Supporting the process of sewer pipes inspection using machine learning on embedded devices

Mieszko Klusek[1] and Tomasz Szydlo[1]

AGH University of Science and Technology,
Institute of Computer Science, Krakow, Poland

Abstract. We are currently seeing an increasing interest in using machine learning and image recognition methods to support routine human-made processes in various application domains. In the paper, the results of the conducted research on supporting the sewage network inspection process with the use of machine learning on embedded devices are presented. We analyze several image recognition algorithms on real-world data, and then we discuss the possibility of running these methods on embedded hardware accelerators.

Keywords: IoT, machine learning, embedded devices

1 Introduction

Supporting processes and decision-making using machine vision and artificial intelligence is becoming more popular in many industries and everyday lives. Cars that support the driver or cameras that suggest what settings to choose for the current scenery become everyday life. One of the processes that can be improved using the above techniques is a visual inspection of sewer networks. Every day, the average operator inspects several hundred meters of sewage networks using robots with installed cameras, where he constantly observes the acquired image and provides information on the condition of pipes, damage present in the network, and structural elements [3]. Long working hours and their monotony may have a negative impact on the quality of the inspections carried out. According to J. Dirksen et al. [2] on average, the operator ignores 25% of defects during the inspection, so additional support by informing the operator in real-time about automatically detected objects or status changes can improve the quality of the work performed.

The inspection process can be supported by machine learning. Unfortunately, the use of cloud computing services is usually not possible due to limited network connectivity at the inspection site. The development of hardware accelerators for launching artificial intelligence models has recently made it possible to use it on embedded devices with limited resources that operators work with on a daily basis. We have conducted research to analyse the possibility of using machine vision methods to support the process of visual inspection of sewage networks on embedded devices.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_27

This paper presents a comprehensive approach to the ML-based automatic detection of defects and structural elements of sewage networks based on video material analysis from the network inspection. The possibility of using the methods of classification, detection, and image segmentation was analyzed. The considered data set includes photos from different cameras; thus their quality and resolution are varied. The photos are single frames showing defects or structural elements, wherein the case of other works described in the literature was mostly a large number of consecutive video frames. Finally, two image analysis methods were used - segmentation and classification for which performance studies were carried out for many network architectures. The research was conducted to analyze these algorithms' execution time based on deep convolutional neural networks on embedded devices using hardware accelerators.

The organization of the paper is as follows. The section 2 describes the related work and section 3 discusses the research methodology. Sections 4 and 5 analyses the machine learning methods for video processing. The section 6 describes the evaluation, while section 7 concludes the paper.

2 Related work

The section presents issues related to the context of the research work. The purpose of the sewage network inspection process is justified and the equipment used for this purpose is discussed. The concept of observation is introduced, i.e. a description of a damage, structural element or conditions inside a section of the sewage network, which is an elementary component of the inspection description.

2.1 Pipe inspection process

The main method of inspection of sewer networks is video inspection using CCTV cameras. It is performed periodically by companies providing such services to assess the network's quality and plan possible repairs. The horizontal part of the network inspected is called a section and usually connects two wells - vertical elements of the sewer pipes infrastructure.

The detail of the process and its steps differ depending on the standard in force in a given country or region. Nevertheless, the main idea of the process is common to all standards. For example, the standard described by MSCC4 [11] defines the process carried out using a device equipped with a camera, selected depending on the size of the pipe and the expected water filling inside. Small cameras are used for the smallest pipes, pushed by a cable transmitting video material (the so-called push camera). For larger pipes, controlled travelling robots are used, optionally with a raised structure that allows the camera's centring in pipes of larger diameter. For pipes with an expected high level of water inside, floating robots are used (eg. *Proteus Float Raft*), and for the largest ones, prototypes of flying drones are being developed.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_27

The nomenclature of observations may differ depending on the applicable standard. Particular standards[1] also differently define the degree of detail of the description of a given observation, while the defects and structural elements that are described by these observations are the same for all standards. Based on this, a high-level part of the description of these observations can be distinguished, which is common to all standards. Among the observations, we can distinguish those corresponding to defects, structural elements and the conditions prevailing inside.

2.2 Machine learning methods used in the inspection process

J.B. Haurum and T.B. Moeslund [5] analyzed the results of publications on the automation of visual inspection of sewage networks from the last 25 years. Initial work is based on image analysis using methods such as morphological operations [16], oriented gradient histogram (*HOG*) [4], and support vector machines (*SVM*) [8]. Some of them deal with the subject of segmentation of some classes (e.g. cracks) using classical methods of image segmentation.

Along with the development of convolutional networks, subsequent works indicate their use for image analysis, initially using classification, object detection, and segmentation. The authors point out the problem of comparing works by differentiating the data set, the number of classes and considered metrics. For this reason, it is difficult to determine the best solution at a given moment, therefore, in the following part, selected works using deep neural networks describing the latest solutions from 2017-2020 will be analyzed.

M. Wang and J.C.P Cheng [17] proposed a solution based on the detection of observations in photos using the *Faster R-CNN* model. Their dataset included photos containing 4 classes of observations. A year later, the same authors [18] propose usage of the proprietary network architecture called *DilaSeg* for semantic segmentation of three classes of observations. The dataset contains the extracted video frames and segmentation masks for each of the photos. This publication shows an increase in efficiency and a decrease in inference time compared to the *FCN-8s* architecture. The authors point out that the cost of computation is rarely taken into account in the work so far, and this is a key factor that should be taken into account when implementing a solution for embedded devices.

D. Meijer et al. [9] propose a solution that uses image classification and detects 12 different classes of observations. The dataset contains photos taken with the same camera from 30 various inspections, 0.8% of these photos show defects. The publication uses the proprietary convolutional network architecture, the authors have shown that it achieves higher efficiency than the previous solutions. The emphasis was placed on the validation of the solution. It was proposed to introduce new metrics so that it was possible to assess the measure of possible performance improvement in real scenarios in addition to the classification effectiveness.

[1] For example, the Polish standard PNEN13508 or the American NASSCO PACP-6

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_27

Q.Xie et al. [19] proposed using a two-level hierarchical deep convolutional network for the classification of sewage network defects. The first binary level distinguishes between images representing all the considered defects from those without defects, and the second distinguishes between the considered defects. Both models share the network structure in addition to the last output layer, and the network architecture itself is a proprietary solution containing 3 convolutional layers, 3 of *max-pooling* type and 3 fully connected. The dataset consists of photos with 16 types of defects, while only the 6 most common ones were selected for training the model. The presented network results were better than the knowledge transfer using popular pre-trained models such as *VGG-16*, *Inception-V3*, and *Resnet*.

Kunzel et al. [6] describe the solution of semantic segmentation of a full scan of a network segment made with the 360° camera. The image is considered a high-quality image made of multiple 360° photos showing the entire tube unfolded onto a flat area. Successive clippings of this photo are transferred to the model. The network structure is based on *FRRN* [12] architecture. The dataset is a scan of 111 pipes with a length of 4.6 km, 6 classes of observation were considered.

The solution presented by Yin et al. [20] uses the *YOLOv3* model to detect objects. The dataset contains photos with 6 classes of observation. The authors indicate that this solution is able to analyze video in real-time and surpasses previous solutions both in terms of detection efficiency and execution time. The emphasis was also placed on validating the entire video material's solution, not just on individual frames.

Table 1: Comparison of image analysis methods in the problem of observation detection.

Paper	Classic methods			Methods using neural networks		
	Morphological operations	HOG	SVM	Classification	Object detection	Semantic segmentation
[4]	X	X	X			
[8]	X		X			
[16]	X					
[17]					X	
[18]						X
[9]				X		
[19]				X		
[6]						X
[20]					X	

The methods used in selected works have been collected in the Tab. 1. Currently, no works have been found that would deal with the topic of implementing such a solution for embedded devices. The hardware platform is omitted in the works, it is mentioned that a PC, supercomputer or cloud services were used.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_27

The emphasis is on algorithms, not the possibility of their implementation on the target platform.

3 Methodology

The research work aims to verify the feasibility of implementing a solution for the automatic detection of observations in sewage networks based on video material for embedded devices. The methodology of the research is shown in Fig. 1.

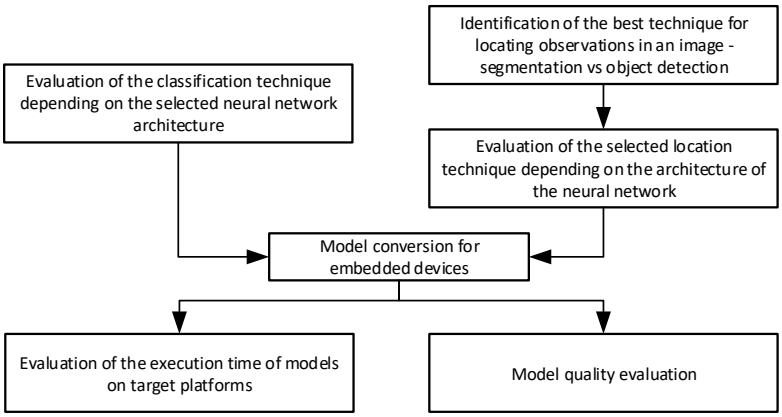


Fig. 1: The course of subsequent stages of research

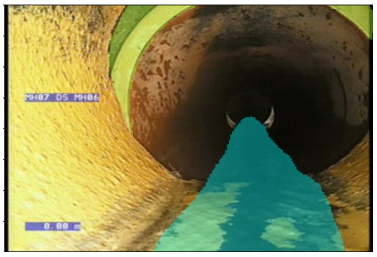
The dataset used in the research had over 20,000 photos. It contained two types of annotations - polygon coordinates defining the position for 22 classes (examples in Fig. 2) and classification labels for 5 classes (some examples are presented in Fig. 3).

Classes containing location information in the form of polygon coordinates have been grouped into nine more general classes because of their visual similarity. The assignment and the number of classes obtained in this way are presented in the Tab. 2. Due to the conditions inside the pipes, the picture quality is mostly poor. Many of them are fuzzy and out of focus, yet still contain human-readable information.

4 Selection of neural network architecture for the classification problem

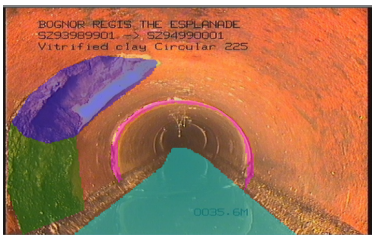
The neural network architectures examined in this work were selected based on their effectiveness on the *ImageNet* dataset. Additionally, the number of model parameters was also taken into account, which translates into its complexity.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_27



(a) Photo example with annotations: water and defective cross joint

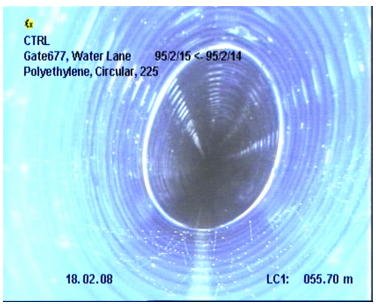
(b) Photo example with annotations: defective transverse joint and axial cracks without discontinuity



(c) An example of a photo with annotations: water, joint, built-in connection and attached sediments

Fig. 2: Examples of data set visualization with annotations in the form of polygon coordinates defining objects' location. Light blue is the color of water, light green is the defective transverse joint, the dark blue is the added joint, the red is the axial fracture without breaking the continuity, the pink is the joint, and the green is the sticky sediment.

(a) Example of a photo with a right deviation label (same as for other deviations)



(b) Example of a photo with a deformation label

Fig. 3: Examples of photos from the dataset with classification labels

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_27

Table 2: The form and number of grouped classes with annotations in the form of polygon coordinates defining the location of objects

Grouped class	The class in the original dataset	Count
fracture	axial fracture without discontinuity	5523
	spiral fracture without breaking the continuity	
	round fracture without breaking the continuity	
	axial fracture with discontinuity	
	spiral fracture with discontinuity	
	round crack with discontinuity	
break	break	717
missing wall fragments	missing wall fragments	776
roots	independent, fine roots	2224
	pile roots	
	complex mass of the roots	
accumulation of material	attached settlements	6562
	postponed settlements	
	other obstacles	
faulty joint	defective longitudinal joint	2946
	defective cross joint	
	defective angle joint	
joint	joint	10062
connection	original connection	2175
	built-in connection	
	incoming connection	
water	water	13656

Therefore, the size and time of inference execution are important for implementation on embedded devices with limited resources. The most diverse models were selected from the available in the *Keras* module of the *Tensorflow* library - *InceptionResNetV2* [10], *Xception* [1] and *MobileNetV2* [15].

The averaged values of the metrics obtained for the trained networks are presented in the Tab. 3. The value of the $F1$ metric for the analyzed models is similar, but the *MobileNetV2* has the smallest network size, thus it will be used in further experiments as the architecture of choice for the image classification problem.

Table 3: Average metric values of the trained classification models.

Model	F1	Recall	Precision
InceptionResNetV2	0.80	0.78	0.81
Xception	0.79	0.74	0.85
MobileNetV2	0.81	0.78	0.84

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_27

5 Selection of the technique for locating the observations

This section describes the course and results of the research carried out in order to choose the technique of locating the observations on the photos. Two classes of solutions were considered - methods of object detection and semantic segmentation.

In the case of object detection, the possibility of running the learned model on embedded devices was taken into account. The lightweight architecture *YOLOv2* (You Only Look Once) [13] was selected for the study for the object detection problem. In the case of semantic segmentation, the architectures and frameworks were selected from the *segmentation-models* package, which implements selected networks using the *Tensorflow* library. The following network architectures were examined - *U-Net* [14] and *FPN* [7]. They were used with encoders - *ResNet101*, *EfficientNetB3* and *MobileNet*-V2. Among the architectures compared, better results were obtained using the *FPN* network, so in further experiments it will be used as the chosen architecture for the problem of semantic segmentation.

Table 4: Quality scores after converting the results from the models: *FPN* and *YOLOv2*.

Score	Semantic segmentation			Object detection
	FPN + MobileNetV2	FPN + EfficientNetB3	FPN + ResNet101	YOLOv2
Precision	0.59	0.63	0.69	0.89
Recall	0.89	0.89	0.89	0.22
F1	0.71	0.74	0.71	0.35

In order to select the localization technique, the results obtained for semantic segmentation and object detection were compared. The results are presented in the Tab. 4. In the comparison, networks that perform semantic segmentation fared much better. Therefore, in further experiments, only semantic segmentation will be considered as a localization technique.

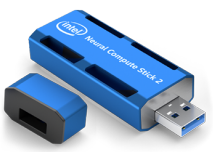
6 Implementation of trained models on embedded devices

This section presents the results of experiments with three hardware accelerators: Intel NCS2[2] (fig. 4a), Google Coral[3] (fig.4b), Nvidia Jetson Nano[4] (fig.4c). The neural network architectures selected in the previous sections were used to train models that need to be converted to run them with hardware accelerators. Both the metrics of the models after the conversion operation and their execution time

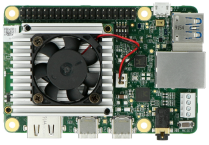
[2] https://movidius.github.io/ncsdk/
[3] https://coral.ai/products/
[4] https://developer.nvidia.com/embedded/jetson-nano-developer-kit

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_27

were examined. The tools used and the target numerical representation of the models' quantized weights for individual accelerators have been collected in the Tab. 5. The versions of the tools used are presented in the Tab. 6.



(a) Intel NCS2 accelerator connected via USB



(b) Google Coral development kit

(c) Nvidia Jetson development kit

Fig. 4: Hardware accelerators used in research

Table 5: The tools used for model conversion and the target numerical representation of the quantized model weights for individual hardware accelerators.

Accelerator	Tools used	Representation of quantized weights
Google Coral	Tensorflow Lite Converter + Edge TPU Compiler	INT8
Nvidia Jetson Nano	Nvidia TensorRT	FP16
Intel NCS2	Intel OpenVINO	FP16

Table 6: Versions of the tools used to convert the models.

Tool	Version
Tensorflow Lite Converter	2.2
Edge TPU Compiler	14.1.317412892
Nvidia TensorRT	7.0
Intel OpenVINO	2020.3.194

The values of the studied models' metrics before and after conversion are presented in the Tab. 7 for segmentation and in Tab. 8 for classification. No metric values for *FPN* are given with *EfficientNetB3* scaffold after converting to *Google Coral* because the compilation of the model to an executable form with *Edge TPU* could not be performed due to an internal compiler error. In the case of semantic segmentation, the values of the Intersection-over-Union (IOU) metric was also calculated as the area of overlap divided by the area of union between the predicted segmentation and the original data.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_27

Table 7: Metrics of semantic segmentation models before and after conversion to executable form on individual hardware accelerators.

Model	Accelerator	IOU	F1	Recall	Precision
FPN + ResNet101	-	0.55	0.77	0.88	0.69
	Coral	0.18	0.61	0.53	0.73
	Jetson Nano	0.55	0.78	0.88	0.70
	NCS2	0.55	0.78	0.88	0.70
FPN + EfficientNetB3	-	0.55	0.74	0.89	0.63
	Coral	-	-	-	-
	Jetson Nano	0.55	0.74	0.89	0.63
	NCS2	0.52	0.71	0.90	0.59
FPN + MobileNetV2	-	0.48	0.71	0.89	0.59
	Coral	0.48	0.72	0.84	0.63
	Jetson Nano	0.48	0.71	0.89	0.59
	NCS2	0.48	0.70	0.90	0.57

Table 8: Classification model metrics before and after conversion to executable form on individual hardware accelerators.

Model	Accelerator	F1	Recall	Precision
MobileNetV2	-	0.81	0.78	0.84
	Coral	0.80	0.76	0.85
	Jetson Nano	0.81	0.78	0.84
	NCS2	0.81	0.78	0.83

Finally, the models were tested on the representative video showing a recording of a complete inspection of a section of the sewage network. For each model, the average analysis time of a single video frame from the test recording is calculated using the formula:

$$t = \frac{\sum_{i=1}^{N} t_i}{N}, \tag{1}$$

where N is the number of frames in the video and t_i is the time to analyze ith frame. The procedure was repeated for each tested hardware accelerator.

In the case of *Google Coral* and *Nvidia Jetson*, the trained models after conversion were run on dedicated development kits, while *Intel NCS2* was connected to the *Raspberry Pi 3B* minicomputer. On each platform, the video was played from a file using *OpenCV*. A comparison of resources available on all platforms is presented in the Tab. 9.

Tab. 10 shows the average time of analysis of one frame using the analyzed segmentation models, the Tab. 11 shows the classification model results. Missing results for the *FPN* with *EfficientNetB3* and platform *Google Coral* model are due to the same build error when examining segmentation metrics. Also noteworthy is the much longer execution time for segmentation models on *Google Coral*, due to the fact that some of the operations performed within the model

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_27

Table 9: Comparison of the available resources of the platforms on which the experiments are performed.

Hardware platform	CPU	RAM
Google Coral Dev Board	ARM Cortex-A53 4 x 1,8GHz	1GB LPDDR4
Nvidia Jetson Nano	ARM Cortex-A57 4 x 1,43 GHz	4GB LPDDR4
Raspberry Pi 3B	Broadcom BCM2837 4 x 1,2 GHz	1GB LPDDR2

cannot be compiled for execution on *Edge TPU* and are instead performed on the CPU.

Table 10: Average time to analyze a single frame using trained segmentation models on each platform.

Model	Hardware platform	Time [ms]
FPN + ResNet101	Google Coral Dev Board	4036
	Nvidia Jetson Nano	234
	Raspberry Pi 3B + Intel NCS2	415
FPN + EfficientNetB3	Google Coral Dev Board	DNR
	Nvidia Jetson Nano	264
	Raspberry Pi 3B + Intel NCS2	480
FPN + MobileNetV2	Google Coral Dev Board	3954
	Nvidia Jetson Nano	121
	Raspberry Pi 3B + Intel NCS2	324

Table 11: Average time to analyze a single frame using a trained classification model on each platform.

Model	Hardware platform	Time [ms]
MobileNetV2	Google Coral Dev Board	7.21
	Nvidia Jetson Nano	95.42
	Raspberry Pi 3B + Intel NCS2	44.65

For comparison, additional runtime tests were carried out on one of the platforms without the use of acceleration, using only the CPU. For this purpose, *Nvidia Jetson* was used due to the largest amount of RAM available. The execution time was converted into the number of frames that could be analyzed per second - *FPS* (frames per second), and the results are presented in Fig. 5 and Fig. 6 respectively for segmentation and classification models.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_27

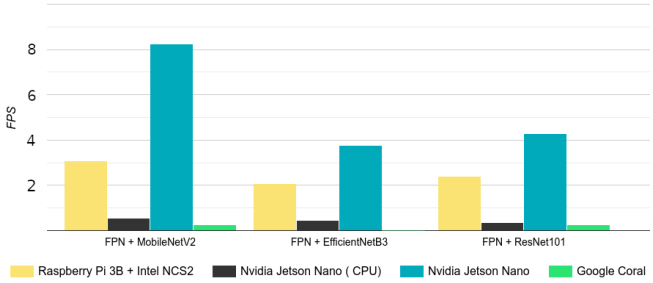


Fig. 5: The number of frames per second that can be processed for the

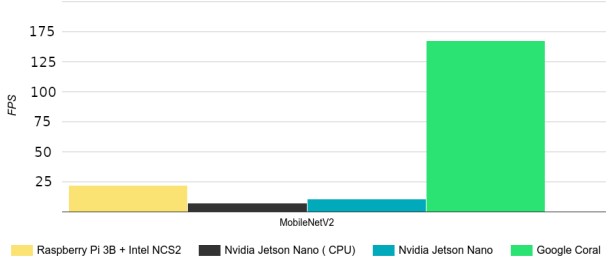


Fig. 6: The number of frames per second that can be processed for the classification model

7 Summary and future work

The values of the model metrics after conversion to the executable form with the use of hardware accelerators and the obtained execution times indicate that it is possible to implement the discussed solution on embedded devices, moreover, the implementation would not be possible without the use of hardware acceleration, as shown in the graphs in Fig. 5 and Fig. 6. Among the tested devices, only *Google Coral* is not suitable for use for too long of the segmentation models. All other configurations of models and accelerators have a total execution time of less than 1s, which is the limit value, since the [20] assumption is made that the observations are visible on the video material for at least one second.

Among the examined methods of locating objects in photos - object detection and semantic segmentation, the use of segmentation gives better results. From the investigated *FPN* and *U-Net* architectures, better results were obtained for *FPN*. Despite the smallest size, the best of the studied architectures for the classification problem turned out to be *MobileNetV2*.

Directions for further development are possible both in the context of the detection mechanism and validation of the usability of the solution. This paper does not deal with the aspect of calculating the water level, although water is detected as one of the segmentation classes. A mask that is detected with fairly high accuracy (metric $IoU = 0.79$ for $FPN/ResNet101$) can be used to calculate the water level using conventional image analysis methods. In addition to the

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_27

water level detection, based on segmentation masks, it is possible to calculate other observations' parameters, such as the connection diameter.

Another aspect is the more detailed classification of the classes of observations detected. The classes present in the original dataset have been grouped into more general ones, so having the results of the grouped class segmentation, one can cut out a part of the original photo limited by the segmentation mask for a given class and forward it to smaller, more specialized classifiers that will refine the detection results.

An important element that can be further developed is the extension of validation of the solution and conducting experiments with camera operators. Dividing them into two groups - with and without software support-and the subsequent analysis of their work results will help answer the question of how the generated prompts affect the quality and time of inspections.

Acknowledgment

The research presented in this paper was partially supported by the funds assigned to AGH University of Science and Technology by the Polish Ministry of Science and Higher Education.

References

1. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1800–1807 (2017)
2. Dirksen, J., Clemens, F., Korving, H., Cherqui, F., Gauffre, P., Ertl, T., Plihal, H., Müller, K., Snaterse, C.: The consistency of visual sewer inspection data. Structure and Infrastructure Engineering (2013)
3. Gay, L.F., Bayat, A.: Productivity Improvement of Sewer CCTV Inspection through Time Study and Route Optimization. Journal of Construction Engineering and Management (2015)
4. Halfawy, M., Hengmeechai, J.: Automated defect detection in sewer closed circuit television images using histograms of oriented gradients and support vector machine. Automation in Construction (2014)
5. Haurum, J., Moeslund, T.: A Survey on Image-Based Automation of CCTV and SSET Sewer Inspections. Automation in Construction (2020)
6. Kunzel, J., Werner, T., Eisert, P., Waschnewski, J.: Automatic analysis of sewer pipes based on unrolled monocular fisheye images. In: 2018 IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 2019–2027 (2018)
7. Lin, T., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 936–944 (2017)
8. Mashford, J., Rahilly, M., Davis, P., Stewart, B.: A morphological approach to pipe image interpretation based on segmentation by support vector machine. Automation in Construction (2010)
9. Meijer, D., Scholten, L., Clemens, F., Knobbe, A.: A defect classification methodology for sewer image sets with convolutional neural networks. Automation in Construction (2019)

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_27

10. Nguyen, L., Lin, D., Lin, Z., Cao, J.: Deep cnns for microscopic image classification by exploiting transfer learning and feature concatenation. pp. 1–5 (05 2018)
11. Orman, N., Lambert, J.E.: Manual of Sewer Condition Classification 4th Edition. WRc (2004)
12. Pohlen, T., Hermans, A., Mathias, M., Leibe, B.: Full-resolution residual networks for semantic segmentation in street scenes. In: Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on (2017)
13. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 779–788 (2016)
14. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. pp. 234–241. Springer International Publishing, Cham (2015)
15. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4510–4520 (2018)
16. Swarnalatha, P., Kota, M., Resu, N.R., Srivasanth, G.: Automated assessment tool for the depth of pipe deterioration. In: 2009 IEEE International Advance Computing Conference. pp. 721–724 (2009)
17. Wang, M., Cheng, J.: Development and improvement of deep learning based automated defect detection for sewer pipe inspection using faster r-cnn. In: Advanced Computing Strategies for Engineering. vol. 10864, pp. 171–192 (2018)
18. Wang, M., Cheng, J.: Semantic segmentation of sewer pipe defects using deep dilated convolutional neural network. 36th International Symposium on Automation and Robotics in Construction (2019)
19. Xie, Q., Li, D., Xu, J., Yu, Z., Wang, J.: Automatic detection and classification of sewer defects via hierarchical deep learning. IEEE Transactions on Automation Science and Engineering **16**(4), 1836–1847 (2019)
20. Yin, X., Chen, Y., Bouferguebe, A., Zaman, H., Al-Hussein, M., Kurach, L.: A deep learning-based framework for an automated defect detection system for sewer pipes. Automation in Construction (2020)

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_27