APPLIED RESEARCH

Developing a Free and Open-Source Semi-Automated Building Exterior Crack Inspection Software for Construction and Facility Managers

PI KO, SAMUEL A. PRIETO, (Member, IEEE), AND BORJA GARCÍA DE SOTO, (Member, IEEE)

S.M.A.R.T. Construction Research Group, Division of Engineering, New York University Abu Dhabi (NYUAD), Experimental Research Building, Abu Dhabi, United Arab Emirates

Corresponding author: Borja García de Soto (garcia.de.soto@nyu.edu)

ABSTRACT Inspection of cracks is an important process for properly monitoring and maintaining a building. However, manual crack inspection is time-consuming, inconsistent, and dangerous (e.g., in tall buildings). Due to the development of open-source AI technologies, the increase in available Unmanned Aerial Vehicles (UAVs) and the availability of smartphone cameras, it has become possible to automate the building crack inspection process. This study presents the development of an easy-to-use, free and open-source Automated Building Exterior Crack Inspection Software (ABECIS) for construction and facility managers, using state-of-the-art segmentation algorithms to identify concrete cracks and generate a quantitative and qualitative report. ABECIS was tested using images collected from a UAV and smartphone cameras in real-world conditions and a controlled laboratory environment. From the raw output of the algorithm, the median Intersection over Unions (IoU) for the test experiments are (1) 0.686 for indoor crack detection experiment in a controlled lab environment using a commercial drone, (2) 0.186 for indoor crack detection at a construction site using a smartphone and (3) 0.958 for outdoor crack detection on university campus using a commercial drone. These IoU results can be improved significantly to over 0.8 when a human operator selectively removes the false positives. In general, ABECIS performs best for outdoor drone images, and combining the algorithm predictions with human verification/intervention offers very accurate crack detection results. The software is available publicly and can be downloaded for out-of-the-box use.

INDEX TERMS Building inspection, construction automation, deep learning, Detectron2, image processing, segmentation.

I. INTRODUCTION

Depending on the maintenance plan, the inspection of buildings is a time-consuming, repeating process with intervals between 5 to 10 years [1]. For the most part, inspections are done manually by inspectors. It is a labor-intense task when done for large buildings (or a portfolio of buildings). Moreover, it is dangerous if the inspection of the exterior of the building is required for high-rise buildings because,

The associate editor coordinating the review of this manuscript and approving it for publication was Larbi Boubchir.

traditionally, it requires an inspector to abseil down over different sides of a building [2]. Also, the manual visual examination is usually done by experienced operators with specialized surveying tools such as magnifiers, crack rulers, etc. Sometimes, the results can be inconsistent and suffer from human subjectivity [3]. Therefore, the manual inspection does not leave consistent computerized data (i.e., digital data), which can be used to compare the results of successive inspections over time later.

In recent years, there have been developments in commercial Unmanned Aerial Vehicles (UAVs) or drones and the

widespread availability of smartphones. Moreover, there are also immense developments in open-source Artificial Intelligence (AI) systems. Therefore, it has now become possible to fuse all these technologies to implement a partially, if not fully, autonomous crack detection system which takes digital images from smartphone cameras, UAVs, or robots as input, then uses AI and image processing techniques to identify the location of the cracks and provides detailed quantitative and qualitative reports. There has been an increase in research utilizing image processing techniques and AI to detect and classify cracks with varying levels of success [4]. Nevertheless, many of these research experiments tend to be very technical or research-oriented and have not been adopted by mainstream construction and facility managers. This research aims to close the gap between theoretical research in crack detection and practical application in the field by developing an Automated Building Exterior Crack Inspection Software (ABECIS). ABECIS has been developed as an easy-to-use, free and open-source graphical crack detection software that runs cross-platform on all operating systems. The software only takes images of buildings (either taken by smartphone cameras, UAVs, or other robotic platforms) as input, detects the cracks based on user-adjusted parameters and automatically generates detailed quantitative and qualitative reports. Therefore, the construction and facility managers can simply select a folder with all the images of the building they want to inspect using ABECIS, and the rest of the process will be automated with minimum need for human intervention, allowing them to harness all the advantages of AI without the need to be technical or knowledgeable about it. ABECIS has been made publicly available through a GitHub repository [5]. Installation instructions can be found in [6]. The main elements and contributions of this paper are as follows:

- A novel open-source Automated Building Exterior Crack Inspection Software (ABECIS) for construction and facility managers has been developed.
- The software is not dependent on a specific platform for the origin of the information, making it accessible to the general public.
- The system provides an easy-to-use interface with a completely automated process that locates and segments cracks within the provided pictures.
- The detected cracks are analyzed to determine their type and their length.
- The software gives the possibility for the human operator to provide feedback in the process, increasing the overall performance of the method.
- Both indoor and outdoor experiments have been performed to test the validity of the software.
- The datasets, results, and software are open-source and have been made publicly available.

In the first part of this paper, an extensive literature review is done through relevant research papers in recent years, as well as a review of publicly available crack detection algorithms online and some industry research papers. Secondly,

the paper addresses the necessary background information on data collection with smartphones and drones, classification of building crack types, deep learning, and instance segmentation algorithms. The third part of this paper discusses the research methodology, lays out the theoretical assumptions, and explains the basic functionality of the software. This is followed by an implementation section, where the process of developing ABECIS, its functionalities and outputs are described. Afterward, the results from testing ABECIS with images taken (a) using a commercial drone in a laboratory environment with a mockup concrete wall, (b) using a smartphone camera in an actual construction site and real urban settings and (c) using a commercial drone outdoors on the university campus are evaluated and discussed. Finally, the conclusion, outlook, and acknowledgment are provided.

II. LITERATURE REVIEW

A literature review was done through a wide variety of conferences and journal publications to identify and summarize the state-of-the-art crack detection techniques and their situation at the time of this study. To focus the paper on the application and usability of these techniques on-site in the real world, this literature review discusses academic research papers (subsection II-A) and addresses the open-source code for crack detection that are publicly available (subsection II-B).

A. LITERATURE REVIEW OF ACADEMIC RESEARCH

A comprehensive summary of relevant papers addressing the use of image processing techniques (with and without the use of AI), with varying degrees of automation to detect defects on buildings or structures with various surface materials, published in recent years, were identified using a keyword-based search. This study identified the state-of-the-art research trends in crack detection and building monitoring techniques in construction in general, using the following keywords: "building inspection," "automation," "deep learning," "machine learning," "crack detection," "building monitoring," "segmentation," "image processing." The sources are selected based on clarity, the novelty in the methods presented, and their relevance to this study. Findings from that review are provided below, and a summary table with all papers reviewed is available in Appendix A (Table 4).

From the literature review and a quick inspection of Table 4 in Appendix A, cracks are some of the most common defects detected when inspecting building exteriors. Most of the approaches are semi-autonomous, meaning that at least the data acquisition during the process is manually commanded, or the methodology requires some manual input during the crack recognition process. The most popular platform is a drone (UAV) for data acquisition (i.e., taking images during the inspection).

Similar studies to the one proposed in this paper have been done. However, the proposed method addressed some

of the limitations. For example, within the approaches that use Artificial Intelligence (AI) for data processing, a few of the methods [7], [8], [9] use an R-CNN [10] approach for data processing. Ayele et al. [7] used images of the Skodsberg concrete bridge in Norway taken with a UAV to detect cracks. The length, width and areas of the cracks were found computationally. However, the study does not provide any datasets or code samples for replicating it. Yu et al. [11] and Cen et al. [12] presented machine learning-based approaches to detect cracks on the concrete surface of bridges. They used a UAV to collect the images to train the algorithm, but those studies do not account for automation in the data collection process. The approach proposed by Yu et al. [11] only provides the classification and localization of the cracks instead of the actual instance segmentation needed to further analyze the crack. In the work of Cen et al. [12], they evaluated how the size of the filter window and the setting of the threshold can affect the results of the crack detection process. However, they only provide a classification of images without a precise localization or instance segmentation of the cracks. Jo and Jadidi [13] presented a system that used multi-layered image processing and deep belief learning to classify road surface cracks. They indicate that their method can be used with a UAV. However, they do not test this particular data collection approach, showcasing just RGB and infrared images collected with a smartphone camera (with the help of a FLIR one attachment). The first step in their processing stage was a multi-layered traditional image filtering, applying different filters such as Gabor, Otsu, Retinex and Prewitt filters in order to extract features of segmentation, edges and background. The features were then sent to a Deep Belief Network (DBN) classifier [14]. Their system is only able to classify images, not actually detect and segment the cracks. Within the AI spectrum, CNN-based approaches are the most popular [15], [16], [17], [18], [19], [20]. Silva et al. [15] used YOLOv4 [21] classifier to classify and detect potholes and cracks in the pavement. Both Yeum et al. [16] and Dorafshan et al. [17] used a CNN-based AlexNet approach [22]. Yeum et al. [16] focused on validating Regions of Interest (ROI) around bridge metal joints for further inspection of the welding areas. Dorafshan et al. [17] compared the performance of the CNN approach against traditional image processing filters, proving the superiority of the AI-based approaches when it comes to crack detection. Chen et al. [23] proposed a two-step deep learning method for the automated detection of façade cracks. They used a UAV to collect the images and put them through a first classification step where a CNN model was used to classify regions of the image as crack or non-crack. Based on a U-Net neural network model, the second step extracts the pixels corresponding to cracks from the crack-labeled regions. They also compared how their algorithm performs against traditional image processing methods. Despite their method being reliable for crack segmentation, they do not provide any analysis of the detected crack.

Some methods focus on traditional image processing techniques without the involvement of AI techniques. For those focusing on crack detection, the data processing involves multiple algorithms based on edge detection [24], [25], [26], [27], [28], [29], [30], [31]. For the most part, methods based on edge detection cannot provide results as reliable as those with AI data processing. For instance, Liu et al. [26] performed a 3D reconstruction of bridge piers with the help of Structure from Motion (SfM) techniques to later project the located crack in 3D. Although they thoroughly analyzed the detected crack, they did not provide quantitative results on how well the algorithm detected said cracks (i.e., there is no information on false negatives). Other approaches do not focus on the precise localization of cracks and rely more on qualitative surface evaluation, such as the absence of paint [32] or variation of defects by measuring the similarity between two pictures [33].

The approaches mentioned above are either manual or semi-autonomous in the sense that none of them consider the automation of the data collection process. Within the few fully autonomous approaches that have been found, Jo et al. [34] proposed a methodology that would perform fully autonomous inspections on concrete surfaces. The work is based on an autonomous control over the UAV during the data collection, considering different agents (e.g., wind) to control the path of the UAV while both RGB and infrared images are being collected. Both images are later processed by a machine learning agent to identify cracks by combining the detections in both spectrums. Nonetheless, they do not provide any results of the proposed methodology. Kang and Cha [35] proposed an ultrasonic beacon system to replace the role of the GPS to provide autonomy to the UAV during the data acquisition. The video data taken with the UAV is later processed with the help of a deep convolutional neural network (CNN) to detect concrete cracks. Their method can successfully locate a small set of cracks present on the concrete floors. However, their approach could only detect and locate cracks without performing actual instance segmentation that could be used to further analyze the crack to provide more data. Li et al. [36] used an autonomous UAV-based system to detect cracks in mining slopes. They compared some of the most traditional edge detectors (i.e., Prewitt, Sobel, Canny) with a U-Net approach to detect rocks and cracks amongst the slopes. Even though their study highlighted the performance of each method for edge detection, it did not provide an assessment of the cracks detected during the experiment or the outcomes of the project; therefore, it is hard to evaluate the effectiveness of the proposed approach.

B. INDUSTRY RESEARCH AND AVAILABLE CRACK DETECTION CODE

To consider the development in the industry and the accessibility of these tools in the public domain, the literature review was extended to include code repositories on GitHub
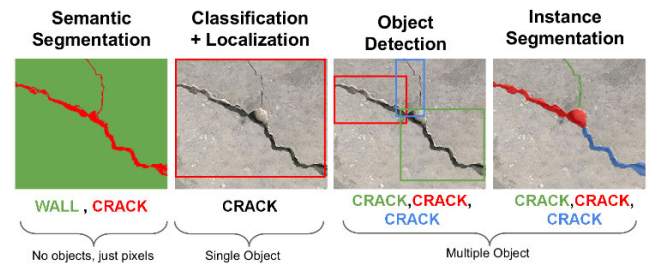
TABLE 1. Summary of review on the crack detection code and industry/commercial products available online.

Reference	Type	Code accessible to General Public	For cracks detected on...	Graphical User Interface Available	Dataset available	Pre-trained model available	Generates Report
Fan Yang et al. [38]	GitHub	✓	Pavement		✓	✓	
Ruiz et al. [39]	Publication	✓	Concrete		✓		
Cha et al. [40]	GitHub	✓	Concrete		✓		
Dais et al. [41]	GitHub	✓	Masonry		✓	✓	
Zou et al. [42]	GitHub	✓	Pavement		✓	✓	
Canon [37]	Publication		Concrete				
ABECIS (this study)	Publication and GitHub	✓	Concrete	✓	✓	✓	✓

FIGURE 1. General types of object detection algorithms for images.

and online writings by the industry, such as Canon [37]. The keyword-based search on search engines (Google, Duck-DuckGo) and GitHub using the keyword "crack detection software" led to the resources summarized in Table 1.

There are existing codes for crack detection available to the public. Except for Zou et al. [42] and Dais [41], that provide their dataset and pre-trained models online so that the public can download their code and run the crack detection algorithms on their own for specific types of cracks (for masonry surface cracks and pavement cracks only), most available code does not offer pre-trained models. Therefore, one must perform the training before the algorithm can be used. Training a machine learning algorithm by oneself has both advantages and limitations. As an advantage, one can fine-tune the parameters, such as the number of epochs, to make the model suitable for the project's specific needs. However, as a limitation, setting up the computing environment and requirements to train the machine learning model may be difficult for unfamiliar individuals. Therefore, everyone cannot utilize this crack detection algorithm unless they can set up the computing environment themselves. It is worth noting that there are several commercially available crack detection algorithms, such as those developed by Dynamic Infrastructure [43] and T2D2 [44, p. 2]. They are proprietary software developed by businesses and cannot be freely accessed. Their technology remains proprietary, closed source, and not accessible for obvious reasons; hence cannot be included or evaluated in this study.

Nevertheless, even if the code and the data are provided online to be accessible by interested stakeholders (e.g., construction and facility managers and building inspectors), those resources are still very limited in many ways. For example, (1) they require the user to be able to operate them using a command-line interface since many of the Python scripts have to be run in the terminal, (2) they assume that the users can set up several things on their own – for instance, there are no instructions on these GitHub pages about how to set up the required machine learning libraries, except simply stating that they are required, e.g., Requirements – PyTorch, and (3) they do not generate a final report summarizing the cracks identified and the analyzed data, which is one of the most important information for construction and facility managers.

Many of the construction and facility managers that are not familiar with coding or do not interact with computers daily might not have the required technical literacy to operate the command line interface or figure out how to set up machine learning libraries on their own, and they would benefit by having a more user-friendly way to do the crack detection and analysis. To address that, ABECIS was developed as a freely accessible and user-friendly crack detection software with an intuitive Graphical User Interface (GUI). The main purpose is to utilize and modify existing algorithms with our own data to produce a readily accessible solution for facility managers, which currently does not exist. ABECIS is designed to work out-of-the-box, with minimal setup by the user. There is no need to go and download the pre-trained model and set up the software directory since it will perform the setup on its own. Moreover, once the input images are selected, the rest of the tasks, from the analysis to the report generation, are also done autonomously. For the parts of the cracks detected with uncertainty, the user will have a choice to intervene manually and reject the wrong identifications. Afterward, a qualitative and quantitative report will be generated along with annotated crack images.

C. CRACK DETECTION ALGORITHMS

Many algorithms currently exist for the detection of objects. However, almost all the algorithms can be generally classified into 4 broad categories. These algorithms can be classified into (a) Semantic Segmentation, (b) Classification and Localization, (c) Object Detection and (d) Instance Segmentation algorithms (Figure 1). Semantic Segmentation classifies every pixel in the image into classes (e.g., wall and crack). Classification and Localization algorithms only show the location of a single object. Object Detection algorithms are superior to the former two since they detect each object (e.g., in Figure 1) and its location, but not shape and size. Instance Segmentation is the most powerful technique among the four because it provides information about each object's location, shape and size. Therefore, for this study, instance segmentation was used as the main detection algorithm.

Hafiz and Bhat [45] did a survey on the state-of-the-art instance segmentation algorithms. Their study indicated that notable instance segmentation algorithms include PANet, YOLACT, and Mask R-CNN. PANet [46] and YOLACT [47] algorithms have their code on GitHub repositories but are no

longer actively maintained and updated. PANet uses an adaptive feature pooling technique to funnel useful information in each feature level through the neural network for better mask prediction. YOLACT achieves real-time instance segmentation by breaking the instance segmentation into two subtasks. Nevertheless, non-updated code makes PANet and YOLACT non-ideal due to the deprecation of dependent Python packages over time. Regarding Mask R-CNN, its official GitHub repository [48] states that Mask R-CNN has been deprecated and is now implemented into an instance segmentation framework called Detectron2 [49, p. 2], which is actively maintained by Meta Research. The comparison among different instance segmentation algorithms is beyond the scope of this study, and given the current state of Detectron2 regarding updateability and overall reported performance, the authors decided to use Detectron2 as the algorithm used to develop ABECIS.

III. BACKGROUND

The ABECIS system is designed in a very flexible way such that it can accept a variety of inputs produced by different methods of data collection – from very accessible, simplistic methods, such as photos taken using smartphones, to complex, professional methods such as photos taken using industrial drones such as the DJI Matrice 300 RTK (Figure 2), or autonomous robotic platforms as suggested by [50]. The only requirement is that the input must be an RGB image. For this study, two methods for image collection, namely cameras on (1) an industrial drone and (2) smartphones were used.

FIGURE 2. DJI Matrice 300 RTK [51].

Although the ABECIS system can function independently of the input method used, the quality of results and analysis produced may differ depending on the quality of input images from different methods. For instance, the images taken by the industrial drones will have more metadata that can be extracted for further analysis (e.g., the distance of the drone from the wall), which can be used to approximate the dimensions of the items of interest (in this case the length of a concrete crack). However, with the images taken by the smartphone, the data for the distance between the smartphone camera and the wall is missing; therefore, the dimensions of the crack cannot be approximated. Moreover, the difference in methods will also affect the accessible locations of the images for crack detection (e.g., drones can facilitate capturing images for crack detection in the façade of a 30th-floor
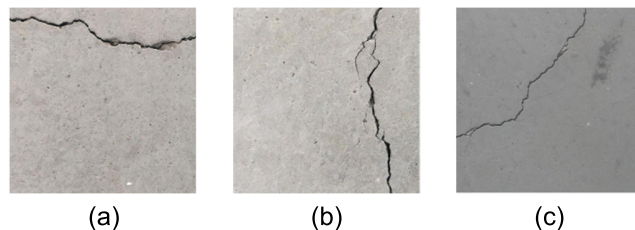
FIGURE 3. Different types of cracks classified in ABECIS: (a) horizontal, (b) vertical, and (c) diagonal from a horizontal perspective (source [53]).

building compared to a manual inspection using a smartphone).

A. CRACK CLASSIFICATION TERMINOLOGY

The classification of the concrete cracks for this study has been done in accordance with the ACI Concrete Terminology report by American Concrete Institute [52]. In particular, concrete cracks have been classified into 3 categories based on their orientation (Horizontal Crack, Vertical Crack and Diagonal Crack). Cracks that are either 180° or 90° to the horizontal are classified as horizontal and vertical cracks, respectively. Other cracks are classified as diagonal cracks (Figure 3). It is assumed that the images of the cracks are upright and not rotated, taken from a horizontal position, and with a camera orientation perpendicular to the plane being inspected.

B. DEEP LEARNING AND CRACK DETECTION

Deep Learning is a technique in Artificial Intelligence (AI) that uses artificial neural networks and is widely used to classify images [54]. Traditionally, crack detection algorithms use non-AI techniques such as Local Binary Patterns and shape-based algorithms [54]. Since 2012, deep learning algorithms have gained popularity and outperformed traditional detection methods [22].

1) OPEN-SOURCE DEEP LEARNING FRAMEWORKS

There has been a significant increase in publicly available open-source deep learning software libraries in recent years, allowing users to quickly create their custom deep learning models. For this study, a computer vision technique called instance segmentation - the task of detecting the distinct objects of interest appearing in the images – is used. Many popular instance segmentation algorithms already exist, such as PixelLib [55]. However, Detectron2 [49, p. 2], a state-of-the-art instance segmentation algorithm developed by Meta (formerly Facebook) Research, has been used for this study. Detectron2 is an improvement on Mask R-CNN and is updated more frequently than other algorithms [49, p. 2], [56].

2) PREPARATION OF DATASET FOR TRAINING

The algorithm must first be trained to detect cracks using instance segmentation. The training requires many

pre-labeled images. This dataset can be obtained by oneself or using publicly available datasets. Özgenel and Sorguç [53] have publicly shared their dataset of concrete crack image data, which has been used in this research to train the model for concrete wall crack detection. For this study, the authors did a manual polygonal annotation of 800 images from [53] using LabelMe software [57], [58] (e.g., Figure 4) so that those images could be used for instance segmentation training during the development of the model. A sample of the labeled images developed for this study is publicly shared and available at [59].

FIGURE 4. Polygonal annotation of images by the authors using LabelMe [57] and used for model training.

IV. RESEARCH METHODOLOGY

A. REAL-WORLD CONSTRAINTS AND ASSUMPTIONS

Numerous constraints and situations can interfere with the crack detection algorithm, producing false results in the real world. Image-processing results can be heavily affected depending on the camera angle, distance to the wall, shadows in images, and blind spots. For this study, the ABECIS system assumes the following from the images captured from the real world:

- The images are upright, not rotated and taken from a horizontal position.
- The images only contain the object of interest (wall) and no other objects.
- All the crack-like features, whether naturally occurring or intentionally man-made (e.g., some architectural features might contain gaps or openings in the walls for aesthetic purposes), will be flagged as possible cracks and verified by humans afterward.
- Camera orientation is perpendicular to the plane of the object under inspection.

These assumptions are made because unless the images taken by inspectors or drones are preprocessed before applying the crack detection algorithm, a high number of undesirable false-positive and false-negative errors are likely to occur [16].
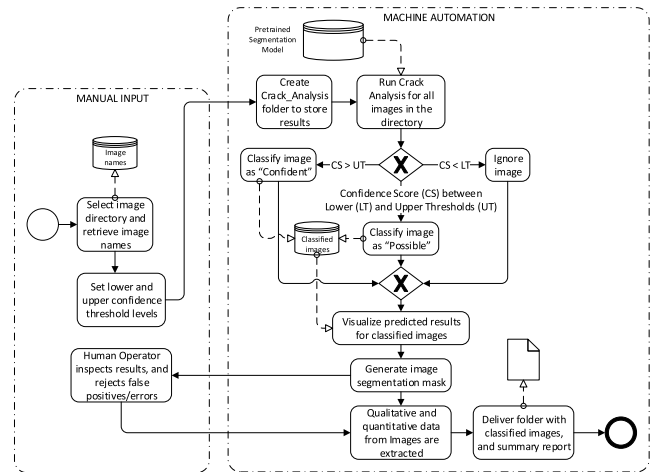

FIGURE 5. System architecture of the automated building exterior crack inspection system (ABECIS) (expanded from [60]).

B. SYSTEM ARCHITECTURE

The overall system architecture for ABECIS is an expansion of the authors' work presented in [60]. It includes human input and machine automation parts (Figure 5). Initially, the system will prompt the user to select a directory on their computer containing all the images to be analyzed for cracks. Afterward, the user will need to adjust the lower and upper confidence score threshold. Confidence scores are numbers between 0 and 100 used to describe how well the algorithm thinks it has detected a crack (100 being a perfect detection). The threshold values set a cutoff point such that identified cracks with a confidence score below the threshold value will be rejected. However, there may be some images of non-cracks for which the software is highly confident that they are cracks, resulting in false positives. In some scenarios, there might be objects which resemble cracks (e.g., dents or scratches on the walls, intentional man-made crack-like grooves, such as small gaps between two walls). For such cases, the algorithm might not be certain of the cracks identified and might require the assistance of a human operator to make a final decision (i.e., verification/intervention).

Therefore, ABECIS has two threshold values (upper and lower) for the confidence scores, which work as shown in Figure 6. Using the pre-trained segmentation model on all the available images, any images containing crack detection confident scores higher than the upper threshold will be classified into a folder named "Confident." In contrast, identified cracks that fall within the upper and lower threshold will be classified into a folder named "Possible". Otherwise, the identified cracks are rejected. These two parameters are fully adjustable by the human operator and are to be decided by their judgment based on the environmental situation and difficulty of the crack detection task. Such threshold values are subjective; however, they are necessary to give more control to users depending on their specific use case.

After the cracks are identified and classified into the "Possible" and "Confident" folders, the detection results are
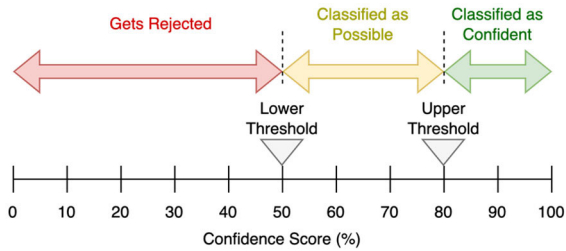
FIGURE 6. Confidence threshold in ABECIS.

FIGURE 8. General overview of the architecture of the used segmentation algorithm.
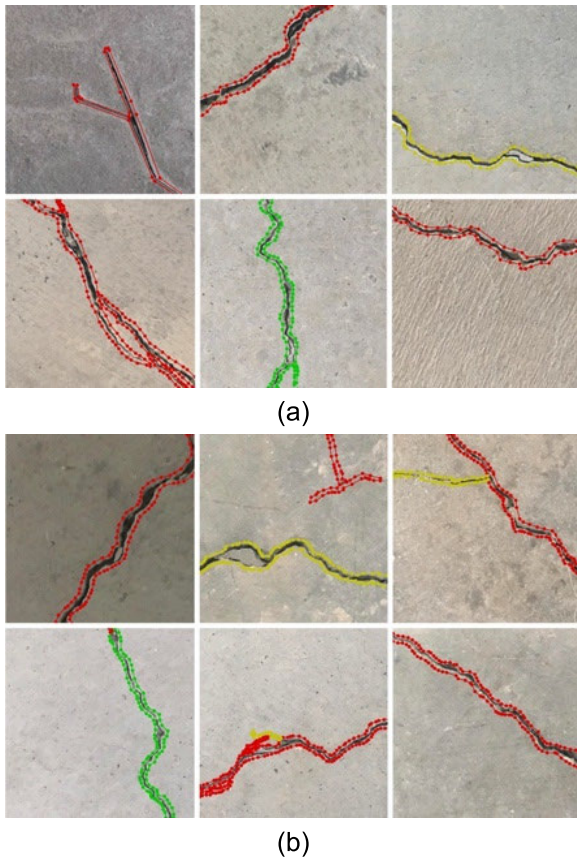

FIGURE 7. Examples of images used for the (a) training and (b) testing datasets (red = diagonal cracks, green = vertical cracks, yellow = horizontal cracks). The dataset was prepared via polygon labeling by the authors in LabelMe using raw images from [49, p. 2]. A subset of the labeled images developed for this study can be found in [59].

ready to be visualized, and all the segmented masks of the images (black and white binary images where white represents the crack regions) are generated for quantitative analysis (e.g., length and area of the crack). The process is done completely autonomously by the software. Afterward, the human operator can use the Graphical User Interface (GUI) to inspect the crack detection results in the "Possible" and "Confident" folders and reject any false positives, if any. Finally, taking this human input into account, the software can generate a detailed report for individual images with the types of cracks, including the time the image was taken (identified from metadata for monitoring purposes) and the length and
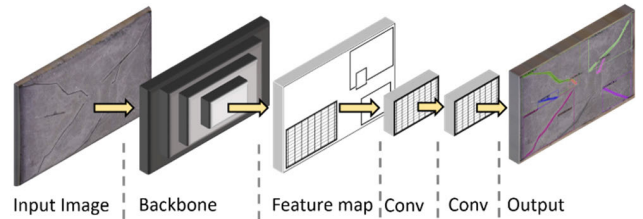
area of the wall cracks in pixels and relative percentages. Therefore, the algorithm contains two instances where human intervention is required (initially to set the threshold parameters and finally to verify the outputs). Outside this, the process is fully automated. It is worth mentioning that ABECIS could run completely autonomously with predefined thresholds. However, construction places are extremely dynamic environments and present a wide variety of conditions and scenarios. Allowing the human operator to provide some minimum input on the system allows for more robust and reliable results that can be used in real construction projects. ABECIS can be downloaded from the GitHub repository [5] and used by construction and facility managers and building inspectors with minimal setup using their own inspection images.

V. DEEP LEARNING MODEL AND SOFTWARE DEVELOPMENT

A. TRAINING THE INSTANCE SEGMENTATION NEURAL NETWORK

The image segmentation model was developed using 800 images and the Detectron2 library using the Özgenel and Sorguç [53] dataset labeled by the authors. The training dataset consists of 700 images, and the testing dataset consists of 100 images (i.e., a distribution of 87.5% for training and 12.5% for testing). Example images of the training and testing datasets can be seen in Figure 7.

The model was trained using COCO-Instance Segmentation, mask_rcnn_R_50_FPN_3x, the pre-trained model. It would take a long time to train a model from scratch; therefore, pre-trained models like these are used to speed up the process. Although Detectron2 contains more pre-trained models, such as C4 and DC5, FPN was chosen because it provides the best speed/accuracy tradeoff, according to Detectron2 documentation. A general overview of the architecture of the used model is shown in Figure 8. For detailed information on COCO-Instance Segmentation, readers are referred to [61, p. 2].

In terms of parameters, ABECIS uses the COCO-Instance Segmentation model in Detectron 2 with the parameters shown in Table 2 below.

The kernel size is the size of the convolutional filter and is set to a combination of (1, 1), (3, 3) and (7, 7) in pixels to capture small and large features in the image. Stride is the

TABLE 2. Detailed parameters and architecture of maskrcnn R50 FPN 3X.

Parameter Name	Value
Kernel Size	(1, 1), (3, 3), (7, 7)
Stride	(1, 1), (2, 2)
Padding	(1, 1), (3, 3)
Bias	False
EPS	1e-05
Number of Features	64, 256

step size of the filter and is set to (1, 1) and (2, 2) in pixels to capture both fine and coarse details in the image. Padding is the number of pixels added to the image to preserve the spatial dimensions and is set to (1, 1) and (3, 3) pixels. Bias is set to false to prevent overfitting. EPS, a small value that is used to prevent numerical instability or divide-by-zero errors, is set to 1e-05 to ensure numerical stability. The number of features, which are the number of output channels for the convolutional layer, is set to 64 and 256.

In addition to these parameters, the configuration file, cfg, is created by merging the parameters from the COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml file. The number of workers for the dataloader is set to 2; the weights are set to the checkpoint URL of the COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml file. The solver is set to have 2 images per batch and a base learning rate of 0.00025 with a maximum number of iterations set to 3000. The number of classes for the ROI heads is set to 3 and can be modified according to the specific application.

Cross-entropy loss [62] and accuracy [63] were calculated to measure how well the model performed. Cross-entropy loss of 0 indicates a perfect model and an accuracy of 1 indicates a perfect model, too. After training for 3,000 epochs, the cross-entropy loss of the model reached close to 0.1 (Figure 9a), and the accuracy was 0.97 (Figure 9b). This indicates that the trained model is reliable.

B. SOFTWARE FEATURES AND DEVELOPMENT

Once the image segmentation model was trained, a cross-platform software application was developed using Detectron2 [49] for the deep learning component, PyQt6 [64] for the Graphical User Interface (GUI), and OpenCV for image processing [65]. Detailed code implementation is available at [5] for interested readers. The application has been tested on Windows and macOS. Although it has not been used in Linux, there is no known incompatibility that would prevent ABECIS from working in Linux as well. Full installation instructions and code are available in the GitHub repository [66]. All the functions described in Figure 5 have been implemented in the ABECIS Graphical User Interface (Figure 10).

The software uses multithreading; therefore, human operators can begin inspecting the results and start rejecting false positives without having to wait for the entire analysis process to finish.
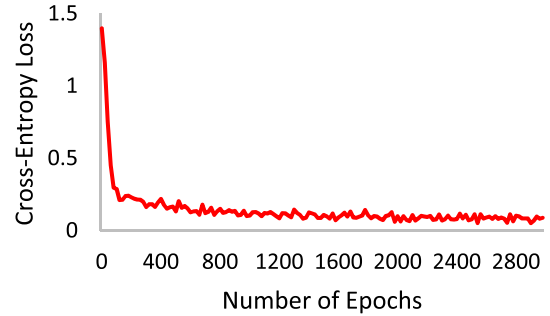
(a)
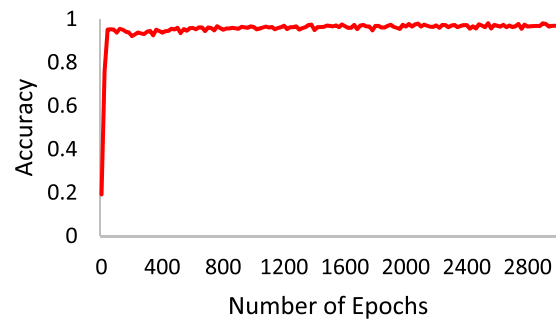
(b)

FIGURE 9. (a) Cross-entropy loss, and (b) accuracy during the training of the neural network model.

C. CRACK LENGTH ESTIMATION

Once the black and white binary masks of the crack detection results are generated, the length of the cracks can be approximated by converting the shape into a one-dimensional object, removing the width dimension using a morphological thinning operation [67]. This operation is an image processing method to reduce the shape of the object into a 1-pixel wide line, called the topological skeleton of the object. This operation is also handled by the ABECIS Graphical User Interface. The process is described in (1).

$$A \wedge B = A - (As^*B) = A \cap (As^*B)^c \qquad (1)$$

where A is the set containing the image pixels with the value 1 (white), B is the structuring element, s* is a hit-or-miss transformation used in morphological image processing [68], and c is the complement operation.

As shown in Figure 11, the original image is analyzed for cracks, and the cracks are segmented. Then, the segmented cracks are converted to a binary mask, followed by the thinning process of the lines to 1-pixel width. Once the lines have a width of 1 pixel, the total length of the cracks in the image is simply the number of white pixels belonging to each crack. Without a metric reference in the image or information regarding the scale, it is not possible to get a proportional conversion between pixels and real-world measurements. Figure 11 is meant to show the process only. Challenging examples are discussed in Sections VI-B and VI-C.
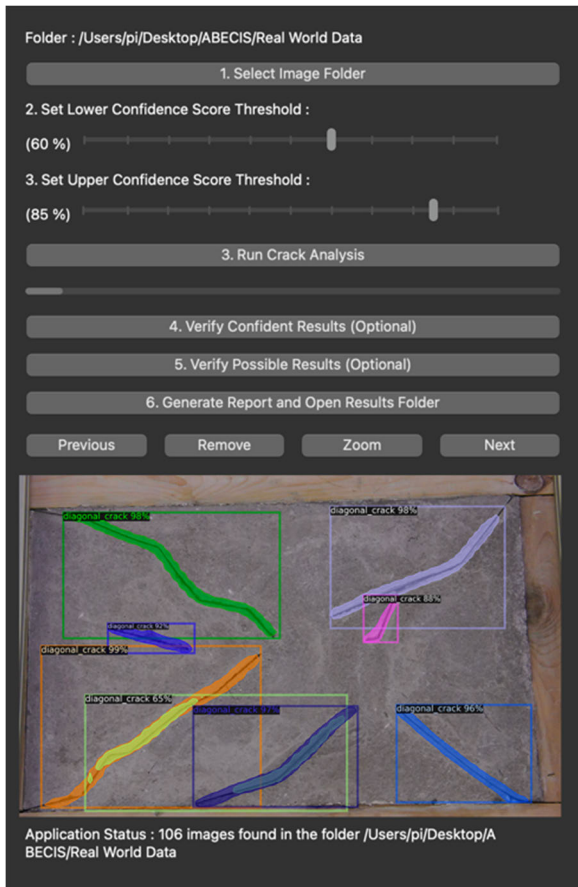
FIGURE 10. Graphical user interface (GUI) for the ABECIS software.
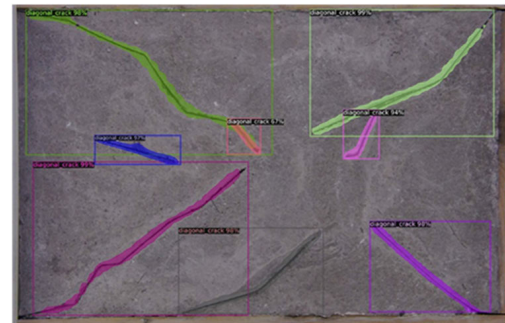
VI. EXPERIMENTATION

Images taken in different environments using different methods were used to evaluate the performance of the ABECIS crack detection software. The results were evaluated (1) visually and (2) using the Intersection over Union (IoU), a metric to evaluate the performance of instance segmentation algorithms based on the extent of overlap of two boxes. The software has been tested in three different environments, (1) a controlled environment in the laboratory with images taken by a commercial drone, (2) a construction site with images taken by a smartphone camera, and (3) an outdoor environment around the New York University Abu Dhabi campus with images taken by a commercial drone (DJI Matrice 300 RTK).

A. LABORATORY EXPERIMENT (DRONE)

A movable mockup wall and a drone were used for the laboratory test (Figure 12). The mockup wall consisted of a concrete surface (approximate dimensions of 63 cm x 76 cm) with a wood frame and 7 intentionally made diagonal cracks. It was mounted on an aluminum frame with wheels for ease of transport. The drone was a DJI Matrice 300 RTK commercial. Altogether, 14 images were taken (the images can be found in [69]) from a distance of 4 meters from the wall and


FIGURE 11. Different components of the crack length estimation process showing (a) original image, (b) instance segmentation of cracks, (c) binary mask resultant from the segmentation process, and (d) morphological thinning operation applied to the binary mask, resulting in lines with a width of 1-pixel.

analyzed using ABECIS. The images taken by the drone have a resolution of 4,056 × 3,040 pixels. The results (i.e., IoU and length) from these images are in Appendix B (Table 5).
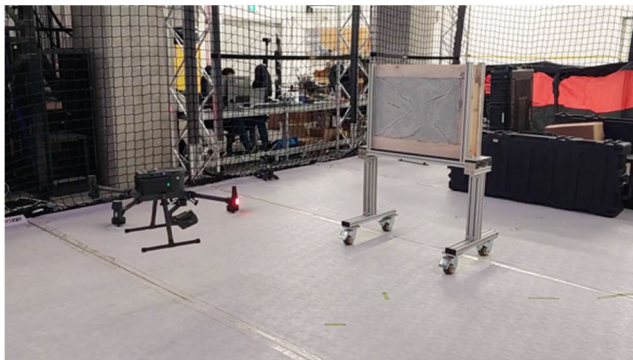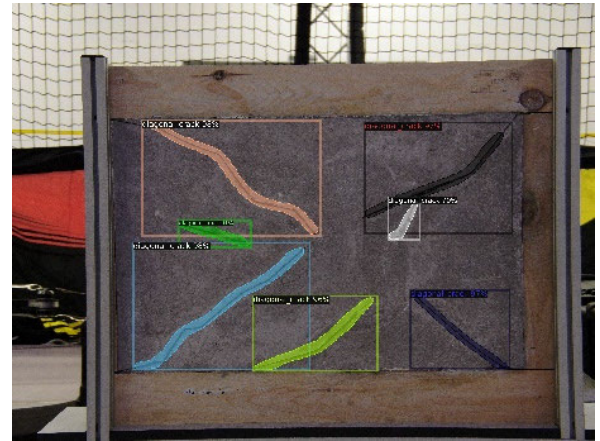
FIGURE 12. General views of the experimental setup showing the mockup wall with concrete cracks and the drone used in (a) standby mode and (b) in operation (i.e., flying and taking images).

From the visual inspection of the different images and the result summarized in the report, it was found that the ABECIS system was able to successfully detect all the cracks in the controlled laboratory environment (Figure 13a), with just a small number of false positives. The false positives occurred when the image contained the surrounding objects in addition to the wall. In such cases, surrounding objects (similar to cracks) were classified as cracks (Figure 13b).

B. ON-SITE EXPERIMENT (SMARTPHONE CAMERA)
For the on-site test, the authors visited an ongoing construction site and took images using an Android smartphone with a 1.9 Megapixel camera with a resolution of 4,000 × 2,252 pixels. Altogether 14 images were taken (the images can be found in [69]), and the reports were generated using ABECIS software. The results can be found in Appendix B (Table 6). Although there are not many cracks in the construction site, the ones present were correctly detected with varying degrees of accuracy (Figure 14).

However, there are also several false positives. These are mainly due to objects that are very similar to cracks (e.g., scratches, markings on walls, pipes, wires). Some examples can be seen in Figure 15. It is worth mentioning that a real ongoing construction site is an extremely difficult scenario for a segmentation process. Nonetheless, all the true positives were detected, and all the resultant false positives can be
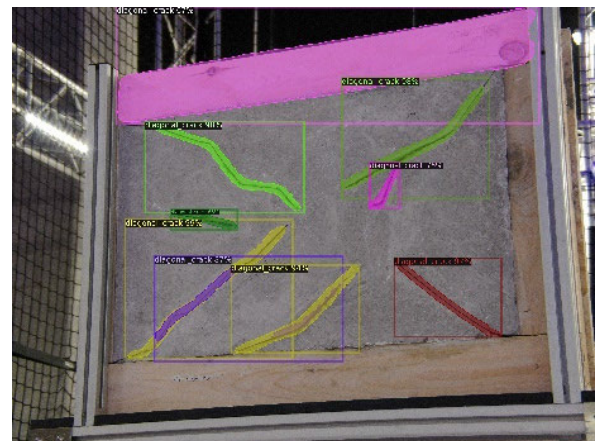
FIGURE 13. General views of the experimental setup showing the mockup wall with concrete cracks and the drone used in (a) standby mode and (b) in operation (i.e., flying and taking images).

manually discarded with an easy manual operation from the ABECIS GUI.

C. OUTDOOR EXPERIMENT (DRONE)
Moreover, for the outdoor test, the authors also took images of the surrounding areas of the NYUAD campus using a DJI Matrice 300 RTK commercial drone. Altogether, 14 images were taken (the images can be found in [69]), and the results can be seen in Appendix B (Table 7). A few cracks found were detected (e.g., Figure 16a). However, the algorithm occasionally missed very thin hairline-type cracks (Figure 16b).

There were a few false positives in the outdoor drone experiment. Many were related to objects resembling cracks, such as the control joints with fillers or areas between the walls or concrete slabs, as in Figure 17a or the gap between two concrete slabs, as in Figure 17b.

VII. RESULTS AND LIMITATIONS
All the results are provided in Appendix B and C. The Repository can be found with detailed information at [69].
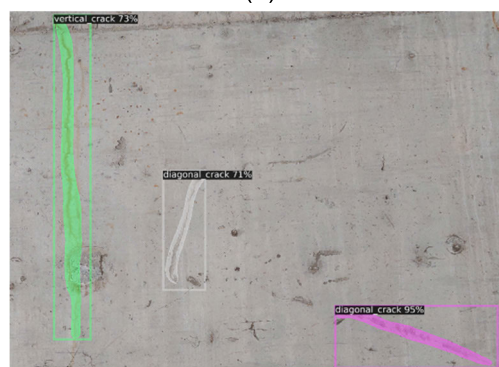
FIGURE 14. Examples of correctly detected (a) vertical and (b) diagonal cracks from the indoor construction site images taken using a smartphone.

FIGURE 15. Construction site false positive examples. (a) Scratches identified as cracks. (b) Stains identified as cracks.

A. INTERSECTION OVER UNION (IOU) EVALUATION

For the evaluation of the performance of the ABECIS system, Intersection Over Union (IoU), a method to quantify the percent overlap between the target mask and the obtained output (Equation 2), was calculated for the 14 images in three categories (i.e., Lab, On-Site and Outdoors). An IoU score of 1 indicates that the model performs perfectly, whereas a score of 0 indicates that the prediction and the ground truth do not overlap at all. The IoU was calculated for a total of 42 (14×3) images in three experiments, and the summary statistics are shown in Table 3.

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \quad (2)$$

To account for the effect of human verification/intervention, two different types of summary statistics for the IoU were calculated for each of the three experiments. One with all the cases and another with True Positives only (indicating the scenario where the inspector reviewing the output would reject those cases). IoU (All) refers to the IoU statistics calculated from the raw crack analysis data directly obtained from ABECIS software, whereas IoU (True Positives) refers to the IoU statistics after the human operator has removed all false positives from ABECIS GUI.

For the IoU for all images, the median of IoU is (1) 0.686 for the indoor crack detection experiment in a

controlled lab environment using a commercial drone, (2) 0.186 for the on-site crack detection at a construction site using a smartphone and (3) 0.968 for the outdoor crack detection using a commercial drone. The interquartile range is the smallest for the Indoor Lab experiment, followed by the Outdoor Campus Experiment and the On-Site Construction Site Experiment.

However, only considering the true positives after human verification, the median of IoU is (1) 0.838 for the indoor crack detection experiment in a controlled lab environment using a commercial drone, (2) 0.967 for the on-site crack detection at a construction site using a smartphone and (3) 0.968 for the outdoor crack detection using a commercial drone. The interquartile range is the smallest for the On-Site Construction Site Experiment, followed by the Indoor Lab experiment and the Outdoor Campus Experiment.

These results indicate that with little or no human verification, ABECIS worked best for the Outdoor Drone inspection, even when compared to the controlled lab experiment. Also, with human verification, the medians of IoUs significantly improve for all categories. These large differences in medians across different categories can be attributed to the condition of the exterior walls not being covered with any obstacles and having a relatively constant background color, making them good candidates for object detection. Moreover, clear weather and wall paint of light color also played a role in
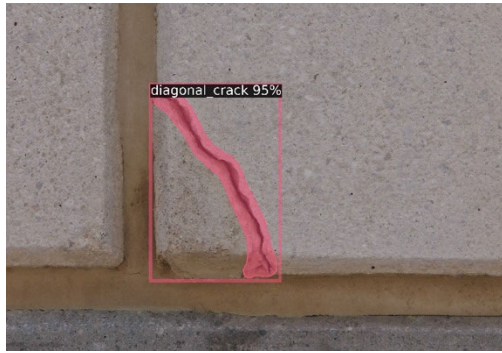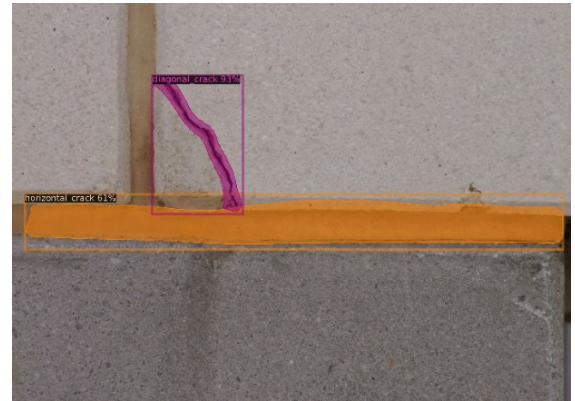
(a)

(b)

FIGURE 16. Examples of correctly detected cracks from outdoor images taken with a drone. (a) Single diagonal crack. (b) Multiple hairline cracks.

(a)

(b)

FIGURE 17. False positive examples from the outdoor experiment. (a) Control joint/filler between walls, identified as a horizontal crack. (b) Area between the parapet wall (vertical element) and ceiling panel (horizontal element) detected as a crack.

the successful detection of the cracks, increasing the contrast between the crack and non-crack pixels. The results from the indoor controlled lab environment with the drone show a median IoU smaller than the outdoor campus experiment. This can be attributed to the number of false positives (e.g., wood planks in the mockup wall and noisy background environment) and unfavorable lighting conditions. Lastly, the construction site experiment has a very large interquartile range and the lowest median. This means that ABECIS had the lowest performance on an active construction site. This is because the construction site is unstructured and contains many temporary elements, such as exposed wires and scaffolding, which were falsely detected as cracks.
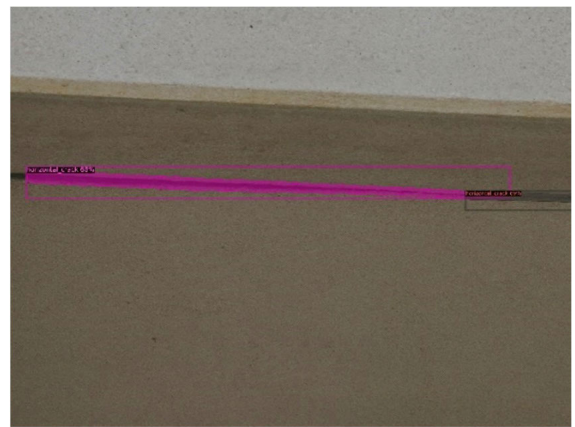
However, considering that all the IoU median values of true positives are greater than 0.8, it can be argued that ABECIS works very well in all indoor, outdoor and construction site environments, using both smartphone and drone cameras when the machine learning algorithm is aided by the human verification process. The IoU score of the trained model can be improved by retraining the model with the most common falsely detected objects (e.g., wires) and removing them from the results. However, this is impractical since many kinds of objects can cause false positives.

B. CRACK LENGTH ESTIMATION

Another of the features provided by ABECIS is the crack length estimation. It is worth mentioning that the length

TABLE 3. Summary statistics for the intersection over union of the three experiments.

Experiment	Lower Adjacent	Lower Quartile	Median	Upper Quartile	Upper Adjacent	Outliers
Indoor – Lab (Drone) - All	0.505	0.642	0.686	0.853	0.999	N/A
Indoor – Lab (Drone) - True Positives	0.642	0.678	0.838	0.938	0.999	N/A
On-Site – Construction Site (Smartphone) - All	0.000	0.000	0.186	0.903	0.968	N/A
On-Site – Construction Site (Smartphone) - True Positives	0.839	0.889	0.967	0.971	0.981	N/A
Outdoor – Campus (Drone) - All	0.370	0.550	0.958	0.968	0.972	N/A
Outdoor – Campus (Drone) - True Positives	0.370	0.550	0.958	0.968	0.972	N/A

measurement given in metric units is only possible if the image has a scale reference to obtain a proportional relationship between the pixel and real-world dimensions or if the image was taken with a device able to measure the distance to the wall (such as the DJI Matrice 300 RTK used in this experiment). The crack length is a useful metric that can assist inspectors in getting a better assessment of the state of the building and if the crack needs urgent repairing or not (also, it would be useful information to generate preliminary repair
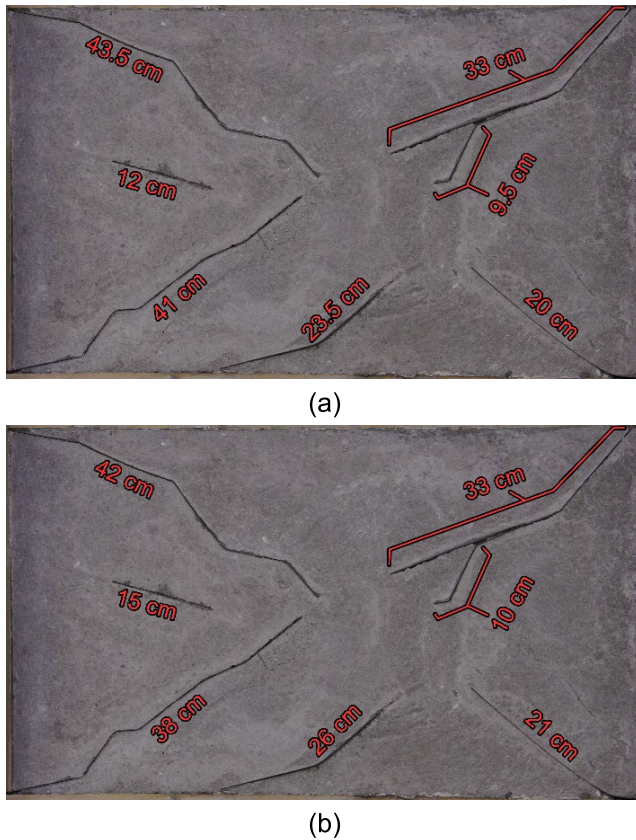
FIGURE 18. (a) Original dimensions of cracks on the mockup wall and (b) corresponding predictions.

TABLE 4. Summary statistics for the Intersection over Union of the three experiments.

Filename	Date/Time Taken	Crack Types	No. of Confident Cracks	Average Confidence Score for Confident Cracks	No. of Possible Cracks	Average Confidence Score for Possible Cracks	Total Crack Length (pixels)	Estimate Total Length (cm)
lab_00003.jpg	2022:03:31 16:21:40	'Diagonal Crack'	3	93.6	2	82.0	185	164
lab_00007.jpg	2022:03:31 16:19:44	'Diagonal Crack'	4	92.0	1	84.4	169	194
lab_00011.jpg	2022:03:31 16:22:06	'Diagonal Crack', 'Horizontal Crack'	2	94.5	5	72.6	150	290
lab_00006.jpg	2022:03:31 16:22:06	'Diagonal Crack'	4	97.4	4	77.4	516	364
lab_00002.jpg	2022:03:31 16:21:10	'Diagonal Crack'	7	95.6	2	63.2	727	202
lab_00012.jpg	2022:03:31 16:20:29	'Diagonal Crack'	6	97.1	2	72.5	609	209
lab_00001.jpg	2022:03:31 16:19:25	'Diagonal Crack'	6	96.5	1	70.4	595	207
lab_00008.jpg	2022:03:31 16:19:58	'Diagonal Crack'	6	96.3	1	83.8	611	186
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

scopes and cost estimates); therefore, it is a valuable asset to be included in the final report generated by ABECIS. In order to test the effectiveness and robustness of the estimations calculated by ABECIS, the controlled lab environment with the mockup wall was used. The ground-truth measurements and the predicted lengths of the cracks present in the mockup wall can be seen in Figure 18.

The length estimations computed for each of the 14 images can be seen in Appendix C (Table 8). The results in the Appendix correspond to the sum of the estimated lengths for all the successfully detected cracks on each image. The computed error is the difference between this value and the sum of all the ground-truth distances. The error median is 8.2%, which is an acceptable margin to provide qualitative assessments of the crack lengths.

C. REPORT GENERATION
After the crack detection is completed and all the measurements have been computed, the results are automatically gathered into a comma-separated values (csv) file format. The generated report contains information such as the image filename, type of cracks detected, date and time extracted from the image metadata, the types of cracks present in the image, and some metrics such as the Number of Confident Cracks and their Average Confidence Score, as well as the number of

Possible Cracks and their Confidence Scores. Moreover, the total crack length in pixels is also provided. From here, the estimated total crack length in centimeters can be calculated provided that a reference object with known length is available in the photo (like in the case of the images taken using the drone. For the images taken using a smartphone, a reference should be added when taking each photo). An excerpt of a report generated is shown in Table 4.

D. LIMITATIONS
One of the limitations of ABECIS is the impact of cluttered and unstructured environments. Images with a cluttered and non-uniform background present a specially challenging scenario when it comes to object detection, making the performance more susceptible to false positives in the detection process. Nonetheless, it is expected that the main use of ABECIS would be in the maintenance stage of the lifecycle of a building, inspecting exterior façades that present a clear and uniform background for the most part. From the results obtained, ABECIS performs best under those conditions.

Another limitation is that estimating the length of the cracks identified (i.e., providing measurements in metric units) is only available if the image contains enough data (or a reference) to establish a proportional relation between pixel and real-world distances. That means that for pictures taken with a smartphone, a metric scale needs to be present in every image (which, in real conditions, might not be practical). However, most modern smartphone models have multiple cameras and even small LiDARs that would allow the software to extract enough metadata from the image to compute the required reference to estimate the length of the detected cracks.

The current implementation has only been tested in concrete-like façades. Although concrete is still the most used material in construction, modern buildings include different

TABLE 5. Summary of different papers identified during the literature review (in reverse chronological order).

A	B	C	D	E	F	G	H
[11]	2021	✓	DSLR*, UAV	Concrete Bridge	Cracks	✓	SA
[25]	2021		UAV	Concrete Bridge	Cracks		SA
[23]	2021	✓	UAV	Concrete	Cracks	✓	SA
[13]	2020	✓	Smartphone	Asphalt	Cracks	✓	SA
[32]	2020		UAV	Metal Bridge	Paint Absence	✓	SA
[15]	2020	✓	UAV	Asphalt	Cracks	✓	SA
[8]	2020	✓	UAV	Concrete Water Tank	Cracks		SA
[7]	2020	✓	UAV	Concrete Bridge	Cracks	✓	SA
[36]	2020	✓	UAV	Mine Slope	Cracks	✓	FA
[26]	2020		UAV	Concrete Bridge	Cracks	✓	SA
[18]	2020	✓	N/A	Damage after earthquake	Cracks, spalling, severe damage	✓	SA
[19]	2020	✓	UAV	Concrete	Cracks	✓	SA
[20]	2020	✓	UAV	Asphalt	Potholes and cracks	✓	SA
[29]	2020		UAV	Concrete Bridges	Biological colonies, efflorescence, cracks and exposed rebar		SA
[30]	2020		UAV	Concrete Bridge	Cracks	✓	SA
[71]	2020	✓	UAV	Façades	De-blurring images for crack detection	✓	SA
[31]	2020		UAV	Concrete Bridges	Cracks		SA
[33]	2019		UAV	Brick Building	Change in appearance	✓	SA
[16]	2019	✓	UAV	Metal Structures	Weld Defects	✓	SA
[72]	2019		UAV	Concrete	Cracks	✓	SA
[73]	2019	✓	Climbing robot	Aircraft surface	Corrosion, stains, and cracks	✓	SA
[34]	2018	✓	UAV	Concrete	Cracks		FA
[35]	2018	✓	UAV	Concrete	Cracks	✓	FA
[74]	2018	✓	N/A	Concrete	Cracks	✓	SA
[27]	2018		UAV	Concrete Bridge	Cracks		SA
[28]	2018		UAV	Wind turbine	Cracks		SA
[75]	2018	✓	UAV	Concrete Lab test	Cracks		FA
[76]	2018		UAV	Steel	Fatigue cracks		M
[9]	2018	✓	UAV	Concrete Bridges	Cracks		SA
[77]	2017		UAV	Concrete Bridge	Cracks	✓	SA
[24]	2017		UAV	Concrete Bridge	Cracks	✓	SA
[12]	2017	✓	UAV	Concrete Bridge	Cracks	✓	SA
[78]	2017		UAV	Concrete Bridge	Cracks	✓	SA
[79]	2017		UAV	Concrete Bridges & Tunnel	N/A	✓	FA
[80]	2017	✓	UAV	Pavement	Cracks	✓	SA

* Digital single-lens reflex camera

TABLE 6. Intersection over union scores of laboratory drone images.

No	Image Name	Crack Type(s)	Image (segmentation mask)	True Positives	False Positives	IoU (All)	IoU (True Positives)
1	lab_00017.jpg	'Diagonal Crack'		8	0	0.999	0.999
2	lab_00001.jpg	'Diagonal Crack'		7	0	0.962	0.962
3	lab_00003.jpg	'Diagonal Crack'		5	0	0.929	0.929
4	lab_00004.jpg	'Diagonal Crack', 'Horizontal Crack'		7	1	0.853	0.892
5	lab_00002.jpg	'Diagonal Crack'		9	1	0.804	0.903
6	lab_00008.jpg	'Diagonal Crack'		7	0	0.773	0.773
7	lab_00012.jpg	'Diagonal Crack'		7	1	0.694	0.783
8	lab_00018.jpg	'Diagonal Crack'		8	0	0.678	0.678
9	lab_00006.jpg	'Diagonal Crack'		7	1	0.665	0.680
10	lab_00022.jpg	'Diagonal Crack'		5	0	0.642	0.642
11	lab_00007.jpg	'Diagonal Crack'		5	0	0.642	0.642
12	lab_00011.jpg	'Diagonal Crack', 'Horizontal Crack'		5	1	0.554	0.654
13	lab_00019.jpg	'Diagonal Crack'		5	1	0.509	0.971
14	lab_00021.jpg	'Diagonal Crack'		4	1	0.505	0.938

types of materials on their exterior façades, which could lead to different results in the segmentation process due to the different nature of the material. Other issues (beyond cracks) also manifest in the exterior of buildings (e.g., water stains/infiltration, sealant deterioration, thermal leaks, etc.), which can accelerate the deterioration of the building and compromise the efficiency of the different building systems. Although ABECIS only focuses on cracks, the algorithm can easily be expanded to identify other issues.

Finally, Detectron2 has been the algorithm selected for ABECIS. However, other instance segmentation algorithms could provide different performances or advantages. It was beyond the scope of this study to do a comparison/evaluation of different algorithms, but this could be considered in future studies.

VIII. CONCLUSION AND OUTLOOK
This study proposes an open-source Automated Building Exterior Crack Inspection Software (ABECIS) for construction and facility managers and building inspectors. The software uses an instance segmentation Artificial Intelligence model developed and trained by the authors using a custom dataset. The system was tested and evaluated in a laboratory (i.e., a controlled environment) and real-world scenarios using different ways to collect images (e.g., cameras on a commercial drone and a smartphone).

ABECIS allows the operation to verify the output provided (i.e., human-in-the-loop). From the raw output of the algorithm without the human verification, the median IoU is greatest for the outdoor crack detection experiments using a drone, followed by the indoor crack detection experiment in a controlled lab environment using a drone and lastly, for the indoor crack detection at a construction site using a smartphone. This indicates that ABECIS performs best for outdoor drone images with minimal human verification. Very often, false positives arise when the environment has too many obstacles and other objects in addition to the wall to be analyzed.

However, these IoU results improve significantly (to over 0.8) when a human operator selectively removes the false positives through the ABECIS user interface. Therefore, combining the ABECIS' predictions and human verification can offer very accurate crack detection for all cases (indoor and outdoor scenarios).

Interested readers can easily install and use our method by visiting the GitHub repository of ABECIS [5].

Ongoing work by the authors includes improving the quality of crack detection of ABECIS software by using a larger training dataset and obstacle detection systems to reduce false

TABLE 7. Intersection over union scores of outdoor construction site images.

No	Image Name	Crack Type(s)	Image (segmentation mask)	True Positives	False Positives	IoU (All)	IoU (True Positives)
1	outdoor_site_00197.jpg	'Vertical Crack'		1	0	0.968	0.968
2	outdoor_site_00200.jpg	'Vertical Crack'		1	0	0.967	0.967
3	outdoor_site_00187.jpg	'Vertical Crack'		1	0	0.946	0.946
4	outdoor_site_00201.jpg	'Horizontal Crack'		1	0	0.903	0.903
5	outdoor_site_00168.jpg	'Diagonal Crack'		2	0	0.848	0.848
6	outdoor_site_00169.jpg	'Diagonal Crack'		3	0	0.839	0.839
7	outdoor_site_00003.jpeg	'Diagonal Crack', 'Horizontal Crack', 'Vertical Crack'		1	3	0.252	0.981
8	outdoor_site_00001.jpeg	'Horizontal Crack', 'Vertical Crack'		2	3	0.120	0.976
9	outdoor_site_00002.jpeg	'Diagonal Crack'		1	3	0.089	0.969
10	outdoor_site_00178.jpg	'Diagonal Crack', 'Vertical Crack'		0	3	0.000	N/A
11	outdoor_site_00136.jpg	'Diagonal Crack', 'Horizontal Crack'		0	2	0.000	N/A
12	outdoor_site_00174.jpg	'Vertical Crack'		0	1	0.000	N/A
13	outdoor_site_00195.jpg	'Diagonal Crack'		0	6	0.000	N/A
14	outdoor_site_00177.jpg	'Diagonal Crack'		0	1	0.000	N/A

TABLE 8. ntersection over union scores of outdoor – campus - drone images.

No	Image Name	Crack Type(s)	Image (segmentation mask)	True Positives	False Positives	IoU (All)	IoU (True Positives)
1	outdoor_drone_00006.jpg	'Diagonal Crack'		1	0	0.972	0.972
2	outdoor_drone_00008.jpg	'Diagonal Crack'		1	0	0.970	0.970
3	outdoor_drone_00016.jpg	'Diagonal Crack'		1	0	0.970	0.970
4	outdoor_drone_00010.jpg	'Diagonal Crack'		1	0	0.968	0.968
5	outdoor_drone_00004.jpg	'Diagonal Crack'		1	0	0.963	0.963
6	outdoor_drone_00018.jpg	'Diagonal Crack'		1	0	0.962	0.962
7	outdoor_drone_00109.jpg	'Diagonal Crack'		2	0	0.962	0.962
8	outdoor_drone_00115.jpg	'Diagonal Crack'		2	0	0.954	0.954
9	outdoor_drone_00065.jpg	'Diagonal Crack'		3	0	0.796	0.796
10	outdoor_drone_00067.jpg	'Diagonal Crack'		3	0	0.720	0.720
11	outdoor_drone_00058.jpg	'Diagonal Crack'		3	0	0.550	0.550
12	outdoor_drone_00062.jpg	'Diagonal Crack'		1	0	0.502	0.502
13	outdoor_drone_00072.jpg	'Diagonal Crack'		2	0	0.430	0.430
14	outdoor_drone_00070.jpg	'Diagonal Crack'		2	0	0.370	0.370

TABLE 9. Length estimation of cracks in laboratory drone images (ground truth 182.5 cm).

No	Image Name	Estimate Total Length (pixels)	Pixels Per Metric Ratio	Estimate Total Length (cm)	% Error
1	lab_00008.jpg	611	30.55	186	1.9
2	lab_00022.jpg	168	8.40	183	0.3
3	lab_00007.jpg	169	8.45	194	6.3
4	lab_00002.jpg	727	36.35	202	10.7
5	lab_00017.jpg	669	33.45	173	5.2
6	lab_00001.jpg	595	29.75	207	13.4
7	lab_00012.jpg	609	30.45	209	14.5
8	lab_00003.jpg	185	9.25	164	10.1
9	lab_00018.jpg	604	30.20	219	20.0
10	lab_00019.jpg	173	8.65	137	24.9
11	lab_00004.jpg	659	32.95	185	1.4
12	lab_00021.jpg	136	6.80	182	0.3
13	lab_00011.jpg	150	7.50	193	5.8
14	lab_00006.jpg	516	25.80	216	18.3

positives. Future work includes expanding the dimensioning of the cracks (e.g., width, deep).

APPENDIX A
SUMMARY OF PAPERS FROM LITERATURE REVIEW
The descriptions of categories in Table 4 are summarized as follows:

- Column A: Reference ID.
- Column B: The year the paper was published.
- Column C: Indicates if the technique proposed uses AI or traditional image processing (such as Canny Edge Detection [70]).
- Column D: The technique used to collect images for processing. UAV indicates a drone with a camera to collect images. Camera indicates a camera other than the one in the UAV.
- Column E: The type of structure (e.g., buildings, bridges) and/or the material the surface is made of (e.g., concrete, steel).
- Column F: The type of defect/item detected (e.g., cracks, absence of paint).
- Column G: Indicates if the paper provides a qualitative evaluation of the method proposed.

- Column H: Degree of autonomy of the proposed method. FA means fully autonomous, and no human intervention is required for the system to function. SA means semi-autonomous, and a significant amount of manual labor (e.g., manually collecting the input data and tweaking thresholds during the processing) is required for the proposed method. M means a fully manual approach with no level of automation involved.

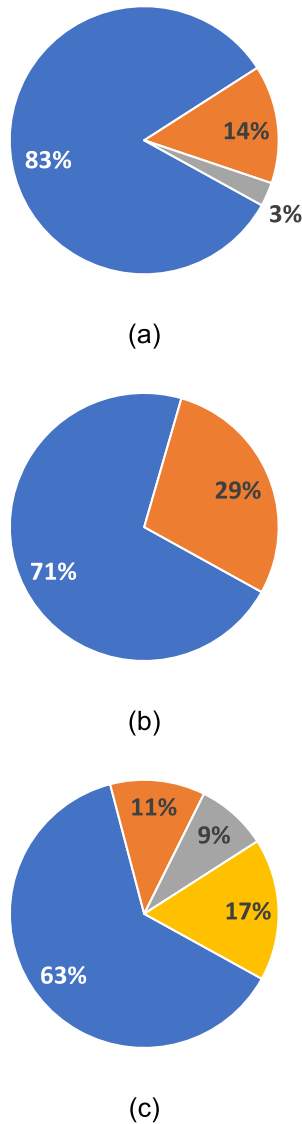The main elements from Table 4 are broken down in Figure 19.

FIGURE 19. Graphical representation of the most representative columns from Table 4. (a) Level of automation – blue, orange, and grey for SA-semi-autonomous, FA-fully autonomous and M-manual, respectively. (b) Percentage of papers that provide a quantitative evaluation – blue: yes, orange: no. (c) Classification according to the type of structure – blue, orange, grey and yellow for concrete, asphalt, metal, and other, respectively.

APPENDIX B
INTERSECTION OVER UNION SCORES
The processed data and images from this appendix can be accessed in [69]. Interested readers may download the high-resolution result images from that reference.

APPENDIX C
LENGTH ESTIMATION OF CRACKS
See Table 9.

REFERENCES

[1] M. P. Ismail, "Selection of suitable NDT methods for building inspection," *IOP Conf. Mater. Sci. Eng.*, vol. 271, Nov. 2017, Art. no. 012085, doi: 10.1088/1757-899X/271/1/012085.

[2] A. M. Paterson, G. R. Dowling, and D. A. Chamberlain, "Building inspection: Can computer vision help?" *Autom. Constr.*, vol. 7, no. 1, pp. 13–20, 1997, doi: https://doi.org/10.1016/S0926-5805(97)00031-9.

[3] L. Parente, E. Falvo, C. Castagnetti, F. Grassi, F. Mancini, P. Rossi, and A. Capra, "Image-based monitoring of cracks: Effectiveness analysis of an open-source machine learning-assisted procedure," *J. Imag.*, vol. 8, no. 2, p. 22, Jan. 2022, doi: 10.3390/jimaging8020022.

[4] B. F. Spencer, V. Hoskere, and Y. Narazaki, "Advances in computer vision-based civil infrastructure inspection and monitoring," *Engineering*, vol. 5, no. 2, pp. 199–222, Apr. 2019, doi: 10.1016/j.eng.2018.11.030.

[5] ABECIS. *SMART Construction Research Group*. Accessed: Apr. 2, 2022. [Online]. Available: https://github.com/SMART-NYUAD/ABECIS

[6] ABECIS Readme. *SMART Construction Research Group*. Accessed: Jun. 19, 2022. [Online]. Available: https://github.com/SMART-NYUAD/ABECIS/blob/main/README.md

[7] Y. Z. Ayele, M. Aliyari, D. Griffiths, and E. L. Droguett, "Automatic crack segmentation for UAV-assisted bridge inspection," *Energies*, vol. 13, no. 23, p. 6250, Nov. 2020, doi: 10.3390/en13236250.

[8] Z. Y. Wu, R. Kalfarisi, F. Kouyoumdjian, and C. Taelman, "Applying deep convolutional neural network with 3D reality mesh model for water tank crack detection and evaluation," *Urban Water J.*, vol. 17, no. 8, pp. 682–695, Sep. 2020, doi: 10.1080/1573062X.2020.1758166.

[9] J.-H. Lee, S.-S. Yoon, H.-J. Jung, and I.-H. Kim, "Diagnosis of crack damage on structures based on image processing techniques and R-CNN using unmanned aerial vehicle (UAV)," in *Proc. SPIE*, Denver, CO, USA, Mar. 2018, p. 35, doi: 10.1117/12.2296691.

[10] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vision (ICCV)*, vol. 17, pp. 2961–2969, Accessed: Jun. 19, 2022. [Online]. Available: https://openaccess.thecvf.com/content_iccv_2017/html/He_Mask_R-CNN_ICCV_2017_paper.html

[11] Z. Yu, Y. Shen, and C. Shen, "A real-time detection approach for bridge cracks based on YOLOv4-FPM," *Autom. Construction*, vol. 122, Feb. 2021, Art. no. 103514, doi: 10.1016/j.autcon.2020.103514.

[12] J. Cen, J. Zhao, X. Xia, and C. Liu, "Application research on convolution neural network for bridge crack detection," in *Proc. 2nd Int. Conf. Comput. Eng., Inf. Sci. Appl. Technol. (ICCIA)*, Wuhan, China, 2017, pp. 1–17, doi: 10.2991/iccia-17.2017.24.

[13] J. Jo and Z. Jadidi, "A high precision crack classification system using multi-layered image processing and deep belief learning," *Struct. Infrastructure Eng.*, vol. 16, no. 2, pp. 297–305, Feb. 2020, doi: 10.1080/15732479.2019.1655068.

[14] Y. Bengio, "Learning deep architectures for AI," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009, doi: 10.1561/2200000006.

[15] L. A. Silva, H. S. San Blas, D. P. García, A. S. Mendes, and G. V. González, "An architectural multi-agent system for a pavement monitoring system with pothole recognition in UAV images," *Sensors*, vol. 20, no. 21, p. 6205, Oct. 2020, doi: 10.3390/s20216205.

[16] C. M. Yeum, J. Choi, and S. J. Dyke, "Automated region-of-interest localization and classification for vision-based visual assessment of civil infrastructure," *Structural Health Monitor.*, vol. 18, no. 3, pp. 675–689, May 2019, doi: 10.1177/1475921718765419.

[17] S. Dorafshan, M. Maguire, N. V. Hoffer, and C. Coopmans, "Challenges in bridge inspection using small unmanned aerial systems: Results and lessons learned," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Miami, FL, USA, Jun. 2017, pp. 1722–1730, doi: 10.1109/ICUAS.2017.7991459.

[18] Y.-J. Cha, W. Choi, and O. Büyüköztürk, "Deep learning-based crack damage detection using convolutional neural networks: Deep learning-based crack damage detection using CNNs," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 32, no. 5, pp. 361–378, May 2017, doi: 10.1111/mice.12263.

[19] S. Bhowmick, S. Nagarajaiah, and A. Veeraraghavan, "Vision and deep learning-based algorithms to detect and quantify cracks on concrete surfaces from UAV videos," *Sensors*, vol. 20, no. 21, p. 6299, Nov. 2020, doi: 10.3390/s20216299.

[20] Y. Pan, X. Chen, Q. Sun, and X. Zhang, "Monitoring asphalt pavement aging and damage conditions from low-altitude UAV imagery based on a CNN approach," *Can. J. Remote Sens.*, vol. 47, no. 3, pp. 432–449, May 2021, doi: 10.1080/07038992.2020.1870217.

[21] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[23] K. Chen, G. Reichard, X. Xu, and A. Akanmu, "Automated crack segmentation in close-range building façade inspection images using deep learning techniques," *J. Building Eng.*, vol. 43, Nov. 2021, Art. no. 102913, doi: 10.1016/j.jobe.2021.102913.

[24] H. Zhang, H. Yu, H. Zhang, and W. Yang, "Accurate extraction of cracks on the underside of concrete bridges," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2017, pp. 2341–2344.

[25] D. Dan and Q. Dan, "Automatic recognition of surface cracks in bridges based on 2D-APES and mobile machine vision," *Measurement*, vol. 168, Jan. 2021, Art. no. 108429, doi: 10.1016/j.measurement.2020.108429.

[26] Y. Liu, X. Nie, J. Fan, and X. Liu, "Image–based crack assessment of bridge piers using unmanned aerial vehicles and three-dimensional scene reconstruction," *Comput.-Aided Civ. Infrastruct. Eng.*, vol. 35, no. 5, pp. 511–529, May 2020, doi: 10.1111/mice.12501.

[27] B. Lei, N. Wang, P. Xu, and G. Song, "New crack detection method for bridge inspection using UAV incorporating image processing," *J. Aerosp. Eng.*, vol. 31, no. 5, Sep. 2018, Art. no. 04018058, doi: 10.1061/(ASCE)AS.1943-5525.0000879.

[28] L. Peng and J. Liu, "Detection and analysis of large-scale WT blade surface cracks based on UAV-taken images," *IET Image Process.*, vol. 12, no. 11, pp. 2059–2064, Nov. 2018, doi: 10.1049/iet-ipr.2018.5542.

[29] D. Ribeiro, R. Santos, A. Shibasaki, P. Montenegro, H. Carvalho, and R. Calçada, "Remote inspection of RC structures using unmanned aerial vehicles and heuristic image processing," *Eng. Failure Anal.*, vol. 117, Nov. 2020, Art. no. 104813, doi: 10.1016/j.engfailanal.2020.104813.

[30] H.-F. Wang, L. Zhai, H. Huang, L.-M. Guan, K.-N. Mu, and G.-P. Wang, "Measurement for cracks at the bottom of bridges based on tethered creeping unmanned aerial vehicle," *Autom. Construct.*, vol. 119, Nov. 2020, Art. no. 103330, doi: 10.1016/j.autcon.2020.103330.

[31] B. J. Perry, Y. Guo, R. Atadero, and J. W. van de Lindt, "Streamlined bridge inspection system utilizing unmanned aerial vehicles (UAVs) and machine learning," *Measurement*, vol. 164, Nov. 2020, Art. no. 108048, doi: 10.1016/j.measurement.2020.108048.

[32] F. Potenza, C. Rinaldi, E. Ottaviano, and V. Gattulli, "A robotics and computer-aided procedure for defect evaluation in bridge inspection," *J. Civil Structural Health Monitor.*, vol. 10, no. 3, pp. 471–484, Jul. 2020, doi: 10.1007/s13349-020-00395-3.

[33] A. Buatik, "3D model-based image registration for change detection in historical structures via unmanned aerial vehicle," *Int. J. Geomate*, vol. 16, no. 58, pp. 1–4, Jun. 2019, doi: 10.21660/2019.58.8218.

[34] J. Jo, Z. Jadidi, and B. Stantic, "A drone-based building inspection system using software-agents," in *Intelligent Distributed Computing XI*, vol. 737, M. Ivanoviá, C. Bădić, J. Dix, Z. Jovanović, M. Malgeri, and M. Savić, Eds. Cham, Switzerland: Springer, 2018, pp. 115–121, doi: 10.1007/978-3-319-66379-1_11.

[35] D. Kang and Y.-J. Cha, "Autonomous UAVs for structural health monitoring using deep learning and an ultrasonic beacon system with geo-tagging: Autonomous UAVs for SHM," *Computer-Aided Civil Infrastructure Eng.*, vol. 33, no. 10, pp. 885–902, Oct. 2018, doi: 10.1111/mice.12375.

[36] Q. Li, G. Min, P. Chen, Y. Liu, S. Tian, D. Zhang, and W. Zhang, "Computer vision-based techniques and path planning strategy in a slope monitoring system using unmanned aerial vehicle," *Int. J. Adv. Robotic Syst.*, vol. 17, no. 2, Mar. 2020, Art. no. 172988142090430, doi: 10.1177/1729881420904303.

[37] Canon Global. *Detecting Cracks With AI Technology*. Accessed: Apr. 2, 2022. [Online]. Available: https://global.canon/en/technology/crack2019.html

[38] Fyangneil, *Pavement Crack Detection: Dataset and Model*. Accessed: Apr. 2, 2022. [Online]. Available: https://github.com/fyangneil/pavement-crack-detection

[39] A. Flôr, *Arthurflor23/Surface-Crack-Detection*. Accessed: Apr. 2, 2022. [Online]. Available: https://github.com/arthurflor23/surface-crack-detection

[40] Soni. *Concrete-Crack-Detection*. Accessed: Apr. 2, 2022. [Online]. Available: https://github.com/vivek6449/Concrete-Crack-Detection

[41] D. Dais. *Crack Detection for Masonry Surfaces*. Accessed: Apr. 2, 2022. [Online]. Available: https://github.com/dimitrisdais/crack_detection_CNN_masonry

[42] Qinnzou. *DeepCrack: Learning Hierarchical Convolutional Features for Crack Detection*. Accessed: Apr. 2, 2022. [Online]. Available: https://github.com/qinnzou/DeepCrack

[43] *Dynamic—Infrastructure*. Accessed: Jun. 20, 2022. [Online]. Available: https://diglobal.tech/

[44] *T2D2 | Thornton Tomasetti*. Accessed: Jun. 20, 2022. [Online]. Available: https://www.thorntontomasetti.com/capability/t2d2

[45] A. M. Hafiz and G. M. Bhat, "A survey on instance segmentation: State of the art," *Int. J. Multimedia Inf. Retr.*, vol. 9, no. 3, pp. 171–189, Sep. 2020, doi: 10.1007/s13735-020-00195-x.

[46] Sliu. *Path Aggregation Network for Instance Segmentation*. Accessed: Jun. 15, 2022. [Online]. Available: https://github.com/ShuLiu1993/PANet

[47] D. Bolya. *You Only Look At Coefficients*. Accessed: Jun. 15, 2022. [Online]. Available: https://github.com/dbolya/yolact

[48] *Facebookresearch/MaskrCNN-Benchmark: Fast, Modular Reference Implementation of Instance Segmentation and Object Detection Algorithms in PyTorch*. Accessed: Jun. 20, 2022. [Online]. Available: https://github.com/facebookresearch/maskrcnn-benchmark

[49] *Facebookresearch/Detectron2: Detectron2 is a Platform for Object Detection, Segmentation and Other Visual Recognition Tasks*. Accessed: Apr. 2, 2022. [Online]. Available: https://github.com/facebookresearch/detectron2

[50] S. A. Prieto, N. Giakoumidis, and B. García de Soto, "AutoCIS: An automated construction inspection system for quality inspection of buildings," in *Proc. 38th Int. Symp. Autom. Robot. Construction (ISARC)*, Nov. 2021, pp. 1–10, doi: 10.22260/ISARC2021/0090.

[51] DJI. *Matrice 300 RTK—Built Tough. Works Smart*. Accessed: Apr. 2, 2022. [Online]. Available: https://www.dji.com/ae/photo

[52] *ACI Concrete Terminology*, Standard ACI CT-13, 2013, p. 78.

[53] Ç. F. Özgenel and A. G. Sorguç, "Performance comparison of pretrained convolutional neural networks on crack detection in buildings," in *Proc. 35th Int. Symp. Automat. Robotics Construct. (ISARC)*, Taipei, Taiwan, Jul. 2018, pp. 693–700, doi: 10.22260/ISARC2018/0094.

[54] L. Pauly, H. Peel, S. Luo, D. Hogg, and R. Fuentes, "Deeper networks for pavement crack detection," in *Proc. ISARC*, Taipei, Taiwan, Jul. 2017, pp. 479–485, doi: 10.22260/ISARC2017/0066.

[55] Ayoolaolafenwa. *PixelLib*. Accessed: Apr. 2, 2022. [Online]. Available: https://github.com/ayoolaolafenwa/PixelLib

[56] *Faster R-CNN and Mask R-CNN in PyTorch 1.0. Meta Research*. Accessed: Jun. 14, 2022. [Online]. Available: https://github.com/facebookresearch/maskrcnn-benchmark

[57] K. Wada. (2022). *Labelme: Image Polygonal Annotation With Python*. [Online]. Available: http://dx.doi.org/10.5281/zenodo.5711226

[58] *LabelMe Annotation Tool Source Code. MIT CSAIL Computer Vision*. Accessed: Jun. 20, 2022. [Online]. Available: https://github.com/CSAILVision/LabelMeAnnotationTool

[59] P. Ko, B. García de Soto, and S. A. Prieto. (2022). *ABECIS-Crack-Segmentation-Dataset*. [Online]. Available: http://dx.doi.org/10.7910/DVN/QYGDMM

[60] P. Ko, S. A. Prieto, and B. García de Soto, "ABECIS: An automated building exterior crack inspection system using UAVs, open-source deep learning and photogrammetry," in *Proc. 38th Int. Symp. Autom. Robot. Construction (ISARC)*, Nov. 2021, pp. 1–11, doi: 10.22260/ISARC2021/0086.

[61] Detectron2. *Meta Research*. Accessed: Apr. 1, 2022. [Online]. Available: https://github.com/facebookresearch/detectron2

[62] *Loss Functions—ML Glossary Documentation*. Accessed: Apr. 2, 2022. [Online]. Available: https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html

[63] Google Developers. *Classification: Accuracy | Machine Learning Crash Course*. Accessed: Apr. 2, 2022. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/accuracy

[64] *PyQt6 · PyPI*. Accessed: Apr. 2, 2022. [Online]. Available: https://pypi.org/project/PyQt6/

[65] *OpenCV-Python · PyPI*. Accessed: Apr. 2, 2022. [Online]. Available: https://pypi.org/project/opencv-python/

[66] *ABECIS/README.Md at Main SMART-NYUAD/ABECIS*. Accessed: Jun. 20, 2022. [Online]. Available: https://github.com/SMART-NYUAD/ABECIS/blob/main/README.md

[67] M. R. Jahanshahi, S. F. Masri, C. W. Padgett, and G. S. Sukhatme, "An innovative methodology for detection and quantification of cracks through incorporation of depth perception," *Mach. Vis. Appl.*, vol. 24, no. 2, pp. 227–241, Feb. 2013, doi: 10.1007/s00138-011-0394-0.

[68] F. Y. Shih, *Image Processing and Mathematical Morphology: Fundamentals and Applications*, 1st ed. Boca Raton, FL, USA: CRC Press, 2011, doi: 10.1201/9781420089448.

[69] P. Ko. *ABECIS-Results*. Accessed: Jun. 14, 2022. [Online]. Available: https://github.com/SMART-NYUAD/ABECIS-Results

[70] *OpenCV: Canny Edge Detection*. Accessed: Apr. 2, 2022. [Online]. Available: https://docs.opencv.org/4.x/da/d22/tutorial_py_canny.html

[71] H. Yu, W. Yang, H. Zhang, and W. He, "A UAV-based crack inspection system for concrete bridge monitoring," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2017, pp. 3305–3308.

[72] J. Y. Rau, K. W. Hsiao, J. P. Jhan, S. H. Wang, W. C. Fang, and J. L. Wang, "Bridge crack detection using multi-rotary UAV and object-base image analysis," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 42, pp. 311–318, Aug. 2017.

[73] S. Dorafshan, R. J. Thomas, and M. Maguire, "Comparison of deep convolutional neural networks and edge detectors for image-based crack detection in concrete," *Construction Building Mater.*, vol. 186, pp. 1031–1045, Oct. 2018, doi: 10.1016/j.conbuildmat.2018.08.011.

[74] D. Han, "Crack detection of UAV concrete surface images," in *Proc. Appl. Mach. Learn.*, San Diego, CO, USA, Sep. 2019, p. 37, doi: 10.1117/12.2525174.

[75] V. Barrile, G. Bilotta, and A. Nunnari, "UAV and computer vision, detection of infrastructure losses and 3D modeling," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 44, pp. 135–139, Nov. 2017.

[76] D. Kang and Y.-J. Cha, "Damage detection with an autonomous UAV using deep learning," in *Proc. SPIE*, vol. 10598, pp. 7–14, Mar. 2018.

[77] S. Dorafshan, R. J. Thomas, and M. Maguire, "Fatigue crack detection using unmanned aerial systems in fracture critical inspection of steel bridges," *J. Bridge Eng.*, vol. 23, no. 10, Oct. 2018, Art. no. 04018078, doi: 10.1061/(ASCE)BE.1943-5592.0001291.

[78] A. B. Ersoz, O. Pekcan, and T. Teke, "Crack identification for rigid pavements using unmanned aerial vehicles," *IOP Conf. Mater. Sci. Eng.*, vol. 236, no. 1, 2017, Art. no. 012101.

[79] B. Ramalingam, V.-H. Manuel, M. R. Elara, A. Vengadesh, A. K. Lakshmanan, M. Ilyas, and T. J. Y. James, "Visual inspection of the aircraft surface using a teleoperated reconfigurable climbing robot and enhanced deep learning technique," *Int. J. Aerosp. Eng.*, vol. 2019, pp. 1–14, Sep. 2019, doi: 10.1155/2019/5137139.

[80] Y. Liu, J. K. W. Yeoh, and D. K. H. Chua, "Deep learning–based enhancement of motion blurred UAV concrete crack images," *J. Comput. Civ. Eng.*, vol. 34, no. 5, Sep. 2020, Art. no. 04020028, doi: 10.1061/(ASCE)CP.1943-5487.0000907.

PI KO is currently pursuing the bachelor's degree in computer engineering with New York University Abu Dhabi (NYUAD). In addition to his studies, he is a Research Assistant with the S.M.A.R.T. Construction Research Group, NYUAD. His research interests include drones, automation, and artificial intelligence.

SAMUEL A. PRIETO (Member, IEEE) received the B.Sc. degree in industrial engineering with a concentration in electrical, electronics, and automation engineering and the Ph.D. degree from Castilla La Mancha University (UCLM), Spain, in 2020. He was a Research Assistant at UCLM University, from 2015 to 2022. He was a Visiting Researcher at the School of Energy, Geoscience, Infrastructure and Society, Heriot-Watt University, U.K. He has been with the S.M.A.R.T. Construction Research Group, New York University Abu Dhabi (NYUAD), since 2020, conducting research in the areas of automation in construction, especially in robotics, 3D scanning, and BIM. He is currently a Postdoctoral Associate at NYUAD.

BORJA GARCÍA DE SOTO (Member, IEEE) received the B.Sc. degree (cum laude) in civil engineering from Florida International University (FIU), the M.Sc. degree in civil engineering with a concentration in engineering and project management from the University of California at Berkeley, the M.Sc. degree in civil engineering with a concentration in structural design from FIU, and the Ph.D. degree from ETH Zurich, Switzerland.

He is an Assistant Professor of civil and urban engineering with New York University Abu Dhabi (NYUAD) and a Global Network Assistant Professor with the Department of Civil and Urban Engineering, Tandon School of Engineering at New York University (NYU). He is the Director of the S.M.A.R.T. Construction Research Group at NYUAD and conducts research in the areas of automation and robotics in construction, cybersecurity in the AEC industry, artificial intelligence, lean construction, and BIM. He has extensive experience in the industry as a Structural Engineer, a Project Manager, and a Construction Consultant. He is a Professional Engineer (PE) with licenses in California and Florida and has international experience in multiple aspects of construction projects.

Dr. García de Soto is a member of the Editorial Board of *Automation in Construction* and the General Secretary of the International Association for Automation and Robotics in Construction (IAARC).

• • •