervals for the F_{1score} metric, for a confidence value of 95%, to assess the statistical significance of the results when comparing different strategies. Additionally, the confidence intervals for the *Precision*, *Recall*, and F_{1score} metrics are also shown in the bar graphs.

In the evaluation with real data, *Precision-Recall* curves are also included. In the case of the `DPOM`, `ACF`, `KINECT2`, `UNIHALL`, `PCL-MUNARO`, `RGBCNN`, and `RGBCECDCNN` algorithms, we use those already provided in (Zhou et al., 2017b) and (Bagautdinov et al., 2015). For the `YOLO-Depth` and `YOLO-V3` (Redmon & Farhadi, 2018) algorithms, we generate the *Precision-Recall* curves through a sweep of the algorithm confidence threshold. However, building these curves in our proposal is somehow artificial, leading to curves with strange appearances, as the threshold sweep is done on the Gaussian distribution threshold. In Section 4.2.3 we describe the *Precision-Recall* curve generation procedure for our proposal, and the considerations that should be taken into account when addressing the interpretation of the *Precision-Recall* curves for our system.

In the case of the F_{1score}, the obtained curves are scanned with each of the possible thresholds, to obtain their corresponding F_{1score} and to be able to create the F_{1score} vs threshold curve, which allows choosing the best operation point or range of operation points, with a criterion aimed at getting the best possible F_{1score}.

An additional parameter of the evaluation metrics is the region on which they are calculated, including restrictions on the image plane and the depth range. In some cases (that will be clearly stated when showing the results), the evaluation metrics have been calculated considering:

- An image plane region which is smaller than the full-frame size. The objective is to only focus on *full detections* of people, avoiding the cases of *incomplete persons* in which they could otherwise be partially *occluded* by the image borders.

- A depth range that is smaller than the full depth range of the sensor. The objective is avoiding measurements greatly contaminated with noise that, for some recording conditions, can be found near the image borders or at depths near the sensor sensing limits.

4.2.3. Precision-Recall curves generation for the PD3net algorithm

In this section we provide details on how the threshold to be optimized in PD3net works. The need for this explanation is given because it is not a conventional confidence threshold like the one that can be found in algorithms such as YoloV3 (in which the threshold sweep generates a smooth variation in the performance curves), so that it can produce effects that are radically different from the usual ones, which will be reflected in the appearance of the *Precision-Recall* curves.

Figure 13 shows a schematic example of a Gaussian-like likelihood/confidence map, which could be generated by our system. In the 2D representation of the figure (left image), we can see two small Gaussian-like regions with a small overlap between them. In the 3D representation (right image), the apparent overlap between the two Gaussian structures is greater than the one observed in 2D.

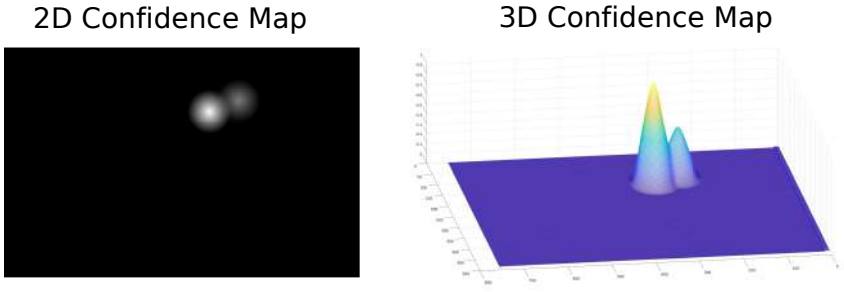


Figure 13: Schematic representation of the likelihood/confidence map in 2D (left) and 3D (right).

The decision threshold that we use in the PD3net algorithm can be shown geometrically as a plane parallel to the XY plane that would *cut* the Gaussian distributions at a given height (decision threshold value), separating them both by their maximum height and by their overlap at the threshold level. So, unlike conventional thresholds, this threshold affects both the inter-Gaussian overlap and the maximum confidence peak.

Figure 14 shows the three-dimensional representation of the two detected Gaussians after the application of three thresholds values at 0.1, 0.4, and 0.8.

In the case of the 0.1 threshold, it can be roughly seen that the two Gaussians are overlapping by their intersection at the threshold level, so that in terms of detections, it would generate a single detected person, as there would be a single joint area in the 2D map. In the case of the 0.4 threshold, it can be observed that the Gaussians no longer have overlap considering the threshold level, so that the two Gaussians will be separated

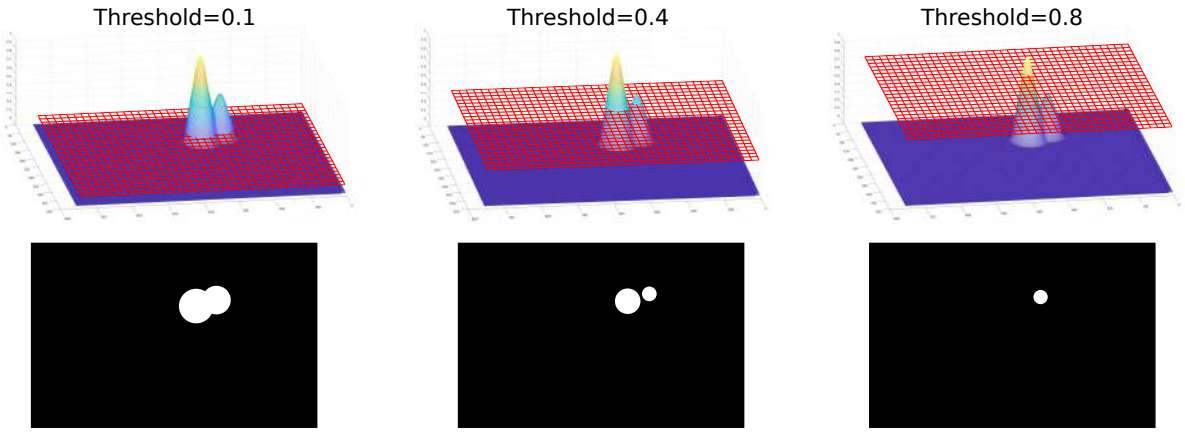


Figure 14: Representation of the effect of different threshold values.

in the 2D map, and both of them will be correctly detected. Finally, in the case of the 0.8 threshold, only the Gaussian with the highest height would be found in the 2D map, as the other one is below the detection threshold, thus being discarded.

To show a real example on how we build the *Precision-Recall* curves for the `PD3net` system, Figure 15 includes data obtained from a sample frame of the `GFPD` dataset. The upper part of the figure includes the 2D ground truth likelihood/confidence map (left image), showing three users with Gaussians normalized at their maximum peaks of 1.0; and the network prediction (right image), with four estimated Gaussians whose maximum peaks correspond to 1.0, 1.0, 0.7 and 0.3 (not all the predictions will reach the 1.0 level).

In the lower part of Figure 15 we can see two graphs. The one to the left is the *Precision-Recall* graph on the test set, and the one to the right is the $F_{1score} - Threshold$ graph calculated on the training set. In these two graphs we find three well-differentiated sections that will allow to get an idea of the distribution of *Precision-Recall* curve values when varying the threshold value:

- The first region, indicated by a blue ellipse, covers a threshold range below 0.1. In that range, due to the increased Gaussian overlapping and the acceptance of low confidence Gaussians, both the number of false positives and false negatives increase.

- The second region, indicated by a black ellipse, covers a threshold range between 0.1 and 0.4. In that range, as we increase the threshold value the Gaussian overlapping decreases, and also the Gaussians with low confidence values can be discarded, thus decreasing the number of false positives and false negatives.

- The third region, indicated by a green ellipse, covers a threshold range between 0.4 and 0.8, and within this range is where the system performs the best, with well and correctly separated Gaussians,

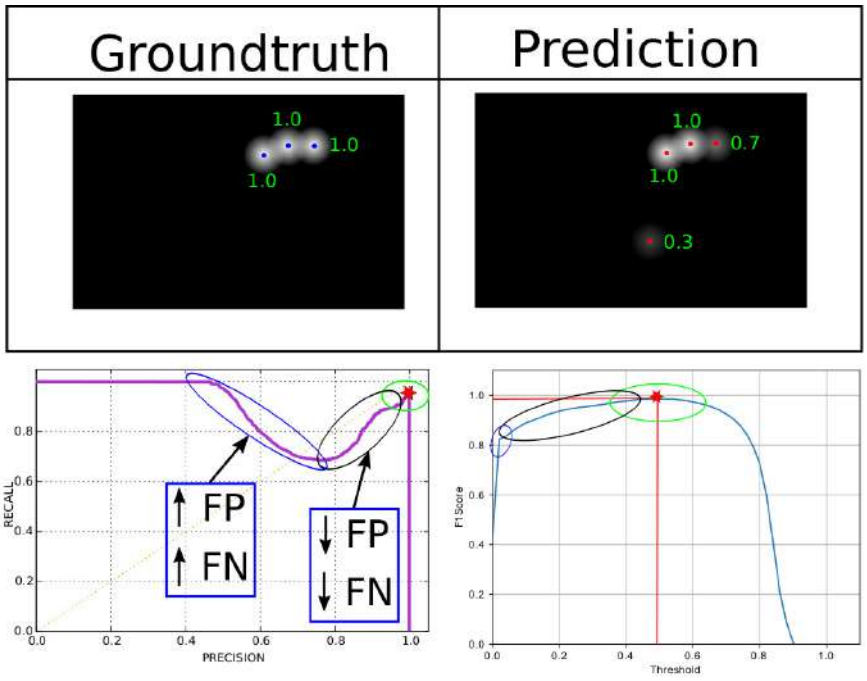


Figure 15: Representation of the threshold variation effect on a GFPD example. From this example, we can conclude that a threshold variation have a simultaneous impact on both the false positive rate (all the low confidence Gaussians for low threshold levels) and the false negatives (all the overlapping Gaussians for low threshold levels).

and good rejection of low confidence ones. This behavior leads to the optimum working point shown as a red star.

4.2.4. Training and threshold selection strategy

As discussed above, one of the key issues in the PD3net proposal is the correct selection of the detection threshold used in the $C^{polished}$ output map. This selection is very relevant because the threshold is responsible for deciding which Gaussian distributions are considered as possible person detections. Both the network and the threshold training have been done using two different approaches:

- *Tuned* network training and threshold selection (Dataset specific): In this scenario, the network is trained on the training subset for each specific dataset, and the corresponding threshold is selected as that achieving the best F_{1score} on this training subset. This way we can evaluate the best possible result with the network and threshold tuned to the conditions of each particular dataset.

- *Global* network training and threshold selection: In this scenario, the network is trained on all the available training subsets, and the threshold is selected as that achieving the best average F_{1score} evaluated on all of these training subsets. This way we can evaluate a more realistic performance using the same single network and threshold for all the datasets.

We will provide results for both approaches in the next subsections.

4.3. Results training with Synthetic Data Only

Our first evaluation task was devoted to exploring the extent to which training with simulated data could cope with the variability found in real datasets. To do so, we first run a set of preliminary experiments using the GESDPD as the training data, and both GFPD and EPFL-LAB for testing.

The first row in Table 7 shows the results of our proposal trained with the GESDPD when evaluated on an independent subset of the GESDPD dataset. The results indicate that the simulated training data seems to be valid to achieve reasonably good results when facing similar simulated conditions, with an overall error below 6.8% and an F_{1score} of 96.5%.

Table 7: Results on the GESDPD and EPFL-LAB datasets, trained on the training subset of the GESDPD.

	#FramesTest	#PeopleTest	FN ($FNR\%$)	FP ($FPR\%$)	*Error*	*Precision*	*Recall*	F_{1score}
GESDPD	2200	3176	212 (6.68%)	3 (0.09%)	6.77%	99.89%	93.32%	96.50 ± 0.11%
EPFL-LAB	950	1959	474 (24.07%)	3 (0.15%)	24.35%	99.80%	75.80%	86.16 ± 1.53%

The second row of Table 7 shows the results of our proposal trained with the GESDPD and evaluated on a testing subset of the EPFL-LAB dataset. In this case, the search area in the image plane was restricted to 280×150 and the depth range considered up to $3.5m$. Our objective was to produce an environment more controlled in which unnecessary noise is removed from the image. For example, this noise was present in the ceiling, which occupies around 40% of the image or on the floor. These results indicate that the simulated training data is not capable of leading to reliable results when facing a realistic dataset, with an overall error above 24% and an F_{1score} of 86.16%.

These preliminary experiments clearly indicate that training with simulated data only can be a good starting point, but it is still far from allowing us to get good results on realistic data. This conclusion led us to develop a training procedure in two stages: first a full training using the simulated data, and then use a small subset of the evaluated datasets to fine-tune the pre-trained model.

4.4. Results on real data

In this section, we present the results and discussion when evaluating our proposal on the realistic datasets described in Section 4.1, and using all the available algorithms in each case. Table 8 shows the results of all the evaluated algorithms in all the available datasets[2]. Empty cells in the table are due to the fact that the

[2]Except for RCNN, RGBCNN, RGBCECDCNN on the UNIHALL dataset that have been excluded due to their low performance, as can be seen in Table A.11 in Appendix A.

corresponding algorithms were not available, and we have replicated the results published by the respective authors on the given datasets.

Table 8: Performance results on all the avaible datasets comparing the "tuned" and "global" versions for the PD3net proposal ($P = Precision, R = Recall$ (best result are displayed with green background, and orange background indicates results within the best one significant bands).

	GFPD			KTP			UNIHALL			EPFL-LAB			EPFL-CORRIDOR		
	P	R	F_{1score}	P	R	F_{1score}	P	R	F_{1score}	P	R	F_{1score}	P	R	F_{1score}
PD3net tuned	99.7	96.36	98.0 ± 0.20	96.2	96.3	96.3 ± 0.64	92.5	99.2	95.7 ± 1.30	98.8	94.5	96.6 ± 1.18	90.3	80.1	84.9 ± 1.05
PD3net global	100.0	95.9	97.9 ± 0.21	95.4	95.1	95.3 ± 0.71	91.2	97.3	94.2 ± 1.51	99.5	92.5	95.9 ± 1.30	90.9	76.1	82.8 ± 1.11
YOLO-V3 (Redmon & Farhadi, 2018)	82.3	86.4	84.3 ± 0.53	99.1	98.2	98.7 ± 0.39	84.5	93.1	88.6 ± 2.05	93.3	93.0	93.2 ± 1.65	58.6	59.8	59.2 ± 1.45
YOLO-Depth	79.8	55.1	65.2 ± 0.69	91.1	72.3	80.6 ± 1.32	63.2	52.4	57.3 ± 3.19	90.2	89.2	89.7 ± 1.98	78.4	47.9	59.5 ± 1.45
PCL-MUNARO (Munaro & Menegatti, 2014)				98.7	76.4	86.1 ± 1.16	82.5	78.2	80.3 ± 2.56	96.9	86.5	91.4 ± 1.83	95.0	56.3	70.7 ± 1.34
DPOM (Bagautdinov et al., 2015)				95.3	94.5	94.9 ± 0.74	90.2	98.1	94.0 ± 1.53	98.5	85.4	91.5 ± 1.82	96.3	70.9	81.7 ± 1.14
ACF (Dollár et al., 2012)				87.4	72.7	79.4 ± 1.36	90.2	85.4	87.7 ± 2.11	83.8	86.4	85.1 ± 2.33	66.3	40.3	50.1 ± 1.47
UNIHALL (Spinello & Arras, 2011)							86.3	84.5	85.4 ± 2.28						
KINECT2 (Kinect-SDK, 2014)										99.8	38.2	55.3 ± 3.24	86.3	41.2	55.8 ± 1.46

For the PD3net algorithm, two results are provided for the conditions discussed in Section 4.2.4: one with the *tuned* version of the network model and threshold, and the other with their *global*ly trained versions. The results show, as expected, a slight decrease in performance when using the global threshold as compared to the tuned one, but these differences are not statistically significant. This support the conclusion that the PD3net strategy is robust enough to face very different training and testing conditions.

In the next subsections we discuss the results for each specific dataset and the *tuned* approach, providing a graphical comparison of the evaluated metrics, along with the *Precision-Recall* and $F_{1score} - threshold$ curves for the *tuned* approach. The discussion of these curves for the *global* approach is later addressed in Section 4.4.7.

4.4.1. Results for the GFPD database

The second column of Table 8 and Figure 16 show the results when evaluating on the GFPD dataset, using our proposal PD3net and the YOLO-V3 and YOLO-Depth algorithms. We could not apply any of the other proposals described in Section 4.2 as they were not readily available.

The results in the table show that both approaches of the PD3net algorithm clearly outperform the YOLO-based strategies, being the best in the three metrics used and in a statistically significant way, placing

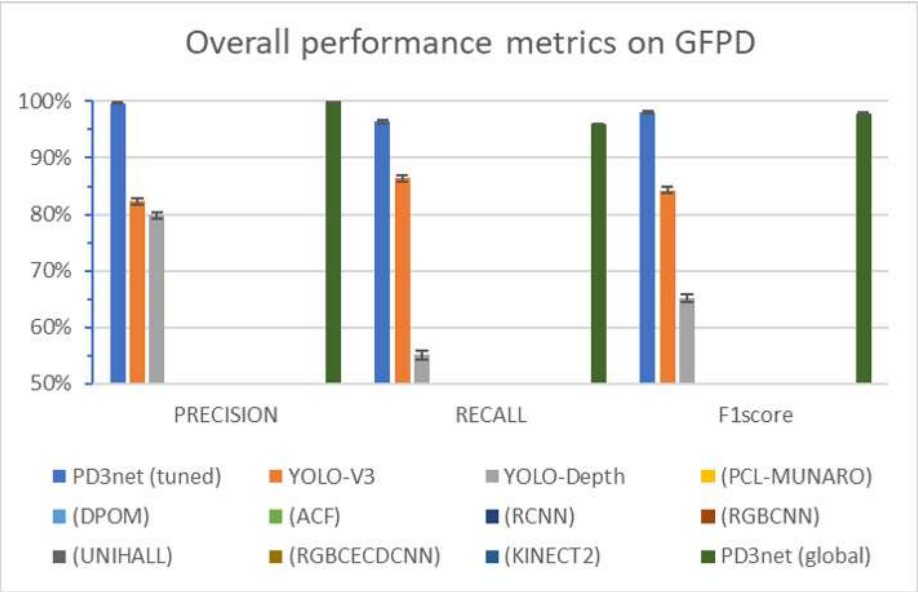


Figure 16: Results on the GFPD dataset.

a great distance between it and YOLO-V3 as the second-best solution. The worse results of the YOLO-based algorithms probably rely in the fact that they were not designed to deal with depth data, and to the great number of occlusions and complex situations that GFPD contains.

Figure 17 shows the *Precision-Recall* curve corresponding to the behavior of the three algorithms and the $F_{1score} - threshold$ curve corresponding to the PD3net algorithm. As described in Section 4.2.3, the appearance of the curve for our proposal is far from standard, and it is due to the effect of the threshold sweep procedure so that the area under the curve should not be taken into account as the comparison metric. If we consider the actual working point of the algorithm (marked with a red start), the figure clearly shows that our working point greatly outperforms the other proposals.

With respect to the *Precision-Recall* curve, the F_{1score} metric is represented as a function of the threshold sweep used to generate the *Precision-Recall* curve. This curve shows how the working point is located in the middle of a reasonably wide and flat area which outperforms the other two algorithms, indicating that its sensitivity to the threshold value is reduced.

Figure B.29 in Appendix B shows a sample frame of the results achieved by our proposal on the GFPD dataset.

The capabilities of our proposal are further established in the next sections where the availability of performance metrics on additional datasets and with a broad range of different algorithms are exploited in the comparisons.

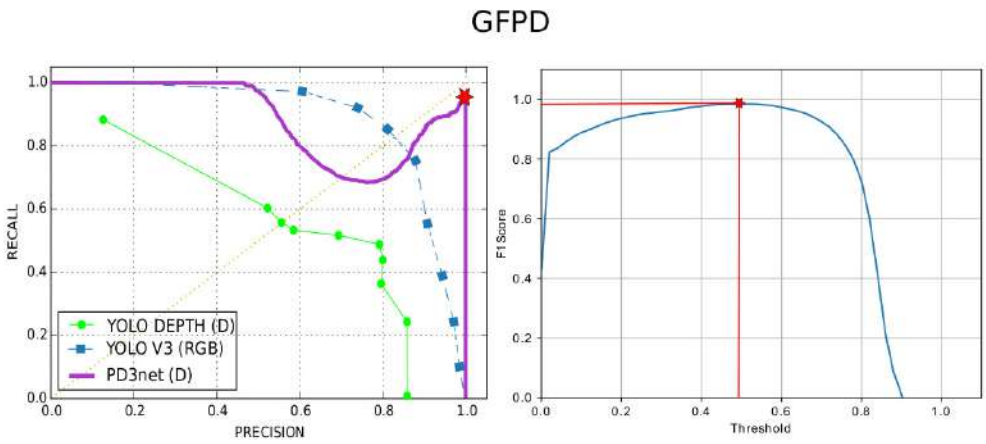


Figure 17: *Precision-Recall* curve comparison and F_{1score}-Threshold results for the experiments on the GFPD dataset.

4.4.2. Results for the KTP database

The third column of Table 8 and Figure 18 show the results when evaluating on the KTP dataset, using our proposal PD3net, the YOLO-V3 and YOLO-Depth algorithms, and three other proposals from the literature: PCL-MUNARO, DPOM and ACF.

From the table, it is surprising the top performance of the YOLO-V3 algorithm in terms of F_{1score}, with statistically significant different as compared with the second-best result (achieved by our PD3net). In this case, the YOLO-V3 system obtains better results than all the other proposals as the KTP database does not contain a lot of hard occlusions and has been prepared from a low frontal perspective, which are the perfect conditions for the operation of YOLO-V3, whose training images are also low frontal and with almost no occlusions.

Figure 19 shows the *Precision-Recall* curve corresponding to the behavior of the five algorithms and the $F_{1score} - Threshold$ curve corresponding to the PD3net algorithm. Again, the later curve shows how the working point is placed in a wide reasonably flat area, indicating that its sensitivity to the threshold value is small. In addition to this, we can see that YOLO-V3 shows a *Precision-Recall* curve that is near the perfection in KTP, compared to DPOM or PD3net.

Figure B.30 in Appendix B shows a sample frame of the results achieved by our proposal on the KTP dataset.

4.4.3. Results for the UNIHALL database

The fourth column of Table 8 and Figure 20 show the results when evaluating on the UNIHALL dataset, using our proposal PD3net, the YOLO-V3 and YOLO-Depth algorithms, and four other proposals from the

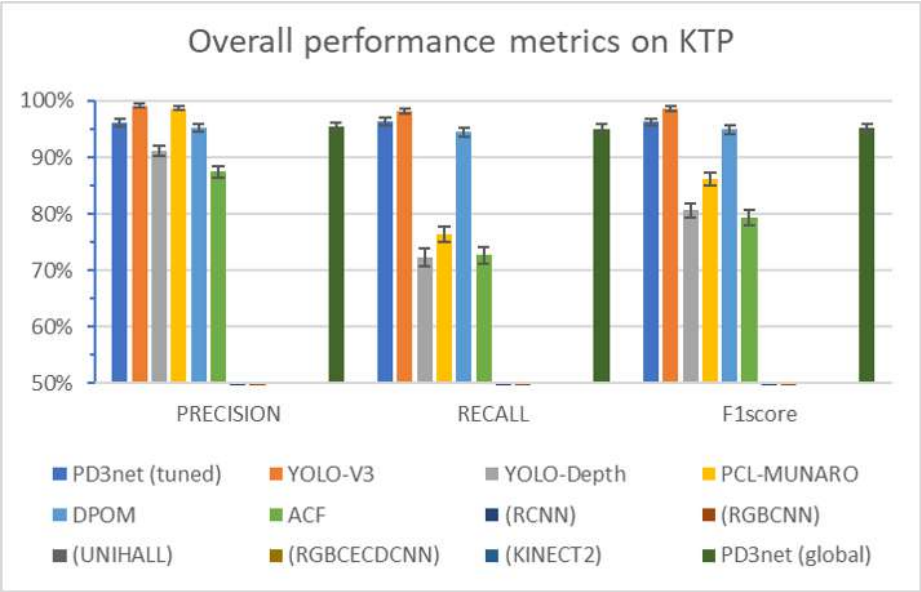


Figure 18: Results on the KTP dataset.

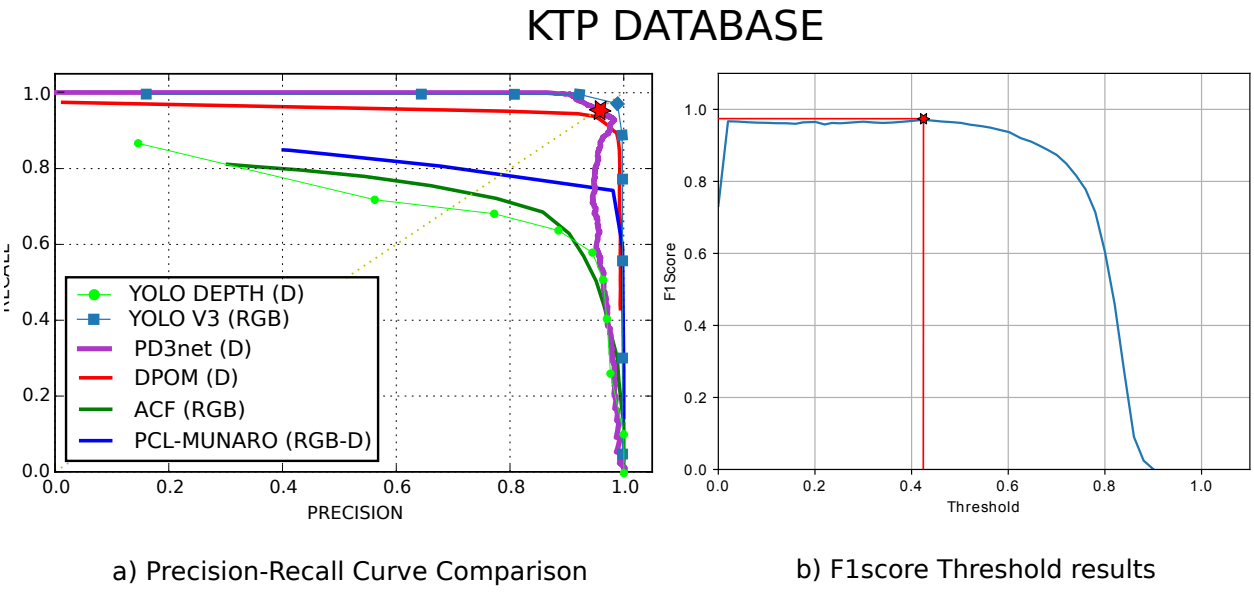


Figure 19: *Precision-Recall* curve comparison and F_{1score}-Threshold results for the experiments on the KTP dataset.

literature: PCL-MUNARO, DPOM, ACF and UNIHALL[3].

In this case, the PD3net algorithm is the best one in terms of the three evaluated metrics, but its improvement as compared with the second best (DPOM) is not statistically significant.

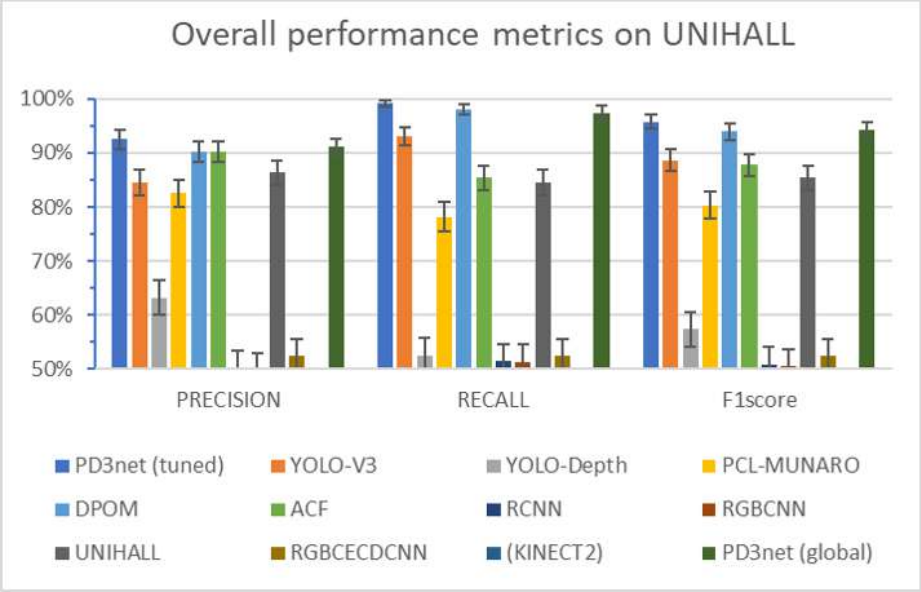


Figure 20: Results on the UNIHALL dataset.

Figure 21 shows the *Precision-Recall* curve and the F_{1score} sweep curve corresponding to PD3net algorithm. With respect to the F_{1score} metric, the curve shows a stable working point that falls sharply compared to other databases from a threshold of 0.6, so we can see that the working point represented by a sharp peak in the *Precision-Recall* curve is, in fact, a long-range of stable working points. In this case, the *Precision-Recall* curve clearly shows that the proposed PD3net algorithm has the best possible working point, very closely followed as the second-best solution by DPOM and with a large distance in this case to the best third solution represented by YOLO-V3, which is again affected by the presence of strong occlusions in this dataset.

Figure B.31 in Appendix B shows a sample frame of the results achieved by our proposal on the UNIHALL dataset.

4.4.4. Results for the EPFL-LAB database

The fifth column of Table 8 and Figure 22 show the results when evaluating on the EPFL-LAB dataset, using our proposal PD3net, the YOLO-V3 and YOLO-Depth algorithms, and four other proposals from the literature: PCL-MUNARO, DPOM, ACF, and KINECT2.

[3]As discussed above, we excluded here the results for RCNN, RGBCNN, RGBCECDCNN due to their low performance (see Table A.11 in Appendix A).

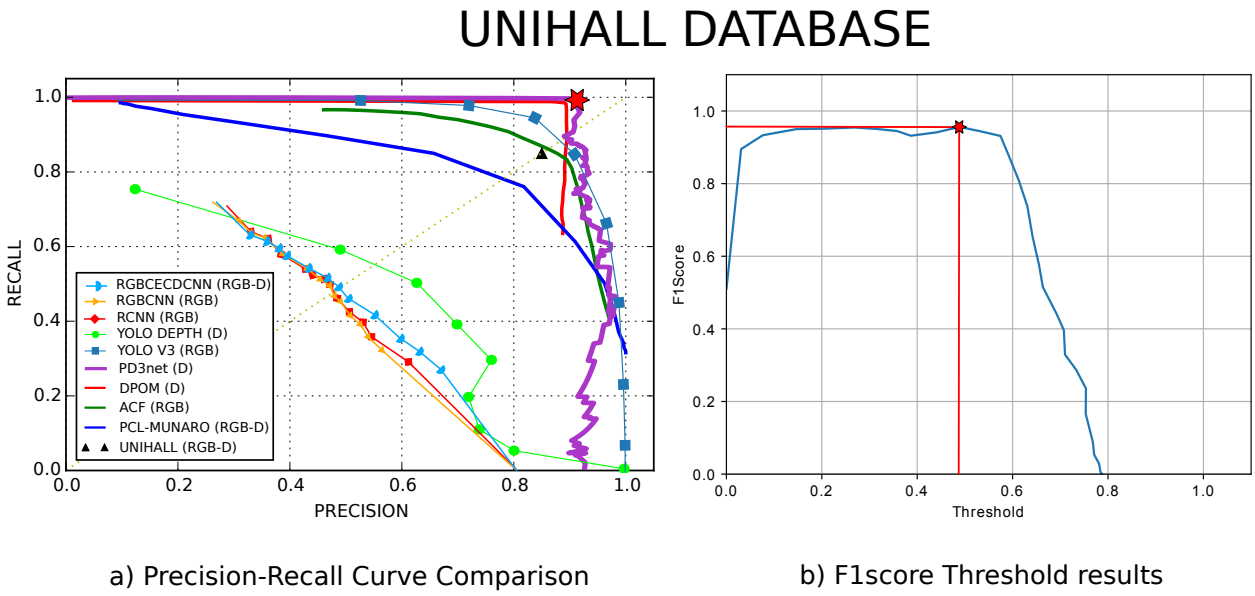


Figure 21: *Precision-Recall* curve comparison and F_{1score}-Threshold results for the experiments on the `UNIHALL` dataset.

In this case, the `PD3net` algorithm is the best in terms of the three evaluated metrics, and its improvements as compared with the second best (`YOLO-V3`) are again statistically significant.

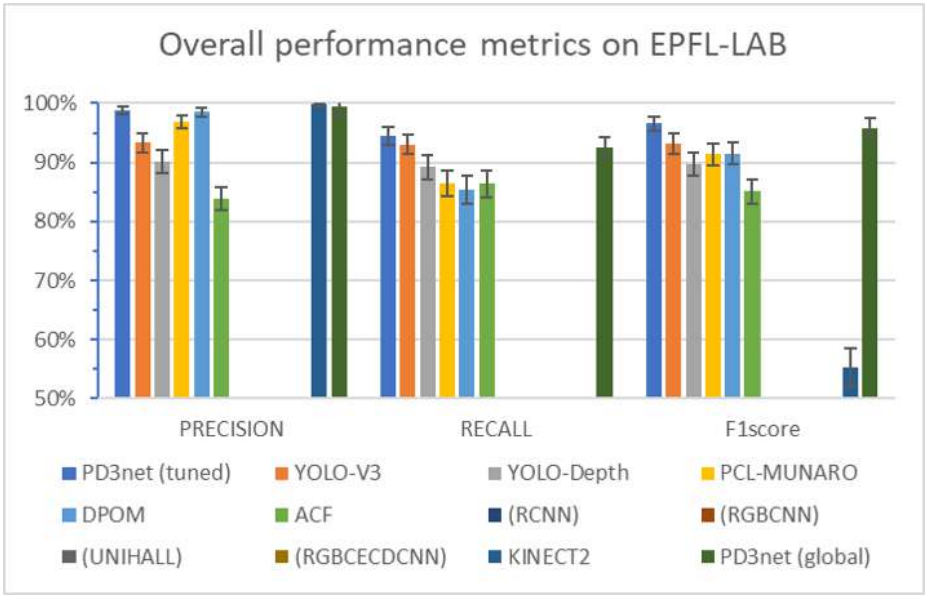


Figure 22: Results on the `EPFL-LAB` dataset.

Figure 23 shows the *Precision-Recall* curve and the F_{1score} sweep curve corresponding to `PD3net` algorithm. Again, the best algorithm in terms of working point in the *Precision-Recall* curve is `PD3net`. This curve exhibits a very different behavior from the one seen in previous datasets, forming a very steep

slope and peak. To demonstrate that this peak represents a large number of good and stable working points, the F_{1score}-*Threshold* graph clearly shows how there is a range of threshold values (from 0.6 to 0.8 that is practically stable in terms of F_{1score}.

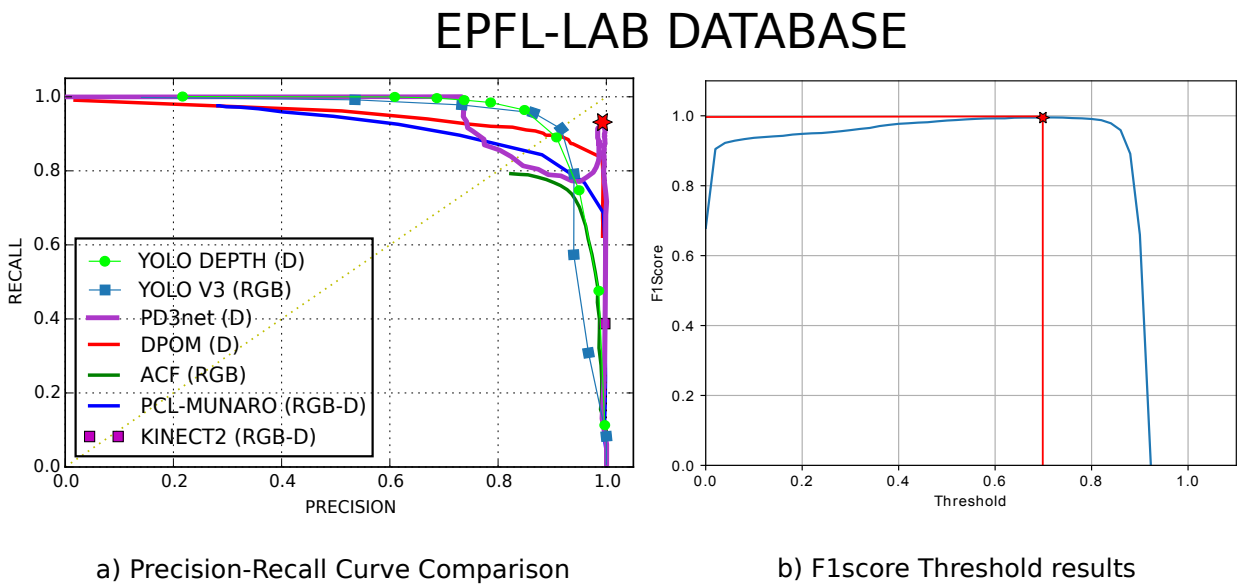


Figure 23: *Precision-Recall* curve comparison and F_{1score}-Threshold results for the experiments on the EPFL-LAB dataset.

Figure B.32 in Appendix B shows a sample frame of the results achieved by our proposal on the EPFL-LAB dataset.

4.4.5. Results for the EPFL-CORRIDOR database

The sixth column of Table 8 and Figure 24 show the results when evaluating on the EPFL-CORRIDOR dataset, using our proposal PD3net, the YOLO-V3 and YOLO-Depth algorithms, and four other proposals from the literature: PCL-MUNARO, DPOM, ACF, and KINECT2. In this case, the PD3net algorithm is again the best in terms of recall and F_{1score} metrics and the third in terms of precision (although the good results in Precision by the DPOM algorithm are related to a poor behavior in terms of Recall). Its improvements in terms of Recall and F_{1score} compared to the second best algorithm (DPOM) are statistically significant.

Figure 25 shows the *Precision-Recall* curve and the F_{1score} sweep curve corresponding to PD3net algorithm. EPFL-CORRIDOR is one of the most complex databases, bringing together difficulties such as many occlusions and people very close to each other, detection in small spaces with a lot of perspective, high noise in the depth data and many false detections due to people occluded by the image borders. All these issues lead to algorithms like YOLO-V3 to drastically reduce their performance, while PD3net manages to keep a more robust performance, this time with a less stable working point in terms of F_{1score}, as compared to the

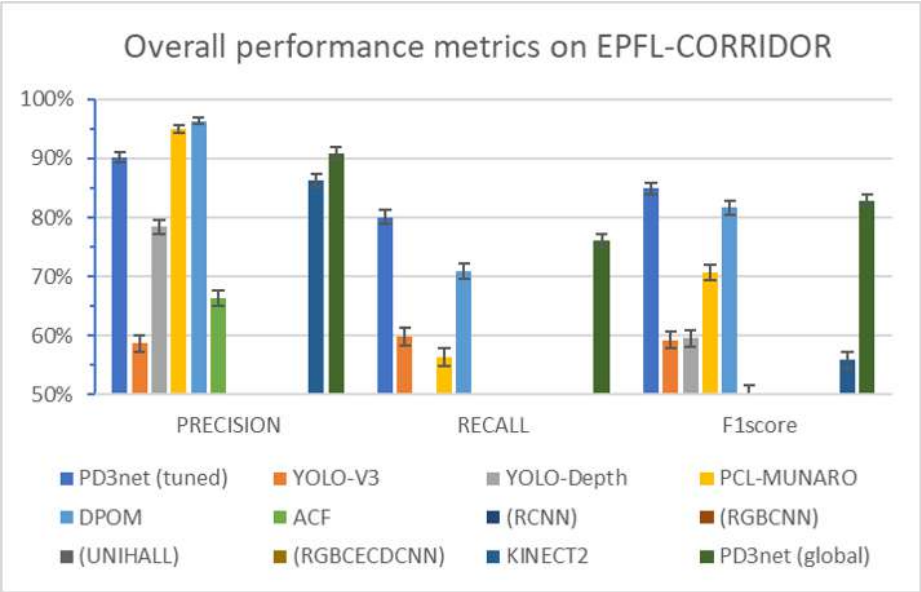


Figure 24: Results on the EPFL-CORRIDOR dataset.

results obtained in other databases. The second best system is again DPOM, in this case with significant differences in the results in terms of working point. Finally, in third place we find PCL-MUNARO closely following the DPOM but with remarkable differences in their *Precision-Recall* curves.

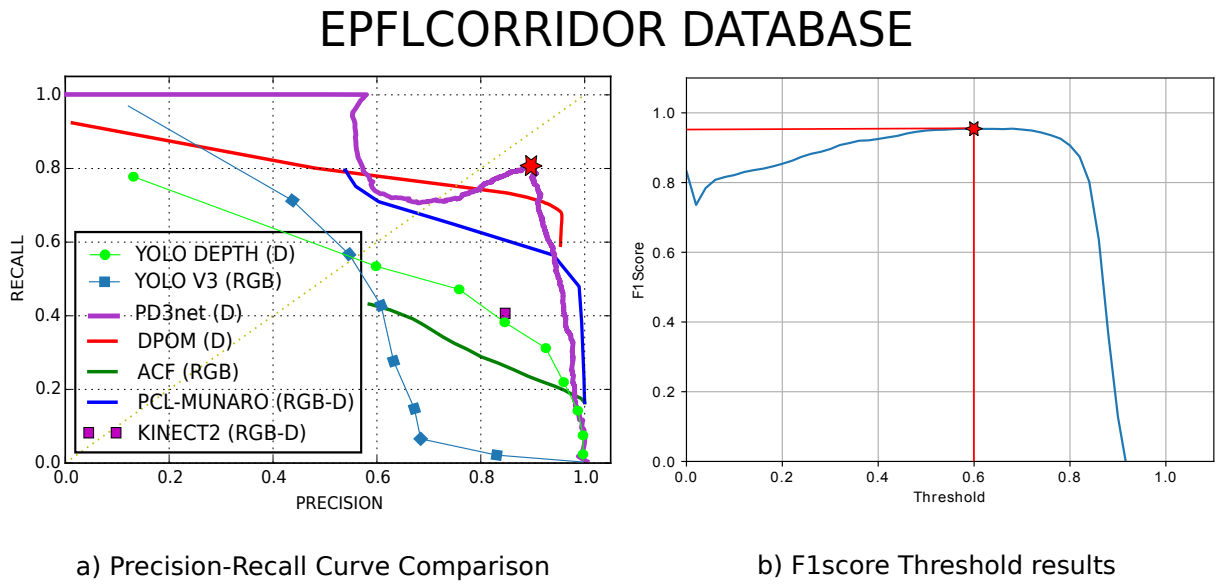


Figure 25: *Precision-Recall* curve comparison and F_{1score}-Threshold results for the experiments on the EPFL-CORRIDOR dataset.

Figure B.33 in Appendix B shows a sample frame of the results achieved by our proposal on the EPFL-CORRIDOR dataset.

4.4.6. *Average results for all the available datasets*

Table 9 and Figure 26 show the weighted average results when evaluating all the available datasets using all the available algorithms[4]. They have been calculated integrating the results from the above sections, weighting each result according to the number of ground truth elements of each testing subset.

Table 9: Average weighted results using all the available datasets.

	Precision	*Recall*	F_{1score}
PD3net tuned	97.51	93.80	95.62 ± 0.24
PD3net global	97.68	92.59	95.07 ± 0.25
YOLO-V3 (Redmon & Farhadi, 2018)	81.02	84.05	82.51 ± 0.45
YOLO-Depth	80.75	57.08	66.88 ± 0.55
PCL-MUNARO (Munaro & Menegatti, 2014)	95.29	68.31	79.57 ± 0.80
DPOM (Bagautdinov et al., 2015)	95.57	83.19	88.95 ± 0.62
ACF (Dollár et al., 2012)	77.67	60.35	67.92 ± 0.93
UNIHALL (Spinello & Arras, 2011)	86.30	84.50	85.39 ± 2.28
KINECT2 (Kinect-SDK, 2014)	88.58	40.69	55.77 ± 1.33

In the average comparison over all the datasets, we can observe that the statistical significance of the results is increased, provided the higher number of considered samples. In terms of *Precision*, the best algorithm is PD3net, with DPOM and PCL-MUNARO coming second and third. In terms of *Recall* the differences increase, with a large distance among the first three best proposals. The top system is again PD3net, followed by DPOM and YOLO-V3. Finally, considering the F_{1score} that relates the two previous metrics offering a final joint one, PD3net comes first, clearly surpassing DPOM, which is the second-best method with a wide margin of statistical significance.

[4]In the table we have again excluded the results for the RCNN, RGBCNN, RGBCECDCNN, due to their low performance, and they have been moved to Table A.12 in Appendix A.

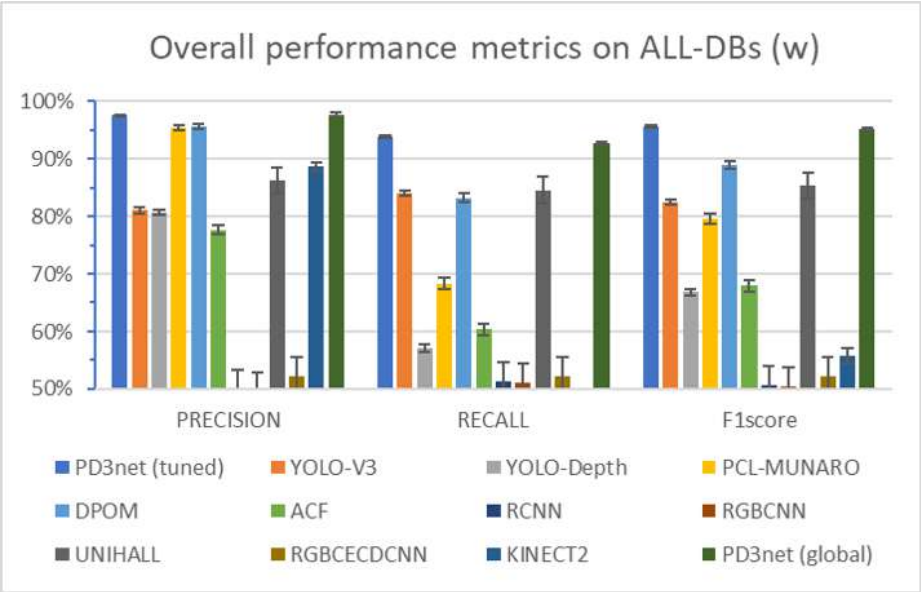


Figure 26: Average weighted results using all the available datasets.

4.4.7. *Discussion on the Precision − Recall curves for the* Global *training approach*

In this section we provide details on the comparison of our proposal with the other state of the art methods considering the *Precision − Recall* and $F_{1score} - Threshold$ curves, when the network model and the threshold have been trained using all the available training subsets (referred to as the *Global* approach in Section 4.2.4).

Figure 27 shows the *Precision-Recall* curves for all the datasets and different algorithms. The curves obtained in this case for each dataset are very similar to those found in Figures 17, 19, 21, 23 and 25, which is consistent with the performance metric results in Table 8, which showed non statistically significant differences between the *tuned* and *global* approaches.

Figure 28 shows the F_{1score}-Threshold curve for the proposed `PD3net` algorithm and the different datasets. We also include the average curve (labeled "Combined F1"), from which a single threshold of 0.54 was selected, as the one with the maximum average F_{1score}. The fact that this best threshold is located around the middle of the threshold span also supports the robustness of the proposed strategy.

4.5. *Computational Performance Evaluation*

Regarding the practical evaluation of our proposal computational complexity, we first experimentally calculated the complexity of the network using the Tensorflow profiling tools. We expressed the complexity of the network using FLOPS (Floating Operations Per Second), estimating a number of 55,6 MFLOPS for our proposal, which is actually a very low number of operations as compared with typical deep learning

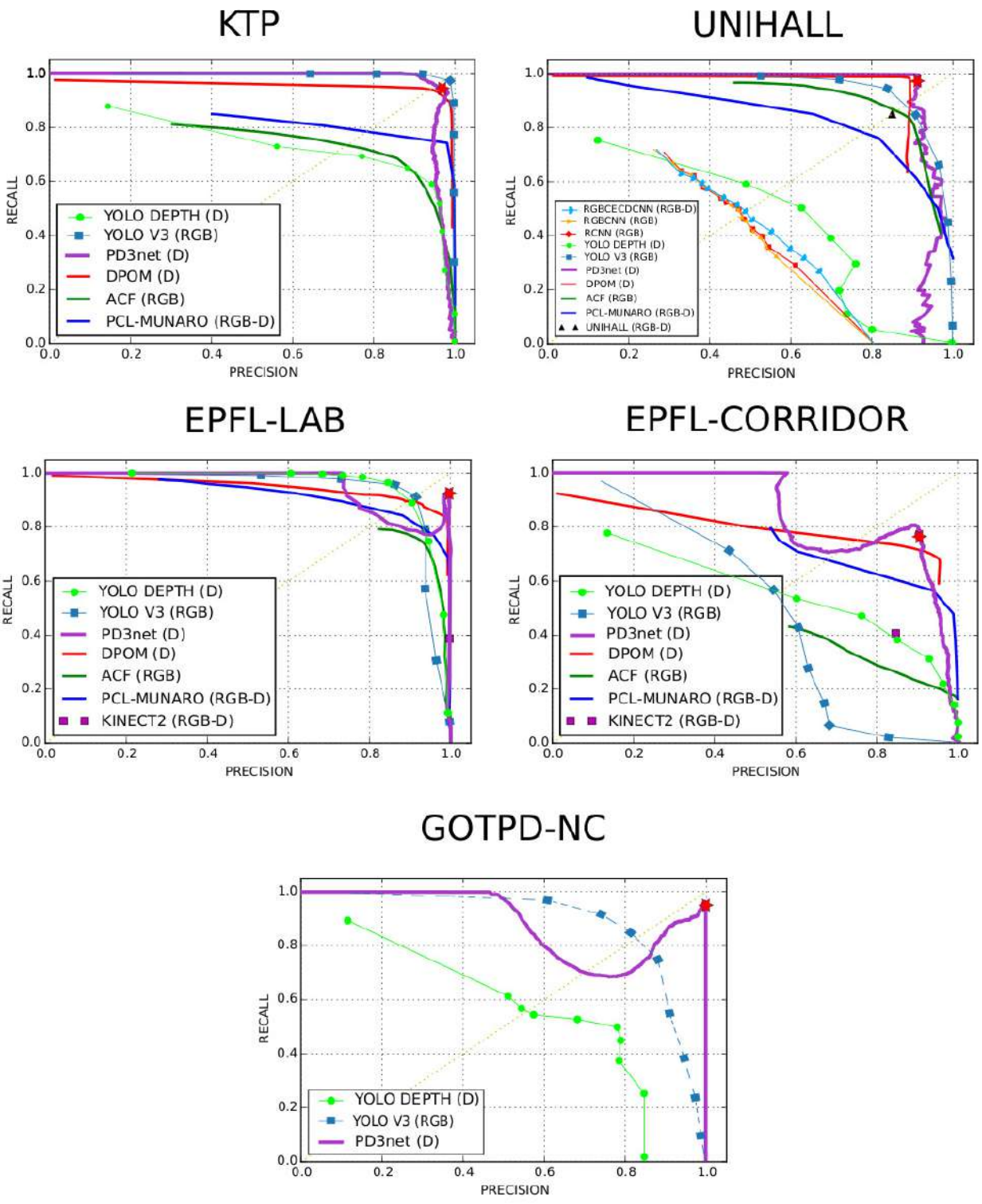


Figure 27: *Precision-Recall* curve comparison using a single global network model and a single global threshold for all datasets.

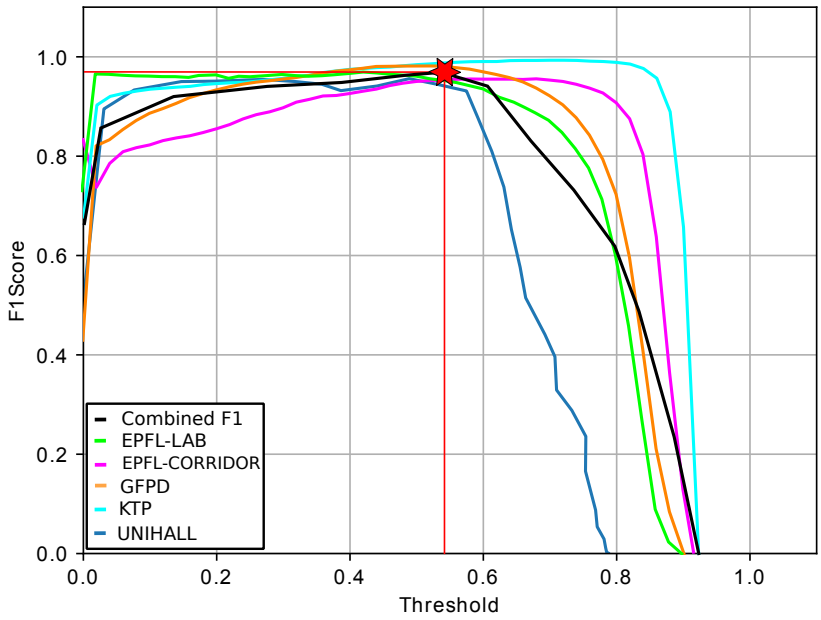


Figure 28: F_{1score}-Threshold results in the experiments using a single global network model and a single global threshold for all datasets.

image processing architectures (Bianco et al., 2018).

We also evaluated the average frame rate of the system, which is 42 FPS (frames per second), benchmarked on a conventional GNU/Linux desktop PC, with a Processor Intel®Core(TM) i7-6700K CPU @ 4.00 GHz with 64 GB of RAM, and an NVIDIA GTX-1080 TI GPU.

This high frame rate implies the possibility of running our system in real time, and it is worth mentioning that its computational complexity does not depend on the number of people in the scene.

5. Conclusions

This work proposes a new people detection method, based on an efficient convolutional neural network architecture, that only uses depth information. The article includes a full description of the network architecture, and extensive evaluation and comparison with a wide range of different state-of-the-art methods for people detection, considering both classical and DNN-based strategies. Our experimental evaluation has been carried out on five different RGB-D image datasets, using different depth sensor technologies (ToF, active stereo, or structured light), and adopting a rigorous experimental evaluation procedure. The scenes used for training and those used for evaluation are as realistic as possible, including a wide variety in what respect to the number of users, occlusion conditions, and background content. We have evaluated the proposed method networks and detection thresholds tuned for each of the databases. We also have tested a

shared neural model with a single detection threshold for all of them, accompanied by a detailed analysis of the results for both approaches.

In most of the evaluated databases, our depth only proposal achieved top performance compared to state-of-the-art methods, including modern RGB DNN-based methods YOLO-V3 or classical methods, such as DPOM. Our system obtains accurate results in challenging scenarios, with a large number of users and severe occlusions. Considering the average overall performance, *i.e.* by averaging the results across all databases, our method clearly surpasses the rest of the approaches with statistically significant differences. This result is especially relevant since the recording conditions across all datasets are very different, even when using a shared network and threshold. Our results show that our architecture and training method, based on fine-tuning a system trained with synthetic data, leads to a high degree of generalization. Therefore, our approach has a potentially high impact on many practical real applications, where a robust, general, and accurate people detection method is needed.

Appendix A. Tables with the detailed information on the experimental work

Regarding the data partition approach for the training and testing subsets, Table A.10 extends the information included in Table 5. It provides full details on the total number of frames (column #framesFull), the total number of people labeled in the ground truth in these frames (column #PeopleFull), and the partition statistics for the training subsets (columns #framesTrain and #PeopleTrain) and for the testing subsets (columns #PeopleTest and #framesTest). Table A.10 also provides details of the used sequences: Total numbers are shown right-aligned, and when the partition involved several sequences, the corresponding data for individual sequences is shown left-aligned. Table A.10 also provides the percentages corresponding to these subsets below the accumulated totals for the #PeopleTrain and #PeopleTest columns to give an idea of their relative sizes.

Table A.11 shows the results of RCNN, RGBCNN, RGBCECDCNN on the UNIHALL dataset that were excluded in Table 8 due to their low performance. They are included here for completeness.

Table A.12 show the weighted average results of of RCNN, RGBCNN, RGBCECDCNN when evaluating all the available datasets using all the available algorithms. They were excluded in Table 9 due to their low performance. They are included here for completeness.

Table A.10: Data partition details.

	Sequence	#framesFull	#PeopleFull	#FramesTrain	#PeopleTrain	#FramesTest	#PeopleTest
GESDPD		**22000**	**54215**	**17600**	**36063 (67%)**	**4400**	**18152 (33%)**
GFPD	1	1690	4462	1690	4462		
	2	900	3760	900	3760		
	3	1090	4337			1090	4337
	4	500	1268	500	1268		
	Total GFPD	**4180**	**13827**	**3090**	**9490 (69%)**	**1090**	**4337 (31%)**
EPFL-LAB	1	**920**	**2287**	**600**	**1385 (61%)**	**320**	**902 (39%)**
EPFL-CORRIDOR	20141008_141829_00	390	899	390	899		
	20141008_1414_30_00	420	813	420	813		
	20141008_141913_00	396	1886	396	1886		
	20141008_141537_00	430	853			430	853
	20141008_141537_00	1100	3581			1100	3581
	Total EPFL-CORRIDOR	**2736**	**8032**	**1206**	**3598 (45%)**	**1530**	**4434 (55%)**
KTP	ROTATION	2200	3299	2200	3299		
	STILL	2100	3420			2100	3420
	Total KTP	**4300**	**6719**	**2200**	**3299 (49%)**	**2100**	**3420 (51%)**
UNIHALL	mensa_seq0_1.1	**2900**	**2979**	**1400**	**2054 (69%)**	**1500**	**925 (31%)**
ALL Datasets	**Total ALL Datasets**	**37036**	**88059**	**26096**	**55889 (63%)**	**10940**	**32170 (37%)**

Table A.11: Performance results of RCNN, RGBCNN, RGBCECDCNN on the UNIHALL dataset, previously excluded in Table 8. Results for the "tuned" and "global" versions for the PD3net proposal (P = *Precision*, R = *Recall* are also shown for comparison.

	GFPD			KTP			UNIHALL			EPFL-LAB			EPFL-CORRIDOR		
	P	R	F_{1score}	P	R	F_{1score}	P	R	F_{1score}	P	R	F_{1score}	P	R	F_{1score}
PD3net tuned	99.7	96.36	98.0 ± 0.20	96.2	96.3	96.3 ± 0.64	92.5	99.2	95.7 ± 1.30	98.8	94.5	96.6 ± 1.18	90.3	80.1	84.9 ± 1.05
PD3net global	100.0	95.9	97.9 ± 0.21	95.4	95.1	95.3 ± 0.71	91.2	97.3	94.2 ± 1.51	99.5	92.5	95.9 ± 1.30	90.9	76.1	82.8 ± 1.11
RCNN (Girshick et al., 2013)							50.1	51.4	50.7 ± 3.22						
RGBCNN (Zhou et al., 2017b)							49.7	51.2	50.4 ± 3.22						
RGBCECDCNN (Zhou et al., 2017b)							52.3	52.3	52.3 ± 3.22						

Table A.12: Average weighted results of RCNN, RGBCNN, RGBCECDCNN using all the available datasets, previously excluded in Table 8. Results for the "tuned" and "global" versions for the PD3net proposal are also shown for comparison purposes.

	Precision	*Recall*	$F_{1\,score}$
PD3net tuned	97.51	93.80	95.62 ± 0.24
PD3net global	97.68	92.59	95.07 ± 0.25
RCNN (Girshick et al., 2013)	50.10	51.40	50.74 ± 3.22
RGBCNN (Zhou et al., 2017b)	49.70	51.20	50.44 ± 3.22
RGBCECDCNN (Zhou et al., 2017b)	52.30	52.30	52.30 ± 3.22

Appendix B. Sample image based results

In this section we include some sample image results when applying our proposal to the different datasets.

Figure B.29: Qualitative example of our proposal on the GFPD dataset (6 people in the ground truth, 6 correct detections).

Figure B.30: Qualitative example of our proposal on the KTP dataset (5 people in the ground truth, 5 correct detections).

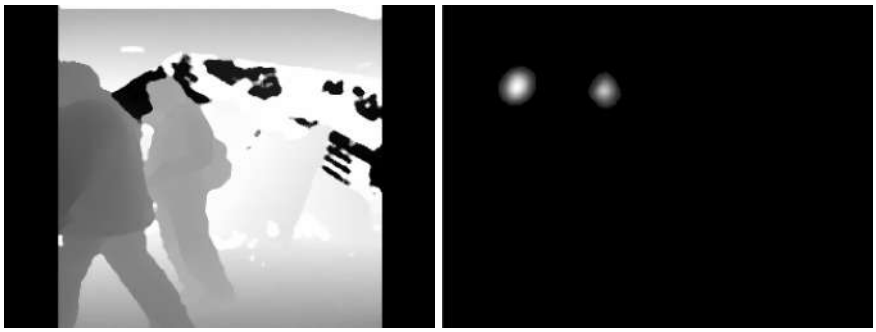


Figure B.31: Qualitative example of our proposal on the UNIHALL dataset (2 people in the ground truth, 2 correct detections).

Figure B.32: Qualitative example of our proposal on the EPFL-LAB dataset (2 people in the ground truth, 2 correct detections).

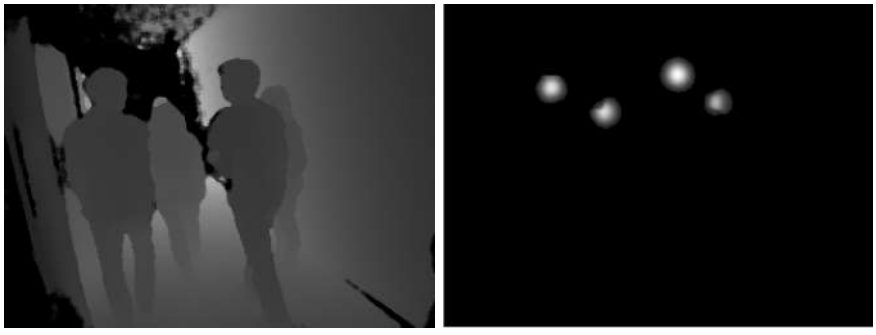


Figure B.33: Qualitative example of evaluation in EPFL-CORRIDOR (4 people in the ground truth, correct detections).

Acknowledgments

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness under projects HEIMDAL-UAH (TIN2016-75982-C2-1-R), ARTEMISA (TIN2016-80939-R), by the Spanish Ministry of Science and Innovation under projects EYEFUL (PID2020-113118RB-C31) and ATHENA (PID2020-115995RB-I00), and by the University of Alcalá under projects ACERCA (CCG2018/EXP-019), ACUFANO (CCG19/IA-024) and ARGOS (CCG20/IA-043).

References

Aguilar, W. G., Luna, M. A., Moya, J. F., Abad, V., Ruiz, H., Parra, H., & Lopez, W. (2017). Cascade classifiers and saliency maps based people detection. In L. T. De Paolis, P. Bourdot, & A. Mongelli (Eds.), *Augmented Reality, Virtual Reality, and Computer Graphics* (pp. 501–510). Cham: Springer International Publishing.

Badrinarayanan, V., Kendall, A., & Cipolla, R. (2015). SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *CoRR*, *abs/1511.00561*. URL: http://arxiv.org/abs/1511.00561. arXiv:1511.00561.

Bagautdinov, T., Fleuret, F., & Fua, P. (2015). Probability occupancy maps for occluded depth images. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2829–2837). doi:10.1109/CVPR.2015.7298900.

Bak, S., San Biagio, M., Kumar, R., Murino, V., & Brémond, F. (2017). Exploiting feature correlations by brownian statistics for people detection and recognition. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *47*, 2538–2549. doi:10.1109/TSMC.2016.2531658.

Barabas, J., Bednar, T., & Vychlopen, M. (2019). Kinect-based platform for movement monitoring and fall-detection of elderly people. In *2019 12th International Conference on Measurement* (pp. 199–202). IEEE.

Bektüzün, E., Küçüksöz, Y. S., & Elif Karslıgil, M. (2013). Real time tracking and detection of unusual circumstances of elderly people with rgb-d camera. In *2013 21st Signal Processing and Communications Applications Conference (SIU)* (pp. 1–5). doi:10.1109/SIU.2013.6531460.

Bianco, S., Cadene, R., Celona, L., & Napoletano, P. (2018). Benchmark analysis of representative deep neural network architectures. *IEEE access*, *6*, 64270–64277.

Blender Online Community (2018). Blender - a 3D modelling and rendering package. URL: http://www.blender.org.

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. arXiv:2004.10934.

Cao, Y., Shen, C., & Shen, H. T. (2017). Exploiting depth from single monocular images for object detection and semantic segmentation. *IEEE Transactions on Image Processing*, *26*, 836–846. doi:10.1109/TIP.2016.2621673.

Chan, A., Liang, Z.-S., & Vasconcelos, N. (2008). Privacy preserving crowd monitoring: Counting people without people models or tracking. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on* (pp. 1–7). doi:10.1109/CVPR.2008.4587569.

Chen, T.-Y., Chen, C.-H., Wang, D.-J., & Kuo, Y.-L. (2010). A people counting system based on face-detection. In *Genetic and Evolutionary Computing (ICGEC), 2010 Fourth International Conference on* (pp. 699–702). doi:10.1109/ICGEC.2010.178.

Choi, W., Pantofaru, C., & Savarese, S. (2012). A general framework for tracking multiple people from a moving camera. *IEEE transactions on pattern analysis and machine intelligence*, *35*, 1577–1591.

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)* (pp. 886–893). Ieee volume 1.

Dan, B.-K., Kim, Y.-S., Suryanto, Jung, J.-Y., & Ko, S.-J. (2012). Robust people counting system based on sensor fusion. *Consumer Electronics, IEEE Transactions on*, *58*, 1013–1021. doi:10.1109/TCE.2012.6311350.

Del Pizzo, L., Foggia, P., Greco, A., Percannella, G., & Vento, M. (2016). Counting people by RGB or depth overhead cameras. *Pattern Recognition Letters*, *81*, 41–50. URL: https://doi.org/10.1016/j.patrec.2016.05.033. doi:10.1016/j.patrec.2016.05.033.

Dollár, P., Appel, R., & Kienzle, W. (2012). Crosstalk cascades for frame-rate pedestrian detection. (pp. 645–659). doi:10.1007/978-3-642-33709-3_46.

Dollár, P., Tu, Z., Perona, P., & Belongie, S. (2009). Integral channel features. doi:10.5244/C.23.91.

Du, X., Lin, T.-Y., Jin, P., Ghiasi, G., Tan, M., Cui, Y., Le, Q. V., & Song, X. (2019). SpineNet: Learning scale-permuted backbone for recognition and localization. arXiv:1912.05027.

Enzweiler, M., Eigenstetter, A., Schiele, B., & Gavrila, D. M. (2010). Multi-cue pedestrian classification with partial occlusion handling. In *2010 IEEE computer society conference on computer vision and pattern recognition* (pp. 990–997). IEEE.

Fuentes-Jiménez, D., Casillas-Pérez, D., Pizarro-Pérez, D., Collins, T., & Bartoli, A. (2018). Deep shape-from-template: Wide-baseline, dense and fast registration and deformable reconstruction from a single image. *CoRR*, *abs/1811.07791*. URL: http://arxiv.org/abs/1811.07791. arXiv:1811.07791.

Fuentes-Jimenez, D., Gutierrez, C. L., Guarasa, J. M., Luna, C., & Pizarro, D. (2020). GFPD geintra frontal person detection dataset. URL: https://www.kaggle.com/dsv/915669. doi:10.34740/KAGGLE/DSV/915669.

Fuentes-Jimenez, D., Losada-Gutierrez, C., & Martín-Lopez, R. (2019a). GESDPD GEintra synthetic depth person detection dataset. URL: https://www.kaggle.com/dsv/915718. doi:10.34740/KAGGLE/DSV/915718.

Fuentes-Jimenez, D., Martin-Lopez, R., Losada-Gutierrez, C., Casillas-Perez, D., Macias-Guarasa, J., Luna, C. A., & Pizarro, D. (2019b). DPDnet: A robust people detector using deep learning with an overhead depth camera. *Expert Systems with Applications*, (p. 113168). URL: http://www.sciencedirect.com/science/article/pii/S0957417419308851. doi:https://doi.org/10.1016/j.eswa.2019.113168.

Galăík, F., & Gargalík, R. (2013). Real-time depth map based people counting. In *15th International Conference on Advanced Concepts for Intelligent Vision Systems - Volume 8192* ACIVS 2013 (pp. 330–341). Berlin, Heidelberg: Springer-Verlag. doi:10.1007/978-3-319-02895-8_30.

Gavriilidis, A., Velten, J., Tilgner, S., & Kummert, A. (2018). Machine learning for people detection in guidance functionality of enabling health applications by means of cascaded SVM classifiers. *Journal of the Franklin Institute*, *355*, 2009 – 2021. URL: http://www.sciencedirect.com/science/article/pii/S0016003217305252. doi:https://doi.org/10.1016/j.jfranklin.2017.10.008. Special Issue on Recent advances in machine learning for signal analysis and processing.

G.Ghiasi, T.Lin, & Q.V.Le (2019). NAS-FPN: Learning scalable feature pyramid architecture for object

detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 7029–7038).

Girshick, R. B., Donahue, J., Darrell, T., & Malik, J. (2013). Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, *abs/1311.2524*.

Golyanik, V., Shimada, S., Varanasi, K., & Stricker, D. (2018). HDM-Net: Monocular non-rigid 3D reconstruction with learned deformation model. *CoRR*, *abs/1803.10193*. URL: http://arxiv.org/abs/1803.10193. arXiv:1803.10193.

Guera, D., & Delp, E. J. (2018). Deepfake video detection using recurrent neural networks. *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, (pp. 1–6).

Han, X., He, T., Ong, Y.-S., & Zhong, Y. (2020). Precise object detection using adversarially augmented local/global feature fusion. *Engineering Applications of Artificial Intelligence*, *94*, 103710. URL: http://www.sciencedirect.com/science/article/pii/S0952197620301330. doi:https://doi.org/10.1016/j.engappai.2020.103710.

Hayashi, T., Nishida, M., Kitaoka, N., & Takeda, K. (2015). Daily activity recognition based on dnn using environmental sound and acceleration signals. In *2015 23rd European Signal Processing Conference (EUSIPCO)* (pp. 2306–2310).

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 2016, Las Vegas, NV, USA, June 27-30, 2016* (pp. 770–778). doi:10.1109/CVPR.2016.90.

Hu, T., Zhang, H., Zhu, X., Clunis, J., & Yang, G. (2018). Depth sensor based human detection for indoor surveillance. *Future Generation Computer Systems*, *88*, 540 – 551. URL: http://www.sciencedirect.com/science/article/pii/S0167739X18308537. doi:https://doi.org/10.1016/j.future.2018.05.083.

Intel (). Intel realsense D435 product. https://www.intelrealsense.com/depth-camera-d435/.

Jeong, C. Y., Choi, S., & Han, S. W. (2013). A method for counting moving and stationary people by interest point classification. In *Image Processing (ICIP), 2013 20th IEEE International Conference on* (pp. 4545–4548). doi:10.1109/ICIP.2013.6738936.

Keskar, N. S., & Socher, R. (2017). Improving generalization performance by switching from adam to SGD. *CoRR*, *abs/1712.07628*. URL: http://arxiv.org/abs/1712.07628. arXiv:1712.07628.

Kim, H., Garrido, P., Tewari, A., Xu, W., Thies, J., Nießner, N., Pérez, P., Richardt, C., Zollhöfer, M., & Theobalt, C. (2018). Deep Video Portraits. *ACM Transactions on Graphics 2018 (TOG)*, .

Kinect-SDK (2014). Kinect for windows SDK 2.0. http://www.microsoft.com/en-us/kinectforwindows/.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR*, *abs/1412.6980*. URL: http://arxiv.org/abs/1412.6980. arXiv:1412.6980.

Lee, K.-D., Nam, M. Y., Chung, K.-Y., Lee, Y.-H., & Kang, U.-G. (2013). Context and profile based cascade classifier for efficient people detection and safety care system. *Multimedia Tools and Applications*, *63*, 27–44.

Li, S., Han, P., Bu, S., Tong, P., Li, Q., Li, K., & Wan, G. (2020). Change detection in images using shape-aware siamese convolutional network. *Engineering Applications of Artificial Intelligence*, *94*, 103819. URL: http://www.sciencedirect.com/science/article/pii/S0952197620301950. doi:https://doi.org/10.1016/j.engappai.2020.103819.

Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., & Dollár, P. (2014). Microsoft COCO: Common objects in context. arXiv:1405.0312.

Liu, J., Liu, Y., Zhang, G., Zhu, P., & Chen, Y. Q. (2015). Detecting and tracking people in real time with RGB-D camera. *Pattern Recognition Letters*, *53*, 16 – 23. URL: http://www.sciencedirect.com/science/article/pii/S016786551400302X. doi:10.1016/j.patrec.2014.09.013.

Luber, M., Spinello, L., & Arras, K. O. (2011). People tracking in RGB-D data with on-line boosted target models. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2011, San Francisco, CA, USA, September 25-30, 2011* (pp. 3844–3849). IEEE. URL: https://doi.org/10.1109/IROS.2011.6095075. doi:10.1109/IROS.2011.6095075.

Luna, C. A., Losada-Gutierrez, C., Fuentes-Jimenez, D., Fernandez-Rincon, A., Mazo, M., & Macias-Guarasa, J. (2017). Robust people detection using depth information from an overhead time-of-flight camera. *Expert Syst. Appl.*, *71*, 240–256. doi:10.1016/j.eswa.2016.11.019.

Moro, A., Wakabayashi, J., Toda, T., & Umeda, K. (2018). A framework for human recognition and counting in restricted area for video surveillance. In *Intelligent Environments (Workshops)* (pp. 139–148).

Munaro, M., & Menegatti, E. (2014). Fast RGB-D people tracking for service robots. *Autonomous Robots*, *37*. doi:10.1007/s10514-014-9385-0.

Odena, A., Dumoulin, V., & Olah, C. (2016). Deconvolution and checkerboard artifacts. *Distill*, *1*. doi:10.23915/distill.00003.

Ramanan, D., Forsyth, D. A., & Zisserman, A. (2006). Tracking People by Learning Their Appearance. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *29*, 65–81. doi:10.1109/tpami.2007.250600.

Redmon, J., Divvala, S. K., Girshick, R. B., & Farhadi, A. (2015). You only look once: Unified, real-time object detection. *CoRR*, *abs/1506.02640*.

Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. *CoRR*, *abs/1804.02767*. URL: http://arxiv.org/abs/1804.02767. arXiv:1804.02767.

Ren, X., Du, S., & Zheng, Y. (2017). Parallel RCNN: A deep learning method for people detection using rgb-d images. In *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)* (pp. 1–6). doi:10.1109/CISP-BMEI.2017.8302069.

Romera, E., Álvarez, J. M., Bergasa, L. M., & Arroyo, R. (2017). Efficient convnet for real-time semantic segmentation. In *2017 IEEE Intelligent Vehicles Symposium (IV)* (pp. 1789–1794). doi:10.1109/IVS.2017.7995966.

Ruiz-Santaquiteria, J., Bueno, G., Deniz, O., Vallez, N., & Cristobal, G. (2020). Semantic versus instance segmentation in microscopic algae detection. *Engineering Applications of Artificial Intelligence*, *87*, 103271. URL: http://www.sciencedirect.com/science/article/pii/S0952197619302398. doi:https://doi.org/10.1016/j.engappai.2019.103271.

Shin, H.-c., Roth, H., Gao, M., Lu, L., Xu, Z., Nogues, I., Yao, J., Mollura, D., & Summers, R. (2016). Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE Transactions on Medical Imaging*, *35*, 1285–1298. doi:10.1109/TMI.2016.2528162.

Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., & Blake, A. (2011). Real-time human pose recognition in parts from single depth images. (pp. 1297–1304). volume 56. doi:10.1109/CVPR.2011.5995316.

Silberman, N., Hoiem, D., Kohli, P., & Fergus, R. (2012). Indoor segmentation and support inference from rgbd images. In *European conference on computer vision* (pp. 746–760). Springer.

Spinello, L., & Arras, K. O. (2011). People detection in rgb-d data. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 3838–3843). IEEE.

Spinello, L., & Arras, K. O. (2011). People detection in RGB-D data. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 3838–3843). doi:10.1109/IROS.2011.6095074.

Stahlschmidt, C., Gavriilidis, A., Velten, J., & Kummert, A. (2013). People detection and tracking from a top-view position using a time-of-flight camera. In A. Dziech, & A. Czyazwski (Eds.), *Multimedia Communications, Services and Security* (pp. 213–223). Springer Berlin Heidelberg volume 368 of *Communications in Computer and Information Science*. doi:10.1007/978-3-642-38559-9_19.

Stahlschmidt, C., Gavriilidis, A., Velten, J., & Kummert, A. (2014). Applications for a people detection and tracking algorithm using a time-of-flight camera. *Multimedia Tools and Applications*, (pp. 1–18). doi:10.1007/s11042-014-2260-3.

Stewart, R., Andriluka, M., & Ng, A. Y. (2016). End-to-end people detection in crowded scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2325–2333).

Susperregi, L., Arruti, A., Jauregi, E., Sierra, B., Martínez-Otzeta, J., Lazkano, E., & Ansuategui, A. (2013). Fusing multiple image transformations and a thermal sensor with kinect to improve person detection ability. *Engineering Applications of Artificial Intelligence*, *26*, 1980 – 1991. URL: http://www.sciencedirect.com/science/article/pii/S0952197613000791. doi:https://doi.org/10.1016/j.engappai.2013.04.013.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2015). Rethinking the inception architecture for computer vision. *CoRR*, *abs/1512.00567*. URL: http://arxiv.org/abs/1512.00567. arXiv:1512.00567.

Tao, L., Burghardt, T., Hannuna, S., Camplani, M., Paiement, A., Damen, D., Mirmehdi, M., & Craddock, I. (2015). A comparative home activity monitoring study using visual and inertial sensors. In *2015 17th*

International Conference on E-health Networking, Application & Services (HealthCom) (pp. 644–647). IEEE.

Tian, Y., Luo, P., Wang, X., & Tang, X. (2015). Pedestrian detection aided by deep learning semantic tasks. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 5079–5087). doi:10.1109/CVPR.2015.7299143.

Uijlings, J. R. R., van de Sande, K. E. A., Gevers, T., & Smeulders, A. W. M. (2013). Selective search for object recognition. *International Journal of Computer Vision*, *104*, 154–171.

Vera, P., Monjaraz, S., & Salas, J. (2016). Counting pedestrians with a zenithal arrangement of depth cameras. *Machine Vision and Applications*, *27*, 303–315. URL: https://doi.org/10.1007/s00138-015-0739-1. doi:10.1007/s00138-015-0739-1.

Verma, N. K., Dev, R., Maurya, S., Dhar, N. K., & Agrawal, P. (2019). People counting with overhead camera using fuzzy-based detector. In N. K. Verma, & A. K. Ghosh (Eds.), *Computational Intelligence: Theories, Applications and Future Directions - Volume I* (pp. 589–601). Singapore: Springer Singapore.

Villamizar, M., Martínez-González, A., Canévet, O., & Odobez, J.-M. (2018). Watchnet: Efficient and depth-based network for people detection in video surveillance systems. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)* (pp. 1–6). IEEE.

Wang, C., & Zhao, Y. (2017). Multi-layer proposal network for people counting in crowded scene. In *2017 10th International Conference on Intelligent Computation Technology and Automation (ICICTA)* (pp. 148–151). doi:10.1109/ICICTA.2017.40.

Wang, S., Chen, L., Zhou, Z., Sun, X., & Dong, J. (2016). Human fall detection in surveillance video based on PCANet. *Multimedia tools and applications*, *75*, 11603–11613.

Wang, Y., Lian, H., Chen, P., & Lu, Z. (2014). Counting people with support vector regression. In *2014 10th International Conference on Natural Computation (ICNC)* (pp. 139–143). doi:10.1109/ICNC.2014.6975824.

Xu, B., Wang, N., Chen, T., & Li, M. (2015). Empirical evaluation of rectified activations in convolutional network. *CoRR*, *abs/1505.00853*. URL: http://arxiv.org/abs/1505.00853. arXiv:1505.00853.

Zhang, G., Pan, Y., Zhang, L., & Tiong, R. L. K. (2020). Cross-scale generative adversarial network for crowd density estimation from images. *Engineering Applications of Artifi-*

cial Intelligence, *94*, 103777. URL: http://www.sciencedirect.com/science/article/pii/S0952197620301743. doi:https://doi.org/10.1016/j.engappai.2020.103777.

Zhang, G., Tian, L., Liu, Y., Liu, J., Liu, X. A., Liu, Y., & Chen, Y. Q. (2016a). Robust real-time human perception with depth camera. In *ECAI* (pp. 304–310).

Zhang, L., Wu, X., & Luo, D. (2015). Human activity recognition with HMM-DNN model. In *2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC)* (pp. 192–197). IEEE.

Zhang, S., Benenson, R., Omran, M., Hosang, J., & Schiele, B. (2016b). How far are we from solving pedestrian detection? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1259–1267). doi:10.1109/CVPR.2016.141.

Zhang, S., Benenson, R., Omran, M., Hosang, J., & Schiele, B. (2018). Towards reaching human performance in pedestrian detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*, 973–986. doi:10.1109/TPAMI.2017.2700460.

Zhang, X., Yan, J., Feng, S., Lei, Z., Yi, D., & Li, S. (2012). Water filling: Unsupervised people counting via vertical Kinect sensor. In *Advanced Video and Signal-Based Surveillance (AVSS), 2012 IEEE Ninth International Conference on* (pp. 215–220). doi:10.1109/AVSS.2012.82.

Zhao, J., Zhang, G., Tian, L., & Chen, Y. Q. (2017). Real-time human detection with depth camera via a physical radius-depth detector and a CNN descriptor. In *2017 IEEE International Conference on Multimedia and Expo (ICME)* (pp. 1536–1541). doi:10.1109/ICME.2017.8019323.

Zhou, K., Paiement, A., & Mirmehdi, M. (2017a). Detecting humans in RGB-D data with CNNs. In *2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA)* (pp. 306–309). doi:10.23919/MVA.2017.7986862.

Zhou, K., Paiement, A., & Mirmehdi, M. (2017b). Detecting humans in rgb-d data with cnns. In *2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA)* (pp. 306–309). doi:10.23919/MVA.2017.7986862.

Zhu, L., & Wong, K.-H. (2013). Human tracking and counting using the kinect range sensor based on adaboost and kalman filter. In *International Symposium on Visual Computing* (pp. 582–591). Springer. doi:10.1007/978-3-642-41939-3_57.