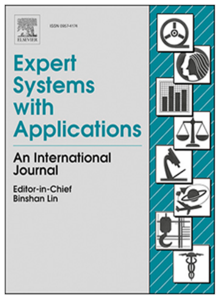
Journal Pre-proof



Multi-camera vehicle counting using edge-AI

Luca Ciampi, Claudio Gennaro, Fabio Carrara, Fabrizio Falchi,
Claudio Vairo, Giuseppe Amato

PII: S0957-4174(22)01171-X
DOI: https://doi.org/10.1016/j.eswa.2022.117929
Reference: ESWA 117929

To appear in: *Expert Systems With Applications*

Received date : 4 June 2021
Revised date : 16 May 2022
Accepted date : 19 June 2022

Please cite this article as: L. Ciampi, C. Gennaro, F. Carrara et al., Multi-camera vehicle counting using edge-AI. *Expert Systems With Applications* (2022), doi: https://doi.org/10.1016/j.eswa.2022.117929.

Highlights

- Smart mobility is crucial for smart cities and traffic-related issues.
- We introduce a multi-camera system able to count cars from images of parking areas.
- We combine a deep learning-based technique and a decentralized geometric-based approach.
- All the algorithms run on the edge devices reducing the traffic on the network.
- Our solution benefits from redundant information from different data sources.

ORCID Information

Below the ORCID information about the authors:

- Luca Ciampi: https://orcid.org/0000-0002-6985-0439
- Claudio Gennaro: https://orcid.org/0000-0002-3715-149X
- Fabio Carrara: https://orcid.org/0000-0001-5014-5089
- Fabrizio Falchi: https://orcid.org/0000-0001-6258-5313
- Claudio Vairo: https://orcid.org/0000-0003-2740-4331
- Giuseppe Amato: https://orcid.org/0000-0003-0171-4315

CRediT author statement

Luca Ciampi: Writing - Conceptualization, Software, Validation, Writing - Original Draft, Writing - Review & Editing

Claudio Gennaro: Conceptualization, Writing - Review & Editing, Funding acquisition

Fabio Carrara: Conceptualization, Investigation, Writing - Review & Editing

Fabrizio Falchi: Writing - Review & Editing, Supervision, Funding acquisition

Claudio Vairo: Writing - Review & Editing, Supervision

Giuseppe Amato: Writing - Review & Editing, Supervision, Funding acquisition

Multi-Camera Vehicle Counting Using Edge-AI

Revised Manuscript (clean)

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

[a] Institute of Information Science and Technologies of the National Research Council of Italy (ISTI-CNR), via G. Moruzzi 1 - 56124, Pisa, Italy

Corresponding Author:
Luca Ciampi
Institute of Information Science and Technologies of the National Research Council of Italy (ISTI-CNR), via G. Moruzzi 1 - 56124, Pisa, Italy
Tel: +39 050 6213054
Email: luca.ciampi@isti.cnr.it

Abstract

This paper presents a novel solution to automatically count vehicles in a parking lot using images captured by smart cameras. Unlike most of the literature on this task, which focuses on the analysis of *single* images, this paper proposes the use of multiple visual sources to monitor a wider parking area from different perspectives. The proposed multi-camera system is capable of automatically estimating the number of cars present in the *entire* parking lot directly on board the edge devices. It comprises an on-device deep learning-based detector that locates and counts the vehicles from the captured images and a decentralized geometric-based approach that can analyze the inter-camera shared areas and merge the data acquired by all the devices. We conducted the experimental evaluation on an extended version of the *CNRPark-EXT* dataset, a collection of images taken from the parking lot on the campus of the National Research Council (CNR) in Pisa, Italy. We show that our system is robust and takes advantage of the redundant information deriving from the different cameras, improving the overall performance without requiring any extra geometrical information of the monitored scene.
Keywords: Smart Parking, Counting Objects, Edge AI, Counting Vehicles, Smart Mobility, Deep Learning

1. Introduction

Traffic-related issues are constantly increasing, and tomorrow's cities cannot be considered intelligent if they do not enable smart mobility. Smart mobility applications, such as smart parking and road traffic management, are nowadays widely employed worldwide, making our cities more livable and bringing benefits to the cities and, consequently, to our lives.

Images are perhaps the best sensing modality to perceive and assess the flow of vehicles in large areas. Like no other sensing mechanism, city camera networks can monitor large areas while simultaneously providing visual data to AI systems to extract relevant information from this deluge of data. However, this application is often hampered by the massive flow of data that must be sent to central servers or the cloud for processing. On the other hand, edge computing is a recent paradigm that promotes the decentralization of data processing to the border, i.e., where the data are generated, thus reducing the traffic on the network and the pressure on central servers. No wonder that combination of recent Computer Vision deep learning-based techniques and the edge computing paradigm is an emerging trend, as witnessed, for example, by Khan et al. (2019) that tackles the face recognition task or by Amato et al. (2019b); Ciampi et al. (2020a) that instead can detect people directly onboard surveillance cameras. Nonetheless, this promising paradigm brings along with it also some new challenges related to the limited computational resources on the disposable edge devices and also concerning security inside IoT networks (Ujjan et al., 2020).

In this work, we tackle the problem of estimating the number of vehicles present in a parking lot using images captured by smart cameras. Whereas classic car counting solutions are sensor-based (e.g., entrance-level photocells, per-space ground sensors), vision-based solutions provide several advantages, such as a) flexibility, as cameras can adapt to more challenging configurations of parking spaces (e.g., undelimited parking lots with non-fixed spaces), b) lower hardware and maintenance cost, as smart cameras can cost few tens of dollars while each monitoring multiple parking spaces, and c) being multi-purpose, as the same hardware can be used to perform additional tasks (e.g., surveillance). However, this vision-based counting task is challenging as the process of understanding the captured images faces many problems, such as shadows, light variation, weather conditions, and inter-object occlusions. Although most of the existing works concerning the vehicle counting task focus on the analysis of *single* images, in many real-world scenarios, one can benefit from using multiple cameras to monitor the same parking lot from different perspectives and viewpoints. Furthermore, multiple neighboring cameras can also help cover a wider area. At the same time, such an approach introduces issues related to merging the knowledge extracted from the single cameras with partially overlapping fields of views (FOVs), as shown in Figure 1.

In this paper, we propose a novel solution to improve car counting when scaled up with multi-camera setups. Specifically, we introduce a multi-camera system that estimates the number of cars present in the *entire* parking lot by combining a state-of-the-art Convolutional Neural Network (CNN), which can locate and count vehicles present in images belonging to individual cameras, along with a decentralized geometry-based approach that is responsible for aggregating the data gathered from all the devices. Our solution performs the task directly on the edge devices (i.e., the smart cameras) without using a central server or cloud, consequently reducing the communication overhead. The total count is built exploiting the partial results computed in parallel by the single cameras and propagated through messages. Hence, our system scales better when the number of monitored parking spaces increases. Moreover, our solution does not require any manual intervention or any extra information about the monitored parking area, such as the location of the parking spaces, nor any geometric information about the camera positions in the parking lot. In short, it is a flexible and ready-to-use solution that allows a simple "plug-and-play" insertion of new cameras into the system.

To validate our multi-camera solution, we employed the *CNRPark-EXT* dataset (Amato et al., 2017), a collection of images taken from the parking lot on the campus of the National Research Council (CNR) in Pisa, Italy. The pictures are acquired by multiple cameras having partially overlapping fields of view and describing challenging scenarios with different perspectives, illuminations, weather conditions, and many occlusions. Since the annotations of this dataset concern single images, we extended it by manually labeling a part of it to be consistent with our algorithm that instead considers the entire parking area. We conducted extensive experiments testing the generalization capabilities of the CNN-based technique responsible for detecting vehicles in single images and the effectiveness of our multi-camera algorithm, demonstrating that our system is robust and benefits from the redundant information deriving from

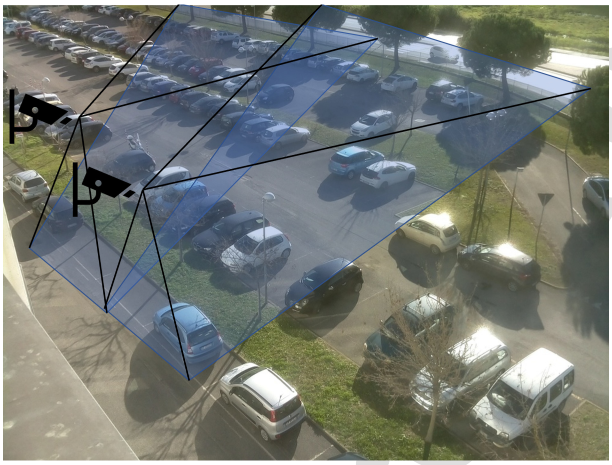


Figure 1: An example of two cameras monitoring the same parking area with partially overlapping fields of view. This redundancy provides robustness and fault-tolerance but also raises the problem of aggregating knowledge extracted from the individual cameras.

the different cameras improving the overall performance.

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

- We introduce a novel multi-camera system able to automatically estimate the number of cars present in the *entire* monitored parking area. It runs directly on the edge devices and combines a deep learning-based detector together with a decentralized technique that exploits the geometry of the captured images.
- We specifically extend the *CNRPark-EXT* dataset (Amato et al., 2017), a collection of images acquired by multiple cameras having partially overlapping fields of views and describing various parking lots. We manually label a subset of it, making it suitable for our considered scenario in which we consider the whole parking area.
- We conduct an experimental evaluation showing that our system is robust, flexible, and can benefit from redundant information from different

cameras while improving overall performance.

We organize the rest of the paper as follows. Section 2 reports other works present in the literature related to our topic. Section 3 describes our multi-camera counting algorithm. Section 4 states the experimental setup, describing the dataset, the metrics, and the implementation details. Section 5 presents and discusses the experiments and the obtained results. Finally, Section 6 concludes the paper with some insights on future directions.

2. Related Work

This section overviews some works related to our, organizing them into two categories. The first one concerns the counting task, while the second regards multi-camera parking lot monitoring systems.

2.1. The counting task

The counting task estimates the number of object instances in still images or video frames (Lempitsky & Zisserman, 2010). This topic has recently attracted much attention due to its inter-disciplinary and widespread applicability and paramount importance for many real-world applications. Examples include counting bacterial cells from microscopic images (Xie et al., 2016; Ciampi et al., 2022), estimating the number of people present at an event (Boominathan et al., 2016; Benedetto et al., 2022), counting animals in ecological surveys to monitor the population of a specific region (Arteta et al., 2016) and evaluating the number of vehicles on a highway or in a car park (Amato et al., 2019a).

Several machine learning-based solutions (especially supervised) have been suggested in the last years. Following the taxonomy adopted in Sindagi & Patel (2018), we can broadly classify existing counting approaches into two categories: counting by regression and counting by detection. Counting by *regression* is a supervised method that tries to establish a direct mapping (linear or not) from the image features to the number of objects present in the scene or a corresponding density map (i.e., a continuous-valued function), skipping the challenging task of detecting instances of the objects (Zhang et al., 2016, 2017; Oñoro-Rubio & López-Sastre, 2016; Ciampi et al., 2020b, 2021). Counting by *detection* is, instead, a supervised approach where we localize instances of the objects, and then we count them (Amato et al., 2018; Ciampi et al., 2018). While regression-based techniques work very well in very crowded scenarios where the single object instances are not well defined due to inter-class and intra-class occlusions, they perform poorly in images with a large perspective and oversized objects. Another remarkable drawback of the regression-based approaches is that they cannot precisely localize the objects present in the scene, eventually providing only a coarse position of the area in which they are distributed.

In this work, we estimate the number of vehicles present in a park area from images collected by smart cameras having large perspectives. The cars close to the cameras are much larger than those far away from them. Therefore, we employ a detection-based method. Furthermore, another reason which led us to discard counting by regression approaches is that we need to know the precise localization (with boundaries) of the detected vehicles. Most of the existing counting solutions do not directly deal with edge computing devices and the consequent constraints due to the limited available computing resources. They use deep learning-based approaches that typically require the use of a GPU and that are computationally expensive. Moreover, they consider the images as single entities. They do not account for the possible benefits of monitoring the same lots from different perspectives or covering a wider parking area with multiple cameras. Instead, our solution runs directly on the edge devices and can estimate the number of vehicles present in the entire parking lot.

2.2. Multi-camera parking lot monitoring

Only a few works addressed parking lot monitoring considering a multi-camera scenario. In Nieto et al. (2019), the authors applied a homography to project the detected vehicles from the plane of each camera to a common plane, where they performed a perspective correction to correct matching between the vehicle detections and the parking spots. Also, the authors in Vítek &

Melničuk (2017) proposed a multi-camera system to classify parking spaces as vacant or occupied. In this solution, the acquired images are processed onboard Raspberry Pi devices. The extracted information about the status of parking spaces is then transmitted to a central server, which evaluates the parking spaces in the overlapping areas. Their algorithm is based on the histogram of oriented gradients (HOG)(Dalal & Triggs, 2005) feature descriptor and support vector machine (SVM) classifier. Since the HOG feature descriptor cannot adequately describe rotated vehicles, the authors have provided a descriptor with additional information about rotation to increase the system accuracy.

However, these solutions rely on prior knowledge of the monitored scene, such as the position of the parking spaces or some geometric information concerning the parking area. For instance, the proposed system in Nieto et al. (2019) requires manually annotating the corners of the parking area and the number of spots. In essence, a preliminary annotation of the new areas and a new training phase of the algorithm are often mandatory operations. Consequently, these techniques are not very flexible. On the other hand, we propose a simple yet effective solution that does not need any extra information about the monitored scene. The smart cameras can automatically localize and count the vehicles present in their field of view, propagating the single results to the other edge devices through messages. A decentralized technique, again running directly on the edge devices, is instead in charge of analyzing and merging these results, exploiting the captured images geometry, and automatically outputs the number of cars present in the entire parking area.

3. Proposed approach

3.1. Overview

In this section, we describe our multi-camera counting algorithm. We based our system on the parallel processing of each of the smart cameras followed by the fusion of their results to estimate the number of vehicles present in the *entire* parking area.

Figure 2 shows an example of our multi-camera counting system, together with its graphical representation. We model our system as a graph G, comprised of n nodes ν_i and one Sink node S, $V = \{\nu_1, \nu_2, \cdots, \nu_n, S\}$. Each node ν_i represents an independent edge device, i.e., a smart camera in our case. Two nodes ν_i and ν_j are considered neighbors if their FOVs overlap. In this case, a directed edge of the graph connects them. Each edge device ν_i can capture images, localize and count the vehicles present in its FOV exploiting a deep learning-based detector, and communicate with its neighboring nodes through messages m_i containing the cars detections. Furthermore, each node ν_i can also run a local counting algorithm in charge of computing partial counting results concerning the estimation of the number of vehicles present in overlapped areas between its FOV and the ones belonging to its neighbors.

The fusion of the partial results is performed by the Sink node S, which is also in charge of providing the final result and synchronizing all the algorithm steps through synchronization signals headed towards the other nodes ν_i. On the other hand, the nodes ν_i can also communicate through messages with the Sink node. Messages can be of two types: i) messages η_i containing the number of cars captured by the node ν_i in its FOV, and ii) messages $\mu_{j,i}$ representing the partial counting estimation related to the overlapping area between two neighboring nodes ν_i and ν_j.

In the following sections, we describe all the steps of our algorithm in detail. First, in Section 3.2, we outline the automatic system initialization performed by the smart cameras themselves, in which they compute the homographic transformations between the scene they are monitoring and the scene observed by the neighboring cameras. Then, in Section 3.3, we describe the CNN-based local counting algorithm that runs on each of the smart cameras and the geometric-based technique helpful for the overlapped areas. Finally, in Section 3.4, we depict the global counting algorithm responsible for the fusion of these individual and partial results, and that finally outputs the number of cars present in the *entire* parking area.

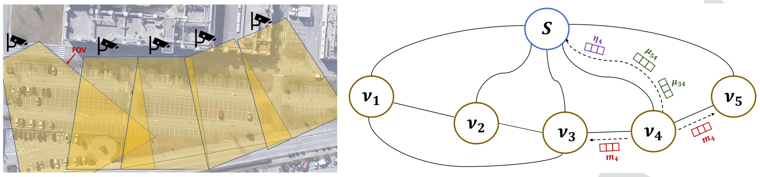


Figure 2: An example of our multi-camera counting system, with $n = 5$ smart cameras. We model it as a graph G, comprised of n nodes ν_i (one for each camera) and one Sink node S, $V = \{\nu_1, \nu_2, \cdots, \nu_n, S\}$. Each node ν_i can capture images, localize and count the vehicles present in its FOV, and communicate with its neighboring nodes through messages m_i containing these detections. Moreover, each node ν_i can run a local counting algorithm in charge of computing partial counting results concerning the overlapped areas between its FOV and the ones belonging to its neighbors, exploiting images geometry. These partial results are sent through messages to the Sink node S, which is responsible for their fusion and provides the final result. Messages to S can be of two types: i) η_i containing the number of cars captured by the node ν_i in its FOV, and ii) $\mu_{j,i}$ representing the partial counting estimation related to the overlapping area between two neighboring nodes ν_i and ν_j.

3.2. Initialization

This step is aimed at *automatically* initializing the system, estimating the geometric relationship between each node (i.e., each scene monitored by a smart camera) and its neighbors. The only hypotheses we impose are i) each smart camera is aware of the IP addresses of its neighbors, i.e., the cameras having the field of view overlapped with its own; ii) the Sink node S is aware of the IP addresses of all the smart cameras belonging to the system.

The Sink node S starts the initialization phase, sending a synchronization signal to the other nodes. Once received, each smart camera captures an image of the scene it monitors and sends it to all its neighbors. Once a smart camera i receives an image from a neighboring camera j, it computes a homographic transformation $H_{j,i}$ between the image j and the image i describing its monitored scene. This allows us to establish a correspondence between the points belonging to the pair of images taken by the two cameras, which will be used subsequently in the algorithm. We formalized the system initialization for a generic node ν_i in the Algorithm 1.

However, finding this homography can be challenging because neighboring cameras can have different angles of view, leading to a perspective distortion between the images captured by them. Given a pair of neighboring nodes ν_i, ν_j, we employ a procedure that starts with finding the SIFT (Lowe, 2004) key-points and feature descriptors of the images i, j captured by the two nodes. Then, we match the two sets of feature descriptors by performing David Lowe's ratio test (Lowe, 2004), and we further filter the matched feature descriptors by keeping only the pairs whose euclidean distance is below a given threshold. Finally, we obtain the homographic transformation by applying the random sample consensus (RANSAC (Fischler & Bolles, 1981)) algorithm to the filtered feature descriptors. All these computations are performed *automatically* without the need of any extra geometric information about the monitored scene, and no manual intervention is needed. Figure 3 shows the concatenation of two neighboring images i and j in which we apply the found homographic matrix to the image i, to have the same perspective as the image j.

Algorithm 1 : Initialization

At each Initialization Signal by S, each node ν_i performs the following steps:

```
1: ReceiveInitSignal()                      ▷ waits the initialization signal from S
2: image_i ← CameraCapture()
3: for each j ∈ J do                        ▷ J is the set of neighboring nodes of node ν_i
4:     SendImage(image_i, ν_j)              ▷ sends image_i to node ν_j
5:     image_j ← ReceiveImage()             ▷ receives image_j from node ν_j
6:     H_{j,i} = ComputeHomography(image_j, image_i)
```

3.3. Local Counting Algorithm

This section describes the local counting algorithm that runs directly onboard the edge devices. It combines a CNN-based counting technique in charge of the localization and the estimation of the number of vehicles present in the acquired single images, i.e., the contents of the messages m_i and the quantities

Figure 3: Example of concatenation of two images using a homographic transformation, where it is also visible the overlapping area between them.

η_i shown in Figure 2, together with a geometric-based approach responsible of estimating the number of vehicles present in the overlapping areas between the nodes and their neighbors, i.e., the quantities $\mu_{j,i}$.

A vehicle counting CNN on the Edge. Each smart camera needs to independently detect and count vehicles from its captured frame. For this step, every approach providing precise localization of the detected vehicles in the pixel space is suitable, and the choice of a particular approach should be guided by resource constraints, e.g., available memory, prediction frequency, or energy consumption, if any. Here, we base our vehicle counting technique on *Mask R-CNN* (He et al., 2017), a popular deep CNN for instance segmentation that operates within the 'recognition using regions paradigm' (Gu et al., 2009). In particular, it extends the *Faster R-CNN* detector (Ren et al., 2017) by adding a branch that outputs a binary mask saying whether or not a given pixel is part of an object. Briefly, a CNN acts as a backbone in the first stage, extracting the input image features. Starting from this feature space, another CNN named Region Proposal Network (RPN) generates region proposals that might contain objects. RPN slices pre-defined region boxes (called anchors) over this space and ranks them, suggesting those most likely containing objects. Once RPN produces the Regions Of Interests (ROIs), they might be of different sizes. Since it is hard to work on features having different sizes, RPN reduces them into the same dimension using the Region of Interest Pooling algorithm. Finally, these fixed-size proposals are processed by two parallel CNN-based branches: one is responsible for classifying and localizing the objects inside them with bounding boxes; the second produces a binary mask that says whether or not a given pixel is part of an object. In the end, given an input image, the network produces per-pixel masks localizing the detected objects together with the associated labels classifying them.

To make our counting solution able to run efficiently directly on the edge devices, we employ, as a backbone, the *ResNet50* architecture, a lighter version of the popular *ResNet101* (He et al., 2016). This simplification is also justified because the more powerful version of Mask R-CNN based on the ResNet101 model was designed for more complicated visual detection tasks than ours. Originally, Mask R-CNN was trained on the *COCO* dataset (Lin et al., 2014) to detect and recognize 80 different classes of everyday objects. In our case, we have to localize and identify objects belonging to just one category (i.e., the *vehicle* category). To this end, we further simplify the model by reducing the number of the final fully convolutional layers responsible for the classification of the detected objects, making the model lighter. Once we have localized the instances of the objects, we count them estimating the number of vehicles present in the scene.

Local counting. The Sink node S starts this phase, sending a synchronization signal to all the smart cameras belonging to the system. Once received the synchronization signal, each node ν_i captures an image belonging to its underlying FOV and feeds it to the previously described CNN-based counting technique obtaining a set of masks masks_i localizing the vehicles present in the scene. The cardinality of this set of masks corresponds to the number of cars present in the image, i.e., the quantity η_i, that is sent with a message to the Sink node S. Then, the node ν_i packs this set of masks masks_i in a message m_i, sends it to all its neighboring nodes ν_j, and receives from them their corresponding set of masks masks_j packed in a message m_j. Once received a message m_j, the node ν_i is responsible for analyzing the potential vehicles present in the overlapped area between its FOV and the one of the node ν_j. To this end, it employs the homographic transformation $H_{j,i}$ computed during the system initialization, as described in Section 3.2. Specifically, it projects the masks belonging to the set masks_j into its image plane, filtering them and discarding the ones that overlap with the masks belonging to the set masks_i having a value of Intersection over Union (IoU) greater than a threshold that we empirically found to be optimal at 0.2. These masks indeed localize vehicles already detected, which should not be considered a second time. On the other hand, the cars left after this filtering are vehicles that were not detected in the FOV underlying the node ν_i, but instead found by the node ν_j, probably because of having a better view of this object. Referring to our graph modeling the system and reported in Figure 2, the number of the discarded cars after this filtering operation corresponds to the message $\mu_{j,i}$, that is sent to the Sink node S. We detail all the described steps in the Algorithm 2 and in the Procedure 3.

Algorithm 2 : Local Counting

At each Computational Signal by S, each node ν_i performs the following steps:

RECEIVECOMPUTSIGNAL() $\triangleright$ waits the computational signal from S
$\text{image}_i \leftarrow$ CAMERACAPTURE()
$\text{masks}_i \leftarrow$ MASKRCNN(image_i)
$\eta_i \leftarrow |\text{masks}_i|$
SENDMESSAGE(η_i, S) $\triangleright$ sends η_i to Sink node S
$m_i \leftarrow$ PACKMESSAGE(masks_i) $\triangleright$ builds message m_i containing masks_i
for each $j \in J$ **do** $\triangleright$ J is the set of neighboring nodes of node ν_i
    SENDMESSAGE(m_i, ν_j) $\triangleright$ sends m_i to node ν_j
    $m_j \leftarrow$ RECEIVEMESSAGE() $\triangleright$ receives message m_j from node ν_j
    $\text{masks}_j \leftarrow$ UNPACKMESSAGE(m_j) $\triangleright$ unpacks m_j containing masks_j
    $\mu_{j,i} \leftarrow$ COMPUTE_μ(masks_i, masks_j, $H_{j,i}$)
    SENDMESSAGE($\mu_{j,i}, S$) $\triangleright$ sends $\mu_{j,i}$ to Sink node S

Algorithm 3 : Computation of μ

μ represents the num of cars detected by ν_j and already detected by ν_i

Each node ν_i performs the following procedure:

procedure COMPUTE_μ(masks$_i$, masks$_j$, $H_{j,i}$)
    n_cars_already_detected $\leftarrow 0$
    for each mask $\in$ masks$_j$ **do**
        mask$_h \leftarrow$ PROJECT($H_{j,i}$, mask) $\triangleright$ projects mask points on plane i
        if mask$_h$ falls within image$_i$ **then**
            $\text{mask}_{\max} \leftarrow \arg\max_{m \in \text{masks}_i} \text{IoU}(\text{mask}_h, m)$
            if $\text{IoU}(\text{mask}_h, \text{mask}_{\max}) > \tau$ **then**
                n_cars_already_detected ++
    return n_cars_already_detected

3.4. Global Counting Algorithm

In this section, we describe the global counting algorithm that runs on the Sink node S, responsible for the fusion of the partial results coming from all the other nodes, and that finally outputs the number of cars present in the *entire* monitored parking area.

This phase starts when S receives all the η_i and the $\mu_{j,i}$ messages, i.e., the number of vehicles estimated in the single FOVs and the estimation of the number of cars already considered in the overlapping areas between neighboring cameras, from all the nodes belonging to the system. Specifically, for each overlapped area shared between a pair of nodes ν_i, ν_j, the node S receives two messages $\mu_{j,i}$ and $\mu_{i,j}$, the contents of which are computed by the two nodes employing two homographic transformations $H_{j,i}$ and $H_{i,j}$, respectively. These two quantities can be potentially different. We choose the best value by aggregating them, choosing between three different functions - max, min and mean, finding that the latter is the best one. Finally, the node S builds the final result, i.e., the estimation of the number of vehicles present in the *entire* parking lot, by summing up the content of all the η_i messages and subtracting the computed aggregated values. We detail all these steps in the Algorithm 4.

Algorithm 4 : Global Counting

The Sink node S performs the following steps:

1: **for each** $(\mu_{i,j}, \mu_{j,i})$ **do**

2: $\quad \overline{\mu_k} \leftarrow \textsc{Aggregate}(\mu_{i,j}, \mu_{j,i})$

3: global_cars_count $\leftarrow \sum_{n=1}^{N} \eta_n - \sum_{k=1}^{K} \overline{\mu_k}$

$\triangleright$ N is the set of nodes, K is the set of aggregations

4. Experimental Setup

In this section, we describe the simulated scenario that we exploited for our experiments. In particular, we extended the *CNRPark-EXT* dataset (Amato et al., 2017), adapting it to be suitable for the counting task so that it was usable for training the vehicles counting CNN running on the smart cameras and applicable to validate our multi-camera algorithm. Furthermore, we briefly describe the *PKLot* dataset (de Almeida et al., 2015), a public dataset comprising parking lot scenes that we exploited for further assessing the generalization capabilities of the local vehicles counting network. Then, we illustrate the employed evaluation metrics, and, finally, we report some implementation details.

4.1. The CNRPark-EXT Dataset

In this work, we exploit the *CNRPark-EXT* public dataset introduced in Amato et al. (2017), a collection of annotated images of vacant and occupied parking spaces on the campus of the National Research Council (CNR) in Pisa, Italy. This dataset represents most of the challenging situations that can be found in a real scenario: nine different cameras capture the images under various weather conditions, angles of view, light conditions, and many occlusions. Furthermore, the cameras have their fields of view partially overlapped. Since this dataset is specifically designed for parking lot occupancy detection, it is not directly usable for the counting task. Indeed, each image, called *patch*, contains one parking space labeled according to its occupancy status - 0 for vacant and 1 for occupied. Since this work aims at counting the cars present in the parking area, we extended it by considering the full images and adapting the ground truth to our purposes.

Specifically, we created a suitable label set to train and evaluate the local vehicles counting based on Mask R-CNN. In this case, labels correspond to *binary masks*, i.e., binary images identifying the polygons surrounding the vehicles we want to detect. Since mask creation is a very time-consuming operation, differently from our previous work (Ciampi et al., 2018), we considered the *raw* masks obtained directly from the bounding boxes localizing the occupied parking spaces. The idea is that we do not need precise polygons that identify the vehicles we want to detect. Still, we can use the region within the delimiters that identify the occupied parking spaces and the underlying part of the car.

On the other hand, to validate our multi-camera algorithm, we built a simulated scenario considering some sequences of images belonging to different cameras captured simultaneously. In other words, a sequence is defined as the set of images captured by the different smart cameras that are monitoring the parking area at the same moment. Hence, a sequence represents a snapshot of the *entire* parking lot at a given timestamp, and it takes into account all the spaces from the available different views. We manually annotated these sequences to obtain the ground truth car counts. Specifically, we considered the single images composing a sequence, counting the vehicles present in the scenes, but taking care of accounting for them just once if they appear in more than one view, i.e., discarding the cars from the global count if they were located in the overlapping areas. We labeled six different sequences, two for each weather condition, considering the images belonging from $camera_2$ to $camera_9$. We did not consider $camera_1$ since it has small and particularly skewed field-of-view overlaps with the other cameras, hindering the automatic homography estimation and the subsequent projections.

4.2. The PKLot Dataset

To further validate the generalization capabilities of the CNN-based local vehicles counting algorithm, we exploited an additional public dataset, named

PKLot (de Almeida et al., 2015). In particular, this dataset is composed by three different scenarios describing three different parking lot scenes - *UFPR04*, *UFPR05* and *PUC*. We considered only the first two subsets since the third one contains images captured from a fixed camera located at the height of the 10th floor of a building, which provides a slanted view of the parking lot and results in a different setting without intra-vehicle occlusions. Since also the *PKLot* dataset, like the *CNRPark-EXT* one, is specifically designed for the parking lot occupancy detection task, we manually re-labeled the ground truth for our purposes as already described in Section 4.1, obtaining a simulation scenario suitable for measure the performance of our solution for the counting task.

4.3. Evaluation Metrics

Following other counting benchmarks, we exploited Mean Absolute Error (*MAE*), Mean Square Error (*MSE*), and Mean Relative Error (*MRE*) as the metrics for the performance evaluation, defined as follows:

$$MAE = \frac{1}{N}\sum_{n=1}^{N}|c_n^{gt} - c_n^{pred}|, \tag{1}$$

$$MSE = \frac{1}{N}\sum_{n=1}^{N}(c_n^{gt} - c_n^{pred})^2, \tag{2}$$

$$MRE = \frac{1}{N}\sum_{n=1}^{N}\frac{|c_n^{gt} - c_n^{pred}|}{\text{num\_spaces}_n}, \tag{3}$$

where N is the total number of the images, c_{gt}, c_{pred} and num_spaces_n are the actual count, the predicted count, and the total number of parking spaces of the n-th image, respectively. Note that as a result of the squaring of each difference, MSE effectively penalizes large errors more heavily than small ones and thus should be more useful when large errors are particularly undesirable. On the other hand, MRE also considers the relation between the error and the total number of objects present in the image.

4.4. Implementation Details

We report in this section some implementation details concerning the Mask R-CNN-based algorithm responsible for the prediction of the number of vehicles in the single images. In particular, we trained the modified Mask R-CNN initializing the weights of the ResNet50 backbone with the ones of a pre-trained model on *ImageNet* (Deng et al., 2009), a popular dataset for classification tasks, and the remaining ones at random. We froze the backbone for the first 10 epochs, and then we trained the whole network for 20 additional epochs. We used Stochastic Gradient Descent (SGD) to perform the CNN parameters update. Concerning the Region Proposal Network, explained in Section 3.3, we exploited a set of five anchors of sizes 16, 32, 64, 128, and 256 pixels. To prevent overfitting, we applied some standard augmentation techniques to the training data: images are horizontally flipped with a 0.5 probability, then their pixels are multiplied by a random value between 0.8 and 1.5, and finally, they are blurred using a Gaussian kernel with a standard deviation of a random value between 0 and 5. Then, to support training multiple images per batch, we resized all pictures to the same size. If an image was not square, we padded it with zeros to preserve the aspect ratio. In the end, we obtained images of size 1024×1024. At inference time, images were resized and padded with zeros to get a square picture of size 1024×1024, and no other augmentations took place.

5. Experiments and Results

In this section, we report the experiments and the obtained results. First, we evaluate the performance against other state-of-the-art solutions of the CNN-based technique responsible for estimating the vehicles in the single images directly onboard the smart cameras, also stressing its generalization capabilities. Then, we validate the effectiveness of our multi-camera algorithm by testing it in the simulated scenario previously described. We demonstrate that our system can benefit from the redundant information deriving from the different cameras, obtaining performance improvements in all the considered counting metrics.

5.1. Experiments on the CNN-based counting solution on the edge

5.1.1. State-of-the-art comparison

We compared our solution with the results obtained in our previous work Ciampi et al. (2018), where we presented a centralized counting approach based on the original version of Mask R-CNN having the ResNet101 model as a features extractor, which has been fine-tuned on a very small manually annotated subset of the CNRPark-EXT dataset, starting from the model pre-trained on the *COCO* dataset (Lin et al., 2014). We filtered the detections considering only the predictions related to the car class, and we counted them. Although this solution is very computationally expensive and unsuitable for edge devices, it represents a direct comparison in terms of counting on the same dataset. We also compared our technique against the method proposed in Amato et al. (2017), an approach for car parking occupancy detection based on *mAlexNet*, a deep CNN designed explicitly for smart cameras. This work represents an indirect method for counting cars in a parking lot, as the counting problem is cast as a classification problem: if a parking space is occupied, we increment the total number of cars; otherwise, we do not. We illustrate the results in Table 1, where we also report the performance obtained using the Mask R-CNN network without a preliminary fine-tuning on the CNRPark-EXT dataset. Our solution performs better than the other considered methods, considering all three counting metrics. In particular, our approach outperforms the solution introduced in Ciampi et al. (2018), despite the latter employing a more deep and powerful CNN, and it is designed to be used as a centralized-server solution. This is explained by the fact that in Ciampi et al. (2018) the authors fine-tuned the CNN using a tiny dataset. Consequently, the algorithm overfits on the training data, and it cannot generalize over the test subset. It is also worthy of notice that our CNN also outperforms the mAlexNet network, even though the latter knows the exact location of the parking spaces. Figure 4 shows some examples of images belonging to different cameras and different weather conditions together with the masks localizing them computed by our counting solution.

Method	CNRPark-EXT MAE	CNRPark-EXT MSE	CNRPark-EXT MRE	PKLot MAE	PKLot MSE	PKLot MRE
(Amato et al., 2017)	1.34	8.00	0.04		-	
(Ciampi et al., 2018)	1.05	4.41	0.03		-	
ResNet50 Mask R-CNN	11.20	247.40	0.30	16.90	522.40	0.48
Our solution	**0.49**	**1.04**	**0.01**	**4.56**	**33.88**	**0.13**

Table 1: Local Counting: Left-side: results obtained using our counting solution on the edge compared with other state-of-the-art approaches; we get the best results on all the three considered counting metrics. Right-side: evaluation of the generalization capabilities on the *PKLot* dataset (de Almeida et al., 2015), using the model trained on the *CNRPark-EXT* dataset; we achieved an error that is approximately four times lower than the one obtained with the COCO pre-trained model.

(a) Image from Camera$_2$ (b) Image from Camera$_8$

Figure 4: Two examples of the output of our counting method. Images are taken from the CNRPark-EXT dataset. We report the predictions and the estimate of the number of vehicles present in the scene.

5.1.2. Generalization capabilities

Errors in vehicle detection and counting are due to many reasons, but critical points are different light conditions and diverse perspectives. Weather conditions might produce significant illumination changes since puddles and wet floors create a textural pattern that may lead to an error, and sunbeams can create reflections on the car windscreen, covering the majority of the images with saturated patterns. When a CNN does not generalize well, it works well only in the conditions where it was trained.

To measure the robustness of our approach to these scenarios, we performed two types of experiments exploiting the *CNRPark-EXT* dataset: i) *inter-weather* and ii) *inter-camera* experiments. In the former, we trained our CNN with images taken in one particular weather condition, and we computed the performance metrics obtained on images having different weather conditions. In particular, we performed three experiments, training respectively on the *Sunny*, *Overcast* and *Rainy* subsets of the CNRPark-EXT dataset. In the latter, we trained our algorithm employing images from one camera, and then we computed the performance metrics on pictures captured by another camera. In particular, we performed two experiments, training with images coming respectively from camera_1 and camera_8. We chose these two cameras because they are particularly representative since one has a side view of the parking lot while the other has a pure front view.

We report the results of the two experiments in Table 2 and Table 3, respectively. We achieve a good generalization in both the considered scenarios. We experienced a larger amount of error when the CNN was trained and tested on two opposite weather conditions, for instance, *Sunny* and *Rainy*, while the more accurate model was the one trained on *Overcast* weather conditions. However, the performance difference is quite small. On the other hand, in *inter-camera* experiments, the model trained on camera_8 is the best, and it has a slight drop in performance only when tested on the camera_1 subset. The model trained on the camera_1 dataset performs in general worse. This is probably due to a bias

	Sunny			Overcast			Rainy		
Train Set	MAE	MSE	MRE	MAE	MSE	MRE	MAE	MSE	MRE
Sunny	-	-	-	0.29	0.34	0.009	0.96	2.78	0.02
Overcast	0.62	1.09	0.02	-	-	-	0.56	1.26	0.01
Rainy	0.84	1.65	0.02	0.49	0.65	0.01	-	-	-

Table 2: CNRPark-EXT: Results of inter-weather experiments in terms of counting metrics obtained when training on sunny, overcast, or rainy weather.

		Test Set								
Metric	Train Set	C1	C2	C3	C4	C5	C6	C7	C8	C9
MAE	C1	-	0.77	1.21	2.53	3.26	2.57	2.88	2.88	1.54
	C8	3.87	0.85	0.76	0.45	0.48	0.71	1.07	-	0.41
MRE	C1	-	0.08	0.05	0.06	0.07	0.05	0.06	0.05	0.05
	C8	0.11	0.09	0.03	0.01	0.01	0.01	0.02	-	0.01
MSE	C1	-	1.48	2.91	10.61	20.24	13.50	19.82	17.30	7.19
	C8	22.60	1.78	1.36	0.57	0.74	0.95	4.97	-	2.13

Table 3: CNRPark-EXT: Results of inter-camera experiments in terms of counting metrics obtained when training on camera 1 and camera 8.

in the CNRPark-EXT dataset, where the majority of the images are captured from a frontal viewpoint.

Moreover, to further validate the generalization capabilities of our approach, we considered our counting network trained on the entire training set of the *CNRPark-EXT* dataset, and we tested it over a different dataset, the *PKLot* dataset (de Almeida et al., 2015). Results are shown in Table 1 where we also report the performance obtained using the Mask R-CNN network without a preliminary fine-tuning on the *CNRPark-EXT* dataset. As we can see, using our solution, we achieve an error that is approximately four times lower than the one obtained with the COCO pre-trained model.

5.2. Experiments on the Multi-Camera Scenario

To the best of our knowledge, there are no annotated datasets in the literature suitable for evaluating counting algorithms operating on multiple FOV-overlapping cameras. The most relevant work in this context is Nieto et al. (2019), in which there are only two overlapping cameras facing each other with an extreme perspective transformation between the two; this makes any automatic perspective computation nearly impossible without manual intervention, and this is a mandatory assumption for our proposed method. Hence, we performed our experiments on the extended version of the CNRPark-EXT dataset created on purpose in this work, which we hope will become a new benchmark for this task. Furthermore, to demonstrate that our algorithm can benefit from the redundant information deriving from the different cameras, we compared the obtained results against a baseline and a simplified version of our algorithm.

Specifically, we compared our solution against a system that is not aware of the other cameras' overlapped areas, and so it just sums up all the vehicles detected by all the cameras belonging to a sequence (Naïve Counting **N**). Then, we considered a more conservative approach, where the nodes employ the homographic transformations only with the purpose of black-masking the overlapped areas (Overlap Masking **M**). This latter baseline then loses the ability to take advantage of monitoring the same lots from different views. However, it is still aware of the locations of the overlapping areas, and it considers the vehicles inside them only once.

Results are shown in Table 4. Our solution obtains the best results compared to the considered baselines in all the three counting metrics and all the employed scenarios. We report the errors concerning the considered six sequences of the CNRPark-EXT dataset, together with the MAE, MSE, and MRE, which summarize the mean results regarding all the scenarios. As an example, in Figure 5 we also report the output of our multi-camera algorithm for a pair of images belonging to two different cameras having a shared area in their field of view, where we highlight in red and blue the masks projected from one camera to the other, using the previously computed homographic transformations.

	Error			Absolute Err.			Squared Err.			Relative Err. (%)		
	N	**M**	**O**	**N**	**M**	**O**	**N**	**M**	**O**	**N**	**M**	**O**
Overcast-1	124	-33	**2**	124	33	**2**	15,376	1,089	**4**	71.6	19.0	**1.2**
Overcast-2	131	-26	**1**	131	26	**1**	17,161	676	**1**	76.1	15.1	**0.6**
Rainy-1	80	-39	**-5**	80	39	**5**	6,400	1,521	**25**	47.6	23.2	**2.9**
Rainy-2	105	-44	**-5**	105	44	**5**	11,025	1,936	**25**	54.4	22.8	**2.6**
Sunny-1	117	-38	**2**	117	38	**2**	13,689	1,444	**4**	68.0	22.1	**1.2**
Sunny-2	113	-37	**2**	113	38	**2**	12,769	1,444	**4**	66.1	22.2	**1.2**
Mean	111.6	-36.1	**-0.5**	111.6	36.3	**2.8**	12,736.6	1,351.6	**10.5**	63.9	20.7	**1.6**

N: Naïve Counting; **M**: Overlap Masking; **O**: Ours (mean aggr., IoU Threshold $\tau = 0.2$)

Table 4: Results using our multi-camera counting algorithm, considering the *entire* parking lot. We compare our solution against a baseline and a simplified version of our algorithm. We report the errors obtained on the six considered sequences (two for each weather condition) of the CNRPark-EXT dataset that we extend on purpose.

(a) Image from Camera_9 (b) Image from Camera_8

Figure 5: Example of the output of our multi-camera algorithm for a pair of images belonging to two different cameras i, j having a shared area in their FOV. We report in green the masks localizing the vehicles detected by a camera in its own FOV, while in red and blue, the masks projected from camera j to camera i and vice-versa, employing the homographic transformations pre-computed during the system initialization.

6. Conclusion

This paper presented a distributed artificial intelligence-based system that automatically counts the vehicles present in a parking lot using images taken by multiple smart cameras. Unlike most of the works in literature, we introduced a multi-camera approach that can estimate the number of cars present in the *entire* parking area and not only in the single captured images. The main peculiarities of this approach are that all the computation is performed in a distributed manner at the edge of the network and that there is no need for any extra information about the monitored parking area, such as the location of the parking spaces, nor any geometric information about the position of the cameras in the parking lot. We modeled our system as a graph. The nodes, i.e., the smart cameras, are responsible for estimating the number of cars present in their view and merging data from nearby devices with an overlapping field of view. Our solution is simple but effective, combining a deep-learning technique with a distributed geometry-based approach. We evaluated our algorithm on the CNRPark-EXT dataset, which we specifically extended and which we hope will become a new benchmark for counting vehicles in multi-camera parking area scenarios. Through an experimental evaluation, we showed how we benefit from redundant information from different cameras while improving overall performance.

There are multiple lines of future development that can help improve the proposed system. Although our multi-camera algorithm is flexible, one limitation relies on computing the homographic matrix between images captured by cameras placed in completely different locations, such as facing each other. In such cases, the two perspectives are totally different, and manual intervention is required to avoid the generation of an inaccurate geometric transformation.

Acknowledgements

This work was partially supported by H2020 project AI4EU under GA 825619, by H2020 project AI4media under GA 951911, and by Tuscany POR

FSE 2014-2020 AI-MAP (CNR4C program, CUP B15J19001040004).

References

de Almeida, P. R., Oliveira, L. S., Britto, A. S., Silva, E. J., & Koerich, A. L. (2015). PKLot – a robust dataset for parking lot classification. *Expert Systems with Applications*, *42*, 4937–4949. URL: `https://doi.org/10.1016%2Fj.eswa.2015.02.009`. doi:`10.1016/j.eswa.2015.02.009`.

Amato, G., Bolettieri, P., Moroni, D., Carrara, F., Ciampi, L., Pieri, G., Gennaro, C., Leone, G. R., & Vairo, C. (2018). A wireless smart camera network for parking monitoring. In *2018 IEEE Globecom Workshops (GC Wkshps)*. IEEE. URL: `https://doi.org/10.1109%2Fglocomw.2018.8644226`. doi:`10.1109/glocomw.2018.8644226`.

Amato, G., Carrara, F., Falchi, F., Gennaro, C., Meghini, C., & Vairo, C. (2017). Deep learning for decentralized parking lot occupancy detection. *Expert Systems with Applications*, *72*, 327–334. URL: `https://doi.org/10.1016%2Fj.eswa.2016.10.055`. doi:`10.1016/j.eswa.2016.10.055`.

Amato, G., Ciampi, L., Falchi, F., & Gennaro, C. (2019a). Counting vehicles with deep learning in onboard UAV imagery. In *2019 IEEE Symposium on Computers and Communications (ISCC)*. IEEE. URL: `https://doi.org/10.1109%2Fiscc47284.2019.8969620`. doi:`10.1109/iscc47284.2019.8969620`.

Amato, G., Ciampi, L., Falchi, F., Gennaro, C., & Messina, N. (2019b). Learning pedestrian detection from virtual worlds. In *Lecture Notes in Computer Science* (pp. 302–312). Springer International Publishing. URL: `https://doi.org/10.1007%2F978-3-030-30642-7_27`. doi:`10.1007/978-3-030-30642-7_27`.

Arteta, C., Lempitsky, V., & Zisserman, A. (2016). Counting in the wild. In *Computer Vision – ECCV 2016* (pp. 483–498). Springer International

Publishing. URL: https://doi.org/10.1007%2F978-3-319-46478-7_30. doi:10.1007/978-3-319-46478-7_30.

Benedetto, M. D., Carrara, F., Ciampi, L., Falchi, F., Gennaro, C., & Amato, G. (2022). An embedded toolset for human activity monitoring in critical environments. *Expert Systems with Applications*, *199*, 117125. URL: https://doi.org/10.1016%2Fj.eswa.2022.117125. doi:10.1016/j.eswa.2022.117125.

Boominathan, L., Kruthiventi, S. S. S., & Babu, R. V. (2016). CrowdNet. In *Proceedings of the 24th ACM international conference on Multimedia*. ACM. URL: https://doi.org/10.1145%2F2964284.2967300. doi:10.1145/2964284.2967300.

Ciampi, L., Amato, G., Falchi, F., Gennaro, C., & Rabitti, F. (2018). Counting vehicles with cameras. In S. Bergamaschi, T. D. Noia, & A. Maurino (Eds.), *Proceedings of the 26th Italian Symposium on Advanced Database Systems, Castellaneta Marina (Taranto), Italy, June 24-27, 2018*. CEUR-WS.org volume 2161 of *CEUR Workshop Proceedings*. URL: http://ceur-ws.org/Vol-2161/paper12.pdf.

Ciampi, L., Carrara, F., Amato, G., & Gennaro, C. (2022). Counting or localizing? evaluating cell counting and detection in microscopy images. In *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. SCITEPRESS - Science and Technology Publications. URL: https://doi.org/10.5220%2F0010923000003124. doi:10.5220/0010923000003124.

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

Ciampi, L., Santiago, C., Costeira, J., Gennaro, C., & Amato, G. (2021). Domain adaptation for traffic density estimation. In *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer*

Graphics Theory and Applications. SCITEPRESS - Science and Technology Publications. URL: https://doi.org/10.5220%2F0010303401850195. doi:10.5220/0010303401850195.

Ciampi, L., Santiago, C., Costeira, J. P., Gennaro, C., & Amato, G. (2020b). Unsupervised vehicle counting via multiple camera domain adaptation. In A. Saffiotti, L. Serafini, & P. Lukowicz (Eds.), *Proceedings of the First International Workshop on New Foundations for Human-Centered AI (NeHuAI) co-located with 24th European Conference on Artificial Intelligence (ECAI 2020), Santiago de Compostella, Spain, September 4, 2020* (pp. 82–85). CEUR-WS.org volume 2659 of *CEUR Workshop Proceedings*. URL: http://ceur-ws.org/Vol-2659/ciampi.pdf.

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)*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2005.177. doi:10.1109/cvpr.2005.177.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2009.5206848. doi:10.1109/cvpr.2009.5206848.

Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus. *Communications of the ACM*, *24*, 381–395. URL: https://doi.org/10.1145%2F358669.358692. doi:10.1145/358669.358692.

Gu, C., Lim, J. J., Arbelaez, P., & Malik, J. (2009). Recognition using regions. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2009.5206727. doi:10.1109/cvpr.2009.5206727.

He, K., Gkioxari, G., Dollar, P., & Girshick, R. (2017). Mask r-CNN. In *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE.

URL: https://doi.org/10.1109%2Ficcv.2017.322. doi:10.1109/iccv.2017.322.

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

Khan, M. Z., Harous, S., Hassan, S. U., Khan, M. U. G., Iqbal, R., & Mumtaz, S. (2019). Deep unified model for face recognition based on convolution neural network and edge computing. *IEEE Access*, *7*, 72622–72633. URL: https://doi.org/10.1109%2Faccess.2019.2918275. doi:10.1109/access.2019.2918275.

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

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

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*, 91–110. URL: https://doi.org/10.1023%2Fb%3Avisi.0000029664.99615.94. doi:10.1023/b:visi.0000029664.99615.94.

Nieto, R. M., Garcia-Martin, A., Hauptmann, A. G., & Martinez, J. M. (2019). Automatic vacant parking places management system using multicamera vehicle detection. *IEEE Transactions on Intelligent Transportation Systems*, *20*, 1069–1080. URL: https://doi.org/10.1109%2Ftits.2018.2838128. doi:10.1109/tits.2018.2838128.

Oñoro-Rubio, D., & López-Sastre, R. J. (2016). Towards perspective-free object counting with deep learning. In *Computer Vision – ECCV 2016* (pp. 615–629). Springer International Publishing. URL: https://doi.org/10.1007%2F978-3-319-46478-7_38. doi:10.1007/978-3-319-46478-7_38.

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

Sindagi, V. A., & Patel, V. M. (2018). A survey of recent advances in CNN-based single image crowd counting and density estimation. *Pattern Recognition Letters*, *107*, 3–16. URL: https://doi.org/10.1016%2Fj.patrec.2017.07.007. doi:10.1016/j.patrec.2017.07.007.

Ujjan, R. M. A., Pervez, Z., Dahal, K., Bashir, A. K., Mumtaz, R., & González, J. (2020). Towards sFlow and adaptive polling sampling for deep learning based DDoS detection in SDN. *Future Generation Computer Systems*, *111*, 763–779. URL: https://doi.org/10.1016%2Fj.future.2019.10.015. doi:10.1016/j.future.2019.10.015.

Vítek, S., & Melničuk, P. (2017). A distributed wireless camera system for the management of parking spaces. *Sensors*, *18*, 69. URL: https://doi.org/10.3390%2Fs18010069. doi:10.3390/s18010069.

Xie, W., Noble, J. A., & Zisserman, A. (2016). Microscopy cell counting and detection with fully convolutional regression networks. *Computer Meth-*

ods in Biomechanics and Biomedical Engineering: Imaging & Visualization, *6*, 283–292. URL: `https://doi.org/10.1080%2F21681163.2016.1149104`. doi:`10.1080/21681163.2016.1149104`.

Zhang, S., Wu, G., Costeira, J. P., & Moura, J. M. F. (2017). Understanding traffic density from large-scale web camera data. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. URL: `https://doi.org/10.1109%2Fcvpr.2017.454`. doi:`10.1109/cvpr.2017.454`.

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

Declaration of interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.