Detection of parking space availability based on video

1st Miloslav Kužela  
Department of Radio Electronics  
FEEC, Brno University of Technology  
Brno, Czech Republic  
240648@vut.cz

2nd Tomáš Frýza  
Department of Radio Electronics  
FEEC, Brno University of Technology  
Brno, Czech Republic  
fryza@vut.cz

Abstract—This paper deals with the use of Machine vision and ML (Machine Learning) for a parking lot occupation detection. It presents and compares an already existing technology that solves such a problem with an AI (Artificial Intelligence) use-case. It introduces tools used to train and create such models and their subsequent results as well as a dataset that was used to verify the trained networks and discusses the future of how such a technology could be used to effectively and more affordably detect occupied parking spaces on parking lots.

Index Terms—Machine Vision, Machine learning, Parking occupancy, Python

I. INTRODUCTION

The use of parking sensors to monitor larger, enclosed parking lots is the norm at this day and age. But with the use of ceiling or wall mounted sensors, a lot of parking lots are left out due to their locations. The cost of such devices must be taken into account. With the recent rise of AI, especially Machine learning and Machine vision, tackling such a problem with these tools could be of benefit. A CCTV (Closed-circuit television) camera and a computer is all that is needed for the solution of this problem. And due to a lot of already existing parking lots having a camera in use, it is more cost efficient. The aim of this article is to show and introduce the reader to a concept of such a system and its tools to create a simple parking detection algorithm using Machine learning and Machine vision. It present tools that helps a user train, test or fine-tune an existing model for such a use case. It also consults the possible ways of utilizing such a system in real world application.

A. Existing commercial solutions

The current state of systems that are used for parking occupancy monitoring can be divided into two groups. One that counts only incoming and outgoing cars, compares it with the total parking spaces available and shows the result on an information board of some sorts. The other type is using active sensors placed near all parking places which detects the presence of a car. Such solution is accurate although expensive and is hard to realize in an open space parking lot. Ground sensors do exist, but due to battery requirements they are not as effective. An example of such sensors include an Ultrasonic sensor, Magnetic or MMW (Millimeter wave) radar. For example an ultrasonic sensor, mostly used in enclosed parking lots, due to the fact that they often require to be mounted on the roof so they can measure the distance between it and the floor. When a car parks under such sensor it detect the change of distance, this information is then sent to a central unit which keeps track of all the parking spaces and their occupancy. When compared to a solution proposed later in this paper, such sensors are more accurate, although that advantage is outweighed by the latter’s price, and the possibility of utilizing an already existing infrastructure such as cameras.

II. METHOD

A. Use of Cameras

When relying on data acquired by the means of a digital camera. It is necessary to be aware of possible inaccuracies that arise when using such technology. If the weather conditions change there is a high chance that the camera’s automatic functions will make the resulting photos look much different than what it captured last time. A simple machine vision algorithm could have a problem with such a fact. So an image preprocessing stage is in order.

B. Initial idea

The use of machine learning in such a problem is simple. Having a model decide between two classes: Car and background. Then apply such a model to a parking lot images and tests its accuracy. The main problem was deciding what existing machine vision network/model to use. The main network candidates were either faster R-CNN (Regional Based Convolution Neural Network) [?] or Retina-net [?]. Considered models were Mobilenet V3 [?] and ResNet50 [?].

C. R-CNN

As a regional proposal network, meaning that the first stage of prediction is locating regions of interest which are then used in prediction, which could be a benefit as the camera is not moving and the regions are staying mostly at a constant location in the image. MobileNetV3 was the newest model available that uses the Convolution network as its base and is designed to be ran on a low power mobile devices. There are two versions, small and large, which correlate to its number of layers. [?]
D. Retinanet

This network uses a one-stage approach instead of the two stage of the previous network mentioned. It uses a FPN(Feature Pyramid Network) and Focal Loss to predict the correct class. Such network is suitable for detecting multiple small object in an image. It was chosen due to its faster one stage approach. Creators of this network recommend the use of a Resnet50 FPN model [?].

E. Dataset

A custom dataset was created to train pre-trained models and models from scratch called T10LOT. It consists of photos of a parking lot located next to a FEEC’s (faculty of electronic engineering and communication) building. Photos were captured on an iPhone X camera with an application that took a photo every 30 minutes. The phone was held up by a 3D printed arm that was fixed to a window frame. The dataset contains over 100 images including different weather conditions such as fog, night and rainy weather. This dataset was used to train and verify the models. All of the images have a resolution of 1980x1080px.

F. Custom tools

Custom tools and scripts were created to assist with the creation of datasets, training and testing. Coded in Python with the use of Jupyter notebook. These tools allow the user to create their own datasets that can be latter used with the training CLI (Command Line Interface) applications. Dataset creation tools run in a Jupyter notebook and feature a GUI (Graphical User Interface). The training function includes a configuration stage that allows user to specify how to train the network, this includes the number of datasets to use, network and model combination, number of epochs, learning rate and more. There is also a testing script that test the trained network on testing images of a selected dataset that were not used during training, this eliminates any possible bias. The script outputs the accuracy and speed of inferencing a single image, this data is later used to decide the fastest network. The annotating application was reworked from a another paper that featured comparing trained models on a multiple datasets [?].

G. Labeling

As mentioned in chapter ?? a tool was created that assist with the labelling/annotating of images. User puts all the images into a specific folder and runs the application, then by clicking around individual parking slots crates polygons that surround it. If all of the images were captured in the same angle and position, there is a button for labelling all the images same way as the initial one. The second stage of labelling is marking each parking space as either occupied or not. This is done with another included tool, that allows the user to annotate individual spaces by just a click. Once the user is finished, program creates a dataset in a format that can be later used with the training and testing scripts.

H. Training

To be able to accurately use a neural network, it firstly needs to be trained on a dataset according to its specific use-case. In this case, images with parking lots. There is a difference between training a model with randomized weights and fine tuning a model with pre-retrained weight. All of the pretrained models featured in this paper were trained on a COCO(Common Objects in Context) dataset, which features a lot of different images featuring different objects. This trains the network to properly do region proposals and is easier to retrain on a dataset that fits the use-case. Models that started with randomized weights values were trained on two datasets that are really similar to the wanted use case, meaning that the camera was positioned above the parking lots and behind a window, the two datasets used are called CNRPark [?] and PKLot [?]. Lastly both pretrained and randomized weights models were trained on a T10LOT dataset for 20 epochs. No augmentation was performed on the training set of images, only the conversion to tensors were applied, although the use of color augmentation would benefit if training with images taken by an IR (Infra-red) cameras.

I. Testing

The testing script, loads the wanted model and runs it in evaluation mode through a set of testing images, that were not used in training. These are randomly selected when creating a dataset with the creation script included in this work. The testing is evaluated based on the F1 score, which is calculated from precision and recall. Lastly there is an option to enable exporting of the results into images, where each individual detection is visible. The current model is currently designed to differentiate between two classes: Car parked in a parking slot and background. When trying to understand the testing results an example of a badly trained model is presented in Fig.??.
Red dot is a false positive detection. Squares that highlight the individual parking spaces turn green if a valid detection is present. Red squares are occupied parking spaces that were not classified as and hence are categorized as false negatives.

\[
\text{precision} = \frac{TP}{TP + FP} \\
\text{recall} = \frac{TP}{TP + FN} \\
F1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}
\]

### III. Results

A handful of model combinations were tested. The testing was focused on their speed and accuracy. Images were provided from the T10LOT datasets, which contains 25 of pictures dedicated for testing. After comparing the results of individual models it is notable that the fastest and most effective ones are from the MobileNetV3 family, where the small version is faster but less accurate and vice-versa.

#### A. Difficult scenarios

It is worth noting the results of inferencing the networks on images that include a difficult scenario such as fog, rain or dark environment. It was surprising to see, that even with water droplets Fig. ??, that were in front of the camera and worked as a simulation of moisture accumulating on the lens, both MobileNetV3 networks were efficient at detecting parked cars, same applies to fog and even darkness Fig. ??, where it is hard even for a human to recognize if a parking slot is occupied or not.

### IV. Hardware Selection and Solution Proposal

The final Parking system will use a raspberry Pi V4 with a camera that overlooks the selected parking lot. The single board computer would need to be connected to a network by either a Wi-Fi connection or an Ethernet cable. Multiple cameras can be used to cover more of the parking place. A map of the parking lot would need to be created by labeling the individual position of parking places in the image. This would ensure that the final algorithm would label these places as occupied or not. The single board computer would run the object detection model and would mark the detected places a occupied, this information would be sent to a database server that could be accessed at any time by an IoT device that would display the current occupancy status of the parking lot.

#### A. Possible shortcomings

As the system is relying solely on cameras it could be easily disturbed if just one camera were to be slightly shifted. The
neural network part of the algorithm wouldn’t mind this fact, but the detection one would lose the correct positions of the individual parking slots. A system that would combat this problem needs to be proposed. The current system is designed to work only during the day, as it doesn’t include IR light emitters or cameras that have an option to disable their IR filters. If such a night vision camera were to be used the network would need to be retrained on a new set of images taken in dark scenario.

V. CONCLUSION

As the number of cars and parking places increases in the world a more accessible and affordable space monitoring system will be required. With the increase use of Machine learning utilizing it in such a use case is possible as was shown in this paper. The use of a single board computer to monitor parking spaces can be cheap and efficient. The tools used for this research are available on a GitHub repository: https://github.com/slavajda02/parking-research-argon as well as a download link containing the testing result images and the created dataset previously mentioned. A further research and test will be conducted on a prototype of a system previously mentioned to see how quickly and effectively can such a computer process individual images.

ACKNOWLEDGMENT

The Author would like to acknowledge the contribution and valuable time provided by the supervisor, doc. Ing. Tomáš Frýza Ph.D., as well as to thanks the authors of other researches focusing on parking detection algorithm.

REFERENCES