buildings

MDPI

Article

Hybrid Data Augmentation for Enhanced Crack Detection in Building Construction

Seung-Mo Choi [1], Hee-Sung Cha [1,*] and Shaohua Jiang [2]

[1] Department of Smart Convergence Architecture, College of Engineering, Ajou University, Suwon 16499, Republic of Korea; mo126713@ajou.ac.kr
[2] Department of Construction Management, Dalian University of Technology, Dalian 116024, China; shjiang@dlut.edu.cn
* Correspondence: hscha@ajou.ac.kr

Abstract: Quality management in construction projects necessitates early defect detection, traditionally conducted manually by supervisors, resulting in inefficiencies and human errors. Addressing this challenge, research has delved into automating defect detection using computer vision technology, yet progress has been impeded by data limitations. Numerous studies have explored generating virtual images to tackle this issue. However, these endeavors have fallen short in providing image data adaptable to detecting defects amidst evolving on-site construction conditions. This study aims to surmount this obstacle by constructing a hybrid dataset that amalgamates virtual image data with real-world data, thereby enhancing the accuracy of deep learning models. Virtual images and mask images for the model are concurrently generated through a 3D virtual environment and automatic rendering algorithm. Virtual image data are built by employing a developed annotation system to automatically annotate through mask images. This method improved efficiency by automating the process from virtual image creation to annotation. Furthermore, this research has employed a hierarchical classification system in generating virtual image datasets to reflect the different types of defects that can occur. Experimental findings demonstrate that the hybrid datasets enhanced the F1-Score by 4.4%, from 0.4154 to 0.4329, compared to virtual images alone, and by 10%, from 0.4499 to 0.4990, compared to sole reliance on real image augmentation, underscoring its superiority. This investigation contributes to unmanned, automated quality inspection aligning with smart construction management, potentially bolstering productivity in the construction industry.

Keywords: virtual image; data augmentation; defect detection; 3D virtual model

Citation: Choi, S.-M.; Cha, H.-S.; Jiang, S. Hybrid Data Augmentation for Enhanced Crack Detection in Building Construction. *Buildings* **2024**, *14*, 1929. https://doi.org/10.3390/buildings14071929

Academic Editors: Yasser Mohamed and Pramen P. Shrestha

Received: 11 May 2024
Revised: 13 June 2024
Accepted: 21 June 2024
Published: 25 June 2024

1. Introduction

1.1. Backgrounds

With the onset of the Fourth Industrial Revolution, there has been a notable boost in productivity across the manufacturing sector [1]. The Fourth Industrial Revolution has markedly enhanced productivity across the manufacturing industry through the implementation of robotics and automation. Nevertheless, the construction industry has been relatively slow to adopt these technologies, resulting in lower productivity gains than other manufacturing industries. The construction industry significantly lags behind in terms of productivity growth compared to other manufacturing sectors [2,3]. Over the last twenty years, the annual growth rate of labor productivity in construction has only reached 1% globally, a stark contrast to the average of 3.6% in manufacturing [4]. Consequently, it becomes imperative to adopt measures aimed at enhancing productivity through the integration of robotics and automation [5]. Yet, the construction industry remains considerably less digitized compared to other sectors, positioning it as one of the least digitalized industries [6,7]. Moreover, the construction sector grapples with challenges related to declining productivity stemming from a dwindling labor and technical workforce, compounded by the aging of the labor force. For instance, in South Korea, the proportion

of construction workers under 30 years of age plummeted from 70.5% in 2012 to 20% in 2022 [8]. The reduction in the younger labor force will result in a shortage of workers to compensate for the impending loss of skilled workers due to the retirement of older, more experienced workers.

The shortage of skilled labor is one of the most significant concerns for the construction industry [9]. The decline in the number of skilled laborers is not only a matter of occupational safety but also affects the quality of construction projects. In fact, human-related factors such as a shortage of skilled workers and supervisors contribute to defects in construction [10]. Quality control efforts demand significant time and resources, with supervisors spending an average of 4 h on site and an additional 3 h writing reports for every hour spent on site [11]. Conventional quality inspections are both labor-intensive and ineffective [12–14]. For instance, in the Nanjing subway construction project, more than 40 inspectors spend roughly 3 h daily in subway tunnels searching for defects [15]. Consequently, there is a growing imperative to enhance productivity by minimizing time and effort, prompting recent research to focus on automating quality management processes to streamline these demands [16].

Research related to quality management automation has progressed in various directions. Kim [17] suggested an approach using Lead Zirconate Titanate (PZT)-based electromechanical impedance (EMI) technology for the automatic detection of concrete cracks, identifying the optimal frequency range that strongly reflects structural conditions, thus contributing to the accuracy of crack detection using EMI technology. Liu [18] used point cloud technology to estimate the location of wall cracks through the 3D reconstruction of 2D images, projecting the 2D image cracks onto the 3D concrete surface with cracks to extract accurate 3D coordinates of the crack edges. Image-based analysis technology is receiving much attention in surface quality management research [19].

Image-based analysis technology is considered a key tool for enhancing the construction industry [20,21]. This technology, applicable across various domains such as progress monitoring, safety management, quality inspection, and resource management, holds significant potential [22]. Object detection technology, a type of deep learning technology, automatically identifies and recognizes target objects from images captured by cameras. Convolutional Neural Network (CNN) is the most widely used deep learning method in this realm [23]. Wang developed a framework integrating a 3D scene reconstruction mechanism and a transfer learning model to automatically pinpoint defects on building surfaces within reconstructed 3D scenes [24]. Similarly, Perez utilized CNN techniques to detect building defects [25]. Automating the detection and location identification process offers the benefit of reducing the need for manual labor and mitigating errors stemming from subjective human judgment in defect assessment through objective evaluation [26–29]. Despite these advantages, the technology is constrained by its reliance on data inherent in deep learning.

Image-based analysis utilizing deep learning or computer vision enables computers to comprehend digital images or videos and make judgments similar to human vision [30]. Computer vision uses supervised learning in training models, requiring a substantial amount of labeled image or video data. This implies that a vast amount of data is essential to enhance model accuracy. However, data acquisition poses a common challenge in many studies [16,31,32]. Collecting data for training is time-consuming and labor-intensive. Moreover, obtaining real-world data from the industry sector is often difficult [33]. To tackle this issue, various augmentation methods have been developed to expand limited data. For image data, typical methods include using Python libraries, with several available libraries [34]. A commonly used library, such as imaging, can modify existing data through rotations, resizing, and noise addition. Nonetheless, the diversity of data obtained through these methods is relatively limited [35]. To overcome this limitation, Generative Adversarial Network (GAN) has been developed to extract features from existing images and generate entirely new ones [36]. However, training GAN models can be intricate and is typically regarded as advanced technology, particularly in industrial settings [37].

This study aims to enhance the conventional method of manually transcribing images of actual defects into a trainable format. To this end, we propose a methodology for generating virtual defect images based on 3D modeling. This approach seeks to reduce the time required for data acquisition, improve efficiency through the automation of image generation and labeling, and contribute to the expansion of data diversity. The research follows these steps: (1) reviewing prior research on the generation of virtual image data and the automation of defect detection, (2) developing methodologies for model generation and automation of image data creation, and (3) conducting an empirical study on the generated image data.

1.2. Scope and Flow of the Study

1.2.1. Scope of the Study

Quality management is integral to all types of construction projects, yet, as shown in Table 1, over half of the defects identified during post-construction quality inspections are attributed to the finishing process [38]. It is difficult for the actual users or occupants of a building to directly observe the structural defects of the building or the defects of elements such as pipes inside the building. Conversely, the interior finishes themselves are a source of direct discomfort. Such defects are immediately visible to the users and directly affect their quality of life. It is, therefore, of paramount importance to manage defects in finishing works. Furthermore, the majority of finishing work is conducted indoors. This implies that there are numerous alterations to construction in indoor environments, accompanied by a correspondingly dynamic evolution of spatial information. In essence, the inspection of finishing work necessitates a more comprehensive examination than that of framing and structural work. Moreover, the periodic visits of a manager to a given space to conduct inspections are an inefficient use of resources. Therefore, we have chosen to concentrate on the finishing process, given its combination of low efficiency and high occurrence of defects.

Table 1. Ratio and cost of defect by building work [38].

Sub-Work		Number of Defects	Ratio (%)	Unit Cost (Unit: KRW)	Defect Repair Cost (Unit: KRW)
Finishing	Paper Hanging	6839	14.91	8096	55,368,544
	Tile	4536	9.89	18,091	82,060,776
	Floor	4332	9.45	15,002	64,988,664
	Interior Finishing	2607	5.68	121,832	33,453,024
	Painting	2469	5.38	5361	13,236,309
	Cleaning	1999	4.36	3343	6,682,657
	Masonry	1004	2.19	23,605	23,699,420

The finishing process encompasses various types of tasks, including drywall installation, brick walls, ceiling work, stair railings, hardware installation, plastering and waterproofing, tile installation, painting, glass/insulation work, window installation, sheet work, and cabinetry, among others. Among these, pre-occupancy inspections have revealed that wallpaper and tile work are the processes most prone to defects (as shown in Table 1). Additionally, tile work stands out as the most expensive to rectify. This underscores the importance of meticulous management of tile work to reduce project expenses. However, there has been limited research conducted on automatically detecting tile defects that occur during construction.

Manual inspection is susceptible to errors and inconsistencies due to subjective judgment, making it time-consuming and inefficient [39]. To address these challenges, numerous studies have explored defect detection in tiles [40–52]. For instance, Wan [50] enhanced the Feature Pyramid Network (FPN) of the You Only Look Once (YOLOv5) deep learning model and applied the Convolutional Block Attention Module (CBAM) module, achieving significant improvements in F1 score and mAP (mean average precision). Zhu [51] utilized

semantic segmentation to accurately delineate objects with polygonal borders, achieving a high mIoU (Mean Intersection over Union) value. Dong [52] proposed a method for detecting surface defects in mosaic tiles using industrial cameras, achieving a high detection rate and reducing detection time substantially. However, while these studies primarily focus on post-manufactured surface defect detection in tile factories, they are not directly applicable to post-installed inspections in on-site construction. To automatically detect defects in tiles after installation, it is crucial to identify defects in images of walls and/or floors made of tiles; yet, research in this specific area is limited. While Stephen [49] conducted a study to detect cracks in tiled walls, it was confined to walls made of a single type of tile and primarily emphasized a machine learning algorithm.

1.2.2. Research Flowchart

Advanced augmentation techniques involve synthetic image generation, creating entirely new data and thereby surpassing the limits of traditional transformations. This includes using GANs or developing models in virtual environments and rendering them to create images. Having delineated the scope of the study, which focused on tile defects, the direction for research is established to ensure that a well-defined and organized study is conducted. This study is initiated from a literature-based investigation, followed by problem identification, research design, research execution and analysis, and conclusions and suggestions, as shown in Figure 1. The flowchart is a generic sequence, and our study reflects its specific characteristics in a more detailed and elaborate plan.

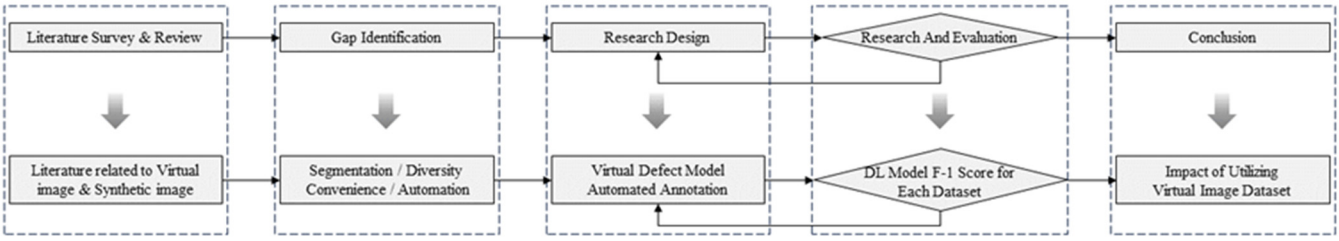


Figure 1. Flowchart of research.

During the preliminary research survey and problem identification phase, we delved into studies focusing on automated defect detection and virtual image data generation to bolster the accuracy of deep learning models. Through this exploration, we pinpointed specific challenges and issues as knowledge gaps. Subsequently, in the research design phase, we formulated the framework for our study. This framework included tasks data collection, virtual data generation, development of an automated annotation system, training deep-learning model, and analysis of the result of training. In the research execution phase, we put the designed framework into action, which involved image collection, setting up the virtual environment, and generating the dataset with the developed system. This dataset served as the basis for training deep learning models and assessing their learning accuracy. Following the model training and validation, we meticulously analyzed the data and interpreted the results to draw meaningful conclusions.

2. Literature Review

This study evaluates the efficacy of data augmentation through virtual data in training deep learning models for defect detection. This literature review focuses on recent advancements in construction defect detection and the application of synthetic data for model training. In particular, it examines how these approaches address the limitations of traditional data collection methods and enhance the accuracy and efficiency of defect detection systems. It includes a comprehensive analysis of various methods employed in defect detection and explores the potential of synthetic data to overcome common challenges in the field.

2.1. Research on Defect Detection

Conventional methods of construction defect detection are time-consuming and labor-intensive, necessitating automation in the construction industry due to a dwindling workforce [53]. Consequently, significant research efforts have been made to automate this process. Lu Y [54] employed the Canny-edge algorithm on images to detect damages, losses, or cracks in exterior masonry finishes. Although effective in localizing suspect areas, this method has limitations in classifying objects accurately as cracks, grout lines, or other defects. This highlights the necessity for more sophisticated classification techniques. Luo [55] and Ichi [56] both utilized thermal cameras to detect structural defects such as separations and delamination, marking a crucial advancement in non-destructive defect detection. Luo proposed a hybrid of spatial-temporal deep learning architectures, while Ichi concentrated on measuring thresholds between defect and non-defect areas, emphasizing the precision of thermal imaging in monitoring structural health. However, since cameras are more affordable and easier to deploy than sensors, numerous recent research projects based on computer vision technology have been conducted [57].

The transition to computer vision-based technologies has prompted numerous studies that employ computer vision techniques. Lee [58] applied the Faster Regions with Convolutional Neural Networks (RCNNs) feature to detect external defects in multi-family housing, such as peeling, cracks, and leaks. The resulting accuracy was 62.7%, which represents a significant improvement over traditional methods. Xu [15] and Dais [59] further enhanced model performance using deep learning. Xu improved mAP scores in tunnel defect detection by employing the Mask-RNNC model, while Dais achieved the highest F1-score of 79.6% using the pre-trained U-net-Mobilenet model on masonry surfaces defect detection. In conclusion, these studies collectively advance the field of construction defect detection by integrating innovative computer vision and deep learning techniques, paving the way for the further exploration of automated, intelligent systems in the construction industry. Nevertheless, the efficacy of these deep learning models is contingent upon the quality and quantity of the data, with a pronounced decline in performance observed in the case of data-scarce defects [16,60–63].

2.2. Research on Training with Synthetic Image Data

2.2.1. Synthetic Image Generated by GAN

Due to challenges in data collection or insufficient data diversity, obtaining adequate data for training deep learning models is often difficult. To address this, an augmentation technique was developed to enhance datasets and improve model accuracy by introducing variability [64]. Traditional augmentation methods involve applying geometric or photometric transformations to original images to generate new image data. A prime example is the use of the ImgAug library [65], which applies various changes such as rotation, shearing, cropping, and noise addition to the original images, creating new data. While beneficial for data diversity and accuracy, these methods may lead to potential overfitting issues [66]. Advanced augmentation techniques involve synthetic image generation, creating entirely new data and thereby surpassing the limits of traditional transformations. This includes using GANs or developing models in virtual environments and rendering them to create images.

GAN technology, recognized as a potent deep learning model for generating new data, is being actively considered across industries that demand large volumes of data [67–69]. Substantial research has been conducted on synthetic data within this framework. For instance, in the medical sector, Qin [70] utilized various GAN models to produce synthetic skin lesion images, examining the discrepancies across images from different models. Similarly, Sandfort [71] employed CycleGAN technology on contrast CT images to create non-contrast counterparts, while Lei [72] adapted the CycleGAN model to transform brain MR images into CT images, facilitating data generation. The agricultural sector also reflects a growing trend of leveraging GAN-based technology for image data augmentation [73]. Concurrently, the urban and construction fields have witnessed

analogous explorations. Dewi [74] incorporated DCGAN with various backbone networks to augment traffic sign image data, demonstrating that integrating synthetic images with original ones could enhance accuracy up to 92%.

Li [75] synthesized high-resolution images of cracked concrete by merging concrete images with hand-painted damage maps using GAN-based networks, showcasing significant authenticity. The study demonstrated that GAN technology has the potential to enhance computer vision-based automation techniques for the detection of construction defects, and it proved that the learning of synthetic images is also valid in the field of construction. Furthermore, Bang [76] detailed a method where target objects from actual site images are cut out, geometrically transformed, and then reintegrated using GAN technology to fill the resultant gaps. Despite GANs' benefits, their limitations are pronounced. GANs necessitate training data for new image generation, with insufficient data potentially leading to mode collapse and diminished data diversity [77].

The paradox lies in the need for ample data to create synthetic data for augmenting scarce object data. Moreover, the GAN image generation method is inapplicable to this study due to its inherent feature-based image creation process. GANs introduce minor diversity to data, generating subtly varied images, and, hence, limiting their use in depicting the dynamic nature of construction sites where continuous change is the norm. The generated images predominantly reflect the processes shown in the training images. Additionally, the auto-labeling of GAN-produced images is challenging, while rendering manual labeling is inefficient and time-consuming. Given these constraints, even though the GAN technology is effective in generating synthetic images, virtual environments are preferred for generating synthetic image data in restricted scenarios like construction sites. Moreover, here, to distinguish between images created based on virtual environments and based on GANs, the term virtual image to refer to images based on virtual environments is used.

2.2.2. Virtual Image Generated by Virtual Environments

Methods for generating synthetic images in virtual environments can be categorized into two types: one that places real background images in a virtual environment and superimposes virtually generated target objects onto these backgrounds, and another that generates both the background and the objects completely virtually. Starting with real image backgrounds, Soltani et al. [78] introduced a method that synthesizes 3D models of target objects onto construction site images, automatically generating images and annotations, thereby improving annotation accuracy, reducing time, and increasing true detection rates. Hwang et al. [79] built a database of 99,800 images in just 42 min using foreground–background cross-oversampling on 3D object-applied real-site background images and web-crawling, achieving an F-1 score of 96.99%.

Kim et al. [80] and Lee et al. [81] placed 3D models of people and construction materials or safety hooks, respectively, in virtual environment backgrounds and used the generated images as data to achieve significant results. Similarly, Barrera-Animas and Delgado [82] created fully 3D construction sites and objects within a virtual environment, and Assadzadeh et al. [83] trained on 3D modeled excavator images to learn poses based on joint locations, validating the effectiveness of images generated in a virtual environment against real image datasets. Numerous studies have also been conducted on building scene understanding, using methods that automatically generate segmentation annotations for object detection targets in 3D-modeled building shapes in virtual environments [84–88]. In addition, Neuhausen et al. [89] created a virtual environment similar to real-world construction sites to train CV deep learning models for tracking worker performance in the field. Research has also been conducted on defect detection, creating magnified images of dented pipes to automatically label data for models used to detect defects in pipes in general industrial settings [90]. The studies reviewed collectively offer unique insights into the advancement of synthetic data augmentation techniques and their potential for use in the construction of defect detection systems.

However, most studies, including those focusing on defects, have been on relatively large objects and are sparse, indicating a lack of active research into generating virtual environment-based data for defect detection, especially for smaller defects such as tile cracks. Also, studies about generating defects cast doubt on the diversity of defects generated, and the simple use of common 3D assets for modeling may not satisfy specific defect requirements, highlighting the need for methods to generate desired defects. In essence, there is a lack of research into the generation of data in virtual environments for the detection of construction-related defects that occur in constantly changing conditions, indicating a lack of methodologies that can be applied to the detection of user-specified defects.

3. Knowledge Gap and Research Objective

Despite extensive research into synthetic image generation, there is a notable scarcity of studies focusing specifically on creating synthetic images tailored for detecting defects that occur during the construction process. In contrast to previous studies, tile defects are notably small, lack a fixed shape, and can manifest in a multitude of shapes. However, there is a paucity of research on the generation of virtual images for them. While GANs represent a potent tool for image synthesis, their application in dynamic environments like construction sites, characterized by high complexity and diversity of defects, is limited. For example, Siu et al.'s [91] work on detecting cracks in underground pipes using enhancing synthetic data generated in a virtual environment with style transitions shows potential in this area; however, the study focused primarily on the static environment of the pipes, which may not be fully representative of the dynamic changes observed on construction sites. Similarly, Qiu et al. [92] demonstrate the benefits of using synthetic data enhanced with YOLOv8-FAM and style transitions for automatic rail fastener defect detection in a relatively static situation. Nevertheless, this study also underscores the limitations of GANs and style transitions in environments with greater dynamic variation and complexity, such as construction sites. In contrast, for defect detection in dynamic environments, virtual environments alone have the potential to generate valuable data without further processing. However, research solely utilizing virtual environments for defect detection remains underexplored, primarily due to the scarcity of studies. Most existing research tends to focus on understanding indoor scenes or detecting large objects like machinery, people, and materials.

This study proposes a new method to generate virtual defect image data for dynamic environments like construction sites, specifically focusing on tile work. Using a hierarchical classification scheme, various defect types are generated in a virtual environment, aiming to bridge the reality gap through rendering. Additionally, automatic segmentation labeling and annotation are employed to streamline preprocessing. The methodology is applied to compare virtual and real image data performance, with potential applications beyond tile work in construction quality management systems.

4. Methodology

4.1. Research Framework

The research framework is structured into five main stages: (1) real image collection, (2) defect type analysis, (3) virtual 3D model and data generation, (4) real image labeling and construction, and (5) the performance comparison of various datasets. This framework systematically lists the tasks performed at each stage, outlining the direction of the research (Figure 2).

In the first stage, real images were collected through Google crawling, web searches, and field photography. Following this, defect type analysis (stage 2) ensured data diversity and the accuracy required for model training, leading to the virtual model creation (stage 3). During this stage, virtual images were generated by applying defects to real building 3D models (stage 4). These generated data were then used for model training, completed with real image labeling. Lastly, the performance of various datasets created was compared to derive the most effective data combination (stage 5). Unlike previous

studies, this approach enables automatic annotation through Mask image rendering and analyzes the contribution of segmentation-applied virtual data in defect detection. Detailed methodologies and contents for each stage are explored in the following sections.

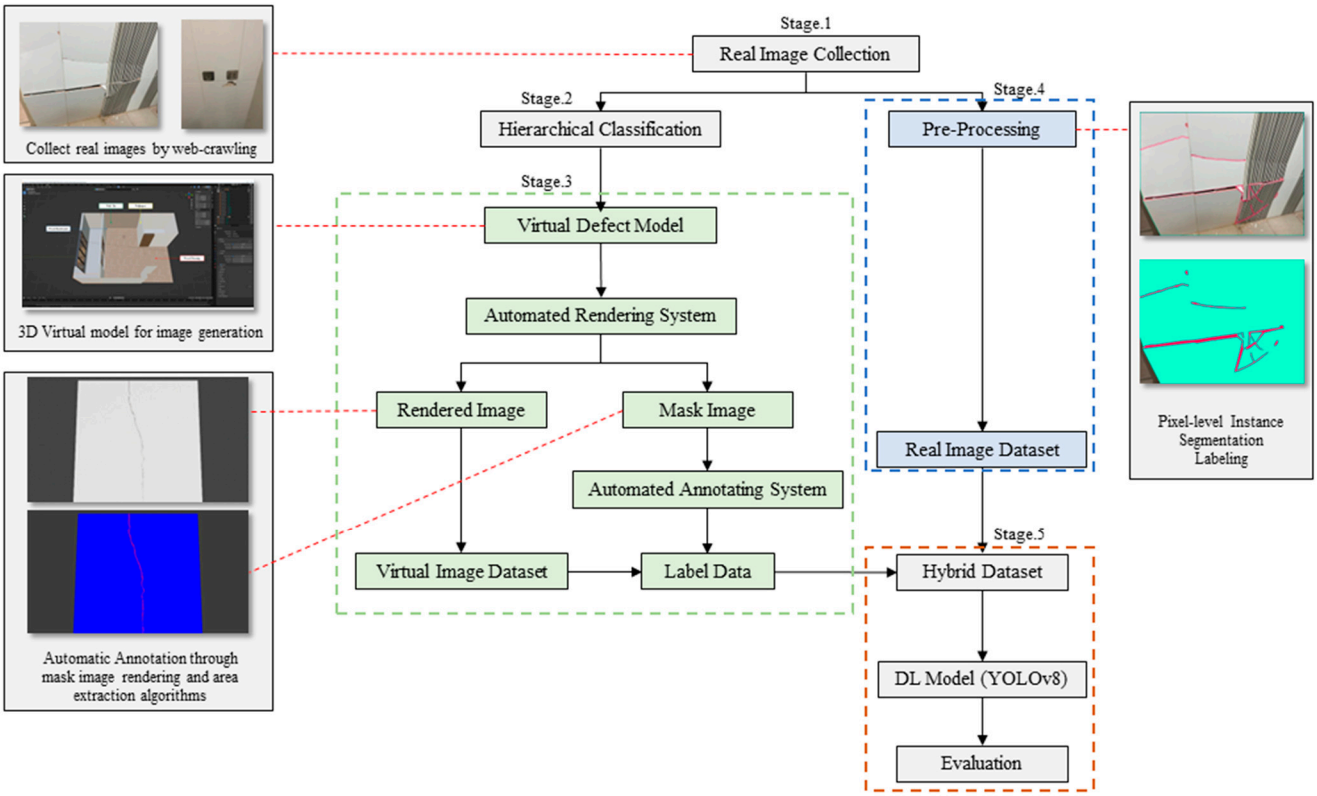


Figure 2. Research framework.

4.2. Real Image Collection

This research seeks to evaluate how replacing real images with virtual ones affects the accuracy of deep learning models trained for defect detection. It involves comparing model performance using both real and virtual images and examining how accuracy changes with different ratios of real and virtual images. Therefore, real image collection was necessary to provide a baseline for comparison and to understand the impact of incorporating real data into the training process.

In this research, image crawling techniques were employed to obtain real images. Image crawling involves extracting all images uploaded on a web page, with Google Images being the target source in this study. Relevant search words, such as "Tile construction (in Korean)", "Tile crack (in Korean)", and "Tile defect (in Korean)", were used to retrieve images suitable for the research. Following extraction, irrelevant or inappropriate images were filtered out. The image crawling process was conducted using Python in VScode, employing the following algorithm (Algorithm 1). The algorithm starts with initiating a Chrome WebDriver and then directs the browser to Google Images and inputs search terms specific to our study needs (e.g., 'Tile cracked'). It then navigates through the search results, downloading each image to our local server. A flowchart accompanying Algorithm 1 visually represents these steps, helping to clarify the sequence and integration of tasks. This approach not only streamlines the acquisition of high-quality images but also aligns with our objective to create a comprehensive dataset for defect detection.

Algorithm 1 Web Image Crawling

```
Install and initiate a Chrome WebDriver
Set the URL to the Google Images search page
Navigate the driver to the specified URL
Wait implicitly for elements to load
Find the search input element by its CSS selector
Enter 'Tile cracked' into the search input
Send a RETURN key to initiate the search
Wait for the search results to load
Scroll the webpage down 60 times to load more images
Attempt to click the 'Show More Results' button if present
Scroll again 60 times after clicking the button if applicable
Collect links to all loaded images
Filter out and store non-empty source URLs in a list
Print the number of found images
Download each image by its URL and save it to the local drive under the C drive
Print a message upon completion of downloads
```

Using this image crawling technique, a total of 340 real tile defect images were collected. The collection comprises 230 images with tile defects and 110 images of tiles without defects. Randomly selected sets of 50 images each were used for validation and testing in the learning process.

4.3. Defect Type Analysis Phase

Defects in construction exhibit diverse characteristics, including variations in location, size, and simultaneous occurrences. For an automatic defect detection model to be effective, it must accurately identify these diverse types. However, simply having a large quantity of image data may not suffice, as a lack of diversity can hinder the model's accuracy. Hence, ensuring data diversity is crucial for enhancing the model's performance in defect detection. To achieve this, we conducted a detailed analysis of defect types using images acquired during the real image collection phase. Hierarchical classification techniques were employed to categorize the defects effectively.

Hierarchical classification arranges data into multiple levels of hierarchy, aiding in organizing information and gaining insights from the data. It typically begins with broad categories and progressively becomes more specific. When used for defect type classification, it can include categories such as work type, defect type, form of defect, location, and number of defects, as depicted in Figure 3. When applied to tile work, the focus of this research, the outcomes are presented in Table 2. By utilizing this classification system, a variety of defect types can be generated in virtual 3D models, thereby ensuring data diversity.

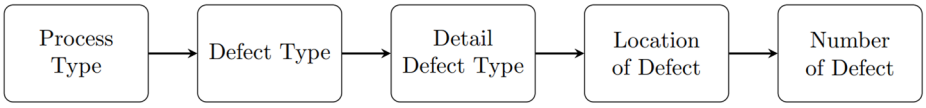


Figure 3. Hierarchical class classification flow.

Table 2. Example of tile hierarchical class classification.

Work Type	Defect Type	Detail Defect Type	Location	Number of Defects
Tile	Crack	Typical Crack	Center/Side/Corner/Whole	Single/Multiple
		Micro Crack	Center/Side/Corner/Whole	Single/Multiple
	Fail	Typical Fail	Center/Side/Corner/Whole	Single/Multiple
		Partial Fail	Center/Side/Corner/Whole	Single/Multiple

4.4. Creation of a Virtual Environment

4.4.1. Creation of Virtual 3D Models

Rather than using GAN models, virtual images have been generated using 3D modeling tools and rendering. Initially, blueprints matching the standards, materials, and scale of the real building were chosen. Public data blueprints from Seoul Housing and Urban Corporation (SHUC) were used for this purpose. In modeling, the focus was on the building's interior walls, especially areas using tiles. Blender was utilized as the modeling software, with Blenderkit and Textures serving as the sources for textures and 3D assets.

Blender, an open-source 3D computer graphics software, is used in various fields such as architecture, product design, and game modeling. It offers a wide range of tools for basic modeling, texturing, shading, and rendering. Moreover, it features scripting capabilities necessary for creating an automatic rendering system, allowing for additional work through Python-based commands. The realism of rendered images is vital for their value as image data, and Blender's Cycles physics-based render engine excels in producing high-quality renderings. Hence, Blender was chosen as the modeling tool for this research owing to its versatility and rendering capabilities.

Within Blender, models were crafted using the Modeling feature, while textures resembling the finishes used in each location were sourced from Blenderkit and Texture. The blueprints indicated various finishes, including wood-patterned flooring, wooden skirting, plain white wallpaper, and ceramic art wall tiles. Models were constructed to incorporate these elements. To accommodate changes in lighting conditions resulting from the sun's position, four models were generated and positioned to face east, west, north, and south around a central sun lamp. Additionally, the brightness of the interior lighting was adjusted lower in one of the models to consider variations in brightness and light reflection during rendering. This methodology resulted in the creation of 3D models, as illustrated in Figure 4.

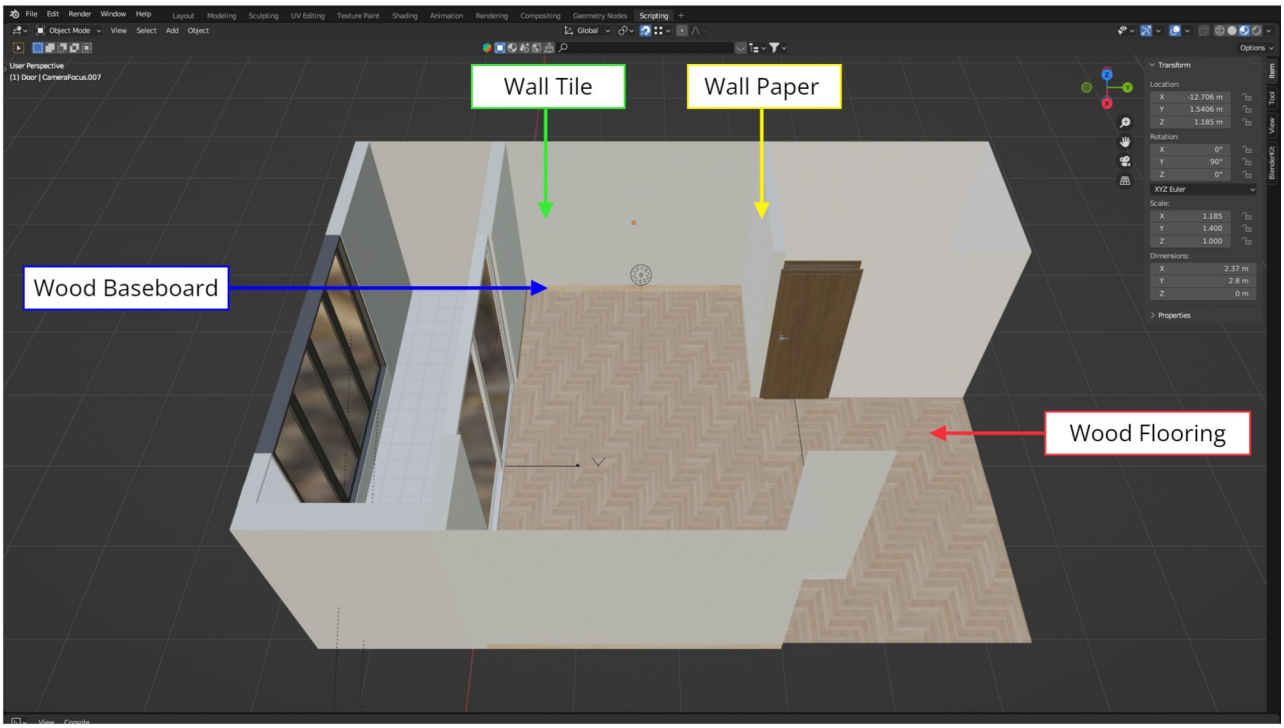


Figure 4. Example of 3D virtual model.

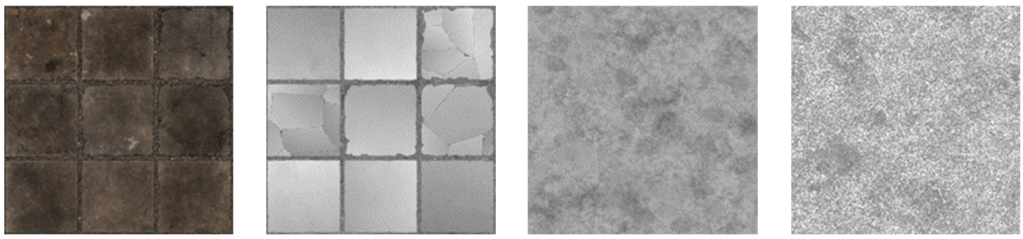
4.4.2. Creation of Virtual Defect Assets

To create virtual defect data, defects need to be incorporated onto the tile surfaces of the previously mentioned 3D models and then rendered. One approach to applying

defects onto tile surfaces involves overlaying 2D defect images onto the tile texture using shading techniques. However, this method is limited to 2D representation and fails to capture changes in shadows or perspectives resulting from variations in lighting or camera angles. Moreover, 3D representation is essential for the object, as images are generated from the multiple viewpoints of a single wall. This necessitates data for Normal Map and Displacement in the texture components, comprising images of the underlying base exposed to defects, images in their pristine state without defects, and area information images outlining the defect regions. An illustrative example of this process is depicted in Figure 5.

Figure 5. Image layering process.

Firstly, the base images that compose the background should consist of a material similar to concrete or plastered surfaces found in construction sites, as seen in Figure 6a. This is because when defects occur in tiles, exposing the base, or when micro-cracks develop, the color recognized in the image is similar to that of concrete. Additionally, the base image is required to contain three-dimensional information about the crack. Therefore, the base image includes Displacement image data for three-dimensional elements, as shown in Figure 6b, where the cracked areas appear indented. This three-dimensional feature allows for the expression of changes under different lighting conditions and shooting angles. The tile texture image is then layered over this base image.



(**a**) Background (**b**) Displacement1 (**c**) NormalMap (**d**) Displacement2

Figure 6. Images for texture.

For this purpose, comprehensive information images, such as Normal Map and Displacement, are included (Figure 6c,d). After selecting the material, the tile texture's shading nodes are imported into the Shading editor of the original base image texture. Then, the nodes of tile texture and base texture are connected. As shown in Figure 7, during the process of layering two image data using a Mix node, the tile texture is placed on the upper node, while the base image data are connected to the lower node. This process layers the tile texture over the base texture. The unique Displacement of both textures is also combined and displayed. The three-dimensional object formed by these layered images is referred to as a "defect asset". When image data containing area information, which differentiates between the exposed base and the tile texture (Figure 8a,b), is added to the Mix

node in Shading, the defect becomes visible in the layered defect asset. Black areas maintain the tile texture, forming a tile with defects. Subsequently, multiple tiles are created and integrated into the 3D model to form a defective 3D model.

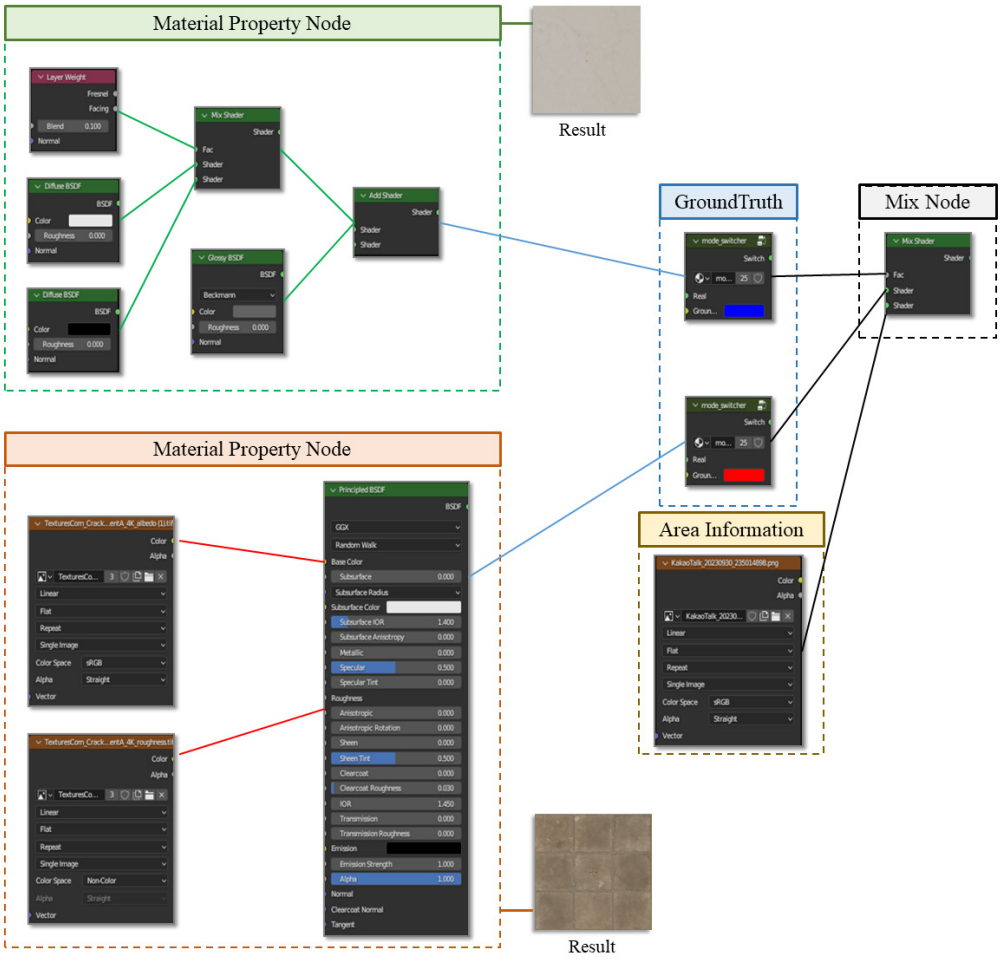
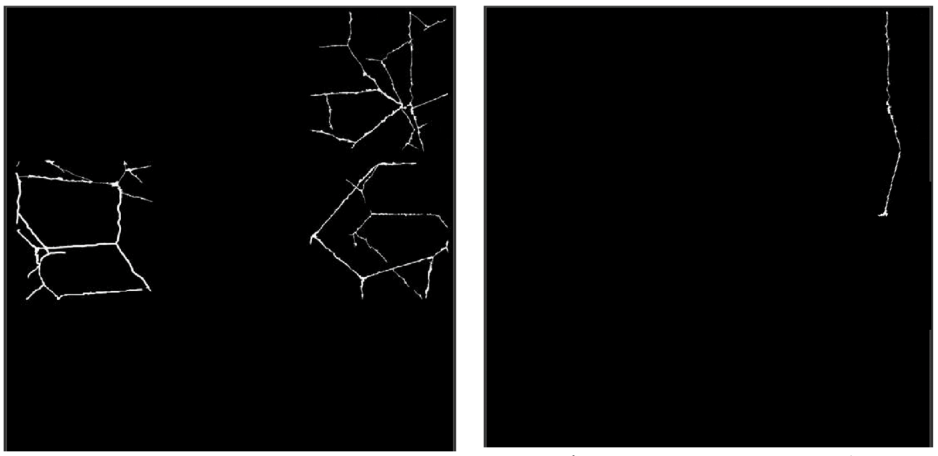


Figure 7. Blender shading nodes used for 3D model.



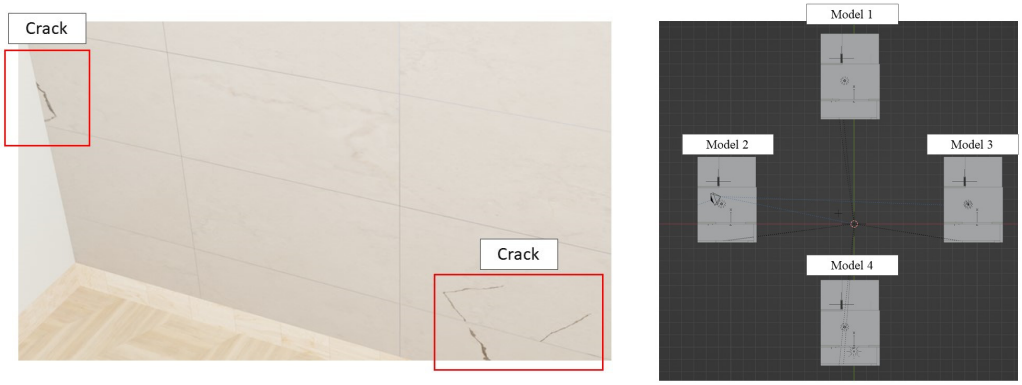
(**a**) Area Image-Normal Crack (**b**) Area Image-Micro Crack

Figure 8. Area information images.

4.4.3. Creation of Defect Models

Defect assets are strategically positioned in areas where tiles serve as finishes within 3D models, such as walls. During the placement procedure, a 2 mm thick grout line is

initially generated and positioned on the wall. Following this, tiles without defects and tiles containing defect assets are randomly arranged on the wall, as illustrated in Figure 9a, to construct the model. Using this approach, approximately 10 variations of defect walls were generated. The size of the tiles ranged from 30 cm × 30 cm to 30 cm × 60 cm, 40 cm × 80 cm, and 60 cm × 60 cm, resulting in four distinct sizes. The tile materials were categorized into two types: stone and ceramic. When combined with the presence or absence of patterns, a total of 11 types were created. Each model, as previously mentioned, was created in four variations, considering the position of the sunlight and the presence of interior lighting, leading to variations as shown in Figure 9b. This approach resulted in the image exhibiting greater variety, including differences in brightness and reflectivity. Different arrangements of defect assets were implemented for each model, ensuring diversity in data. This process completed the preparation for rendering, following which the rendering system was constructed.



(**a**) Virtual Tile Defect Wall (**b**) Virtual Model Arrangement

Figure 9. Tile wall with defect and model arrangement.

4.5. Building an Automatic Rendering System

There are a total of 11 Blender files created, each containing 4 models, with each model requiring 18 images to be rendered, resulting in a total of 792 images to be processed. However, the conventional manual rendering method, capable of rendering only one image at a time, proves inefficient for handling such a large volume of images. Consequently, a rendering system was developed to facilitate the automated generation of multiple images based on predefined parameters. These parameters include the focus target object, focus point, distance from the object to be photographed, rotation angle, shooting height, and the destination for saving rendered images. The center of the target wall to be photographed is selected as the target object, with the rendering target object's coordinates set as the FocusPoint. The camera is set to rotate 10 degrees at a time around this point, totaling a 60-degree rotation to render the images. Additionally, rendering variations based on height are added, focusing on the center of the wall at heights of 0.7 m, 1.4 m, and 2.2 m. Thus, 18 images are rendered from each model, and since 4 models are rendered simultaneously in one Blender file, a total of 72 images are rendered. Moreover, the rendering process simultaneously produces Mask images for segmentation learning purposes.

Segmentation, a crucial technique in digital image processing and computer vision, involves identifying and separating individual objects or specific areas within an image. Its primary goal is to recognize various parts of an image and analyze their characteristics like location, shape, and size in detail. Segmentation divides an image into meaningful segments, enabling detailed extraction of information at the pixel level. For defects such as cracks, which are small and finely detailed, it was determined that the Bounding Box method of image classification is insufficient for accurate detection. For precise detection of cracks, this research uses segmentation to identify the exact location and boundaries of each crack, providing detailed information regarding the length, width, and direction

of the crack, allowing the deep learning model to perform more precise crack detection. Especially, the segmentation of cracks enables clear differentiation from the surrounding environment, preventing confusion with other elements like grout lines. The labeling for segmentation, which involves marking only the target object with a polygon line to differentiate it from the surroundings, is more time-consuming than the Bounding Box method. To automate this process, automatic annotation through Mask images was utilized for efficient preprocessing, and a Mask image rendering function was added. Examples of rendered images and Mask images are shown in Figure 10a,b.

(**a**) Rendered Image (**b**) Rendered Mask Image

Figure 10. Rendered image and mask image samples.

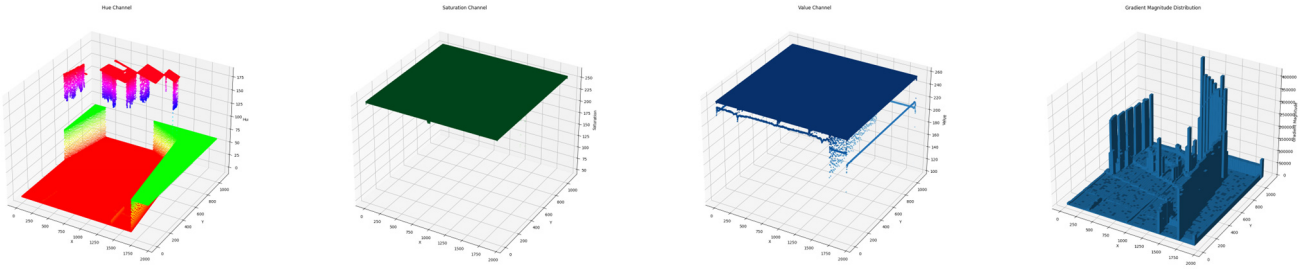
To optimize the generation and annotation of virtual images for machine learning model training, we developed a script in Blender that automates the rendering of realistic and annotated images simultaneously, as outlined in Algorithm 2. This method integrates 'Mix switcher' nodes into the Texture Shading nodes of each component to manage the rendering process. It renders Real and GroundTruth images concurrently, with the GroundTruth images serving as Mask images for annotations. Cracks are marked in red, tiles in blue, and the background in green to mitigate color variations due to the three-dimensional characteristics of the models. This approach not only streamlines the creation of both image types but also ensures consistent, accurate annotations essential for effective model training, enhancing the diversity and utility of the generated images for accurate model evaluation. For rendering, the Cycle, one of Blender's physical render engines, was used, and the Render's Max sample was set to 1024 to ensure the rendering image size was 1920 $\times$ 1080 pixels. The device for rendering was set to GPU compute, and an NVIDIA 3060ti GPU was used.

Algorithm 2 Automated Image Rendering and Mode Switching in Blender

1: Import required libraries and define global configurations
2: Create 'mode_switcher' node group with inputs and output
3: Insert 'mode_switcher' node in target collection materials
4: Define function to toggle between 'Ground Truth' and 'Realistic' modes
5: Render layers and save images for 'real' and 'ground truth' modes
6: Add camera focus on the target object with constraints
7: Relocate camera location for scene variation
8: Automate rendering process for different scenarios

4.6. Creating Segmentation Annotations

Annotation files are generated using the mask images to label the rendered images. These mask images delineate objects with specific colors, such as red, blue, and green. To extract these color-specific areas for annotation, the color range for each object needed to be established. This involved converting the color model of the images from RGB to HSV. Then, to check the HSV information of each pixel, three 3D graphs were formed with Hue, Saturation, and Value values set on the z-axis, and the horizontal and vertical pixel ranges set on the x and y axes, respectively, as shown in Figure 11a. Additionally, the distribution of Gradient Angles for the Canny edge algorithm was checked at the pixel level to detect fine cracks, with results shown in Figure 11b. The HSV range for the red area was determined to be (121, 255), (100, 255), and (0, 251), and, for the green area, it was (50, 105), (100, 255), and (0, 255). The parentheses denote minimum and maximum values. The blue area was excluded from range checking for processing efficiency as all areas except green were considered blue. The Gradient Angle distribution for cracks (red areas) ranged from 0 to 150,000. Through trial and error, the range for the Canny edge algorithm was optimized for detection accuracy with a range from 1 to 1500 found to be most effective.



(**a**) HSV Plot (**b**) Gradient Angle

Figure 11. HSV and Gradient Angle information for automated annotation.

Once the color and Gradient Angle ranges were set, annotations were extracted by adjusting additional parameters such as dilate and erode values, creating index IDs for each label, and modifying the coordinates of the polygon points. Dilate expands the recognized color area while eroding the expanded range allowing for more precise area detection and extraction. These parameter values were also determined through trial and error experimentation, with dilate set to 10,10 and erode to 8,8. After identifying areas, the RETR_EXTERNAL library was used to extract only the outer boundary coordinates of the areas. This approach prevents the misidentification of empty spaces inside a single boundary of a crack, which is crucial for accurate labeling in a deep learning system. The extracted coordinates were in pixel format. The detail for this process is outlined in Algorithm 3.

This script automates the conversion of mask images into a format suitable for machine learning model training. The script initiates by importing essential libraries for image manipulation and data handling. Subsequently, the algorithm iterates through each image, applying color thresholds to segregate different features based on hues in the HSV color space, and isolates these features to generate precise masks. The masks are then refined to handle overlapping areas, thereby ensuring an accurate representation. For each color-coded mask, the script extracts contours to create detailed annotations. This extraction is of paramount importance, as it translates visual data into actionable insights for models. The process culminates in the compilation of these details into a JSON file in a format consistent with the COCO standard, thereby ensuring compatibility with a range of deep learning frameworks. This method enhances the efficiency of preparing large datasets and underscores the script's utility in streamlining the annotation process.

Algorithm 3 Create Annotation File from Mask Images

```
Import required libraries (os, cv2, numpy, json)
Define functions for color checking and mask processing
Initialize variables and set HSV color thresholds
for each image file in the directory do
    Read and convert image to HSV color space
    Initialize and apply color thresholds to create masks
    Refine masks and handle overlapping areas
    for each mask (red, green, blue) do
        Extract contours and create annotations
    end for
end for
Append image details to the 'images' list
Define categories for annotation
Create COCO format output dictionary
Write the output to a JSON file
Import required libraries (os, cv2, numpy, json)
```

The pixel coordinates, obtained from the Python script, originate from the upper-left corner of the image. These coordinates are incompatible with the YOLOv8 deep learning model because YOLO uses normalized (0 to 1) coordinates. To adjust the coordinates for YOLO, the x-coordinates are divided by the width of the image and the y-coordinates by its height. This normalization process ensures that the coordinates range between 0 and 1. This normalization makes the coordinates suitable for use in the YOLO model. The red area's ID was set to 0 and the blue area to 2. ID 1 was omitted in this 3D model as it represents other materials installed on the tiles in real images. After assigning crack as index 0 and tile as index 2, the final index settings were completed. Additionally, the standard YOLO labeling file format, including image append, licenses, and info was established, and the annotation file extraction was conducted. This completes the preparation for training the deep learning model.

5. Results

5.1. Model for Training—YOLOv8

We chose YOLOv8 as our deep learning model because it stands out for its speed, efficiency, ease of use, and impressive precision. Moreover, it offers multiple versions, providing flexibility to choose the most fitting one for our laboratory setup. We specifically opted for this model for object recognition through segmentation. Recent research adopting segmentation techniques has also highlighted the effectiveness of YOLOv8 [93,94].

As illustrated in Figure 12, the YOLOv8 architecture comprises three main components:

- Backbone Network: This is the primary feature extraction module of the YOLOv8, using the CSP (CSP (Cross-Stage Partial): A network design that enhances feature propagation and reuse). Darknet, an enhanced version of the Cross-Stage Partial (CSP) architecture [95]. It processes input images to extract pivotal features for object detection, benefiting from pretraining on datasets such as ImageNet[D] [96].
- Neck Network: Employing a PAN-FPN (PAN-FPN (Path Aggregation Network—Feature Pyramid Network): Combines features at different levels to improve the detection efficacy across scales) structure inspired by PANet, this part of the model enhances the lightweight design while maintaining high performance. It facilitates efficient feature integration across different scales, crucial for detecting objects of varying sizes [97].
- Detection Head: The decoupled head in YOLOv8 separates tasks of classification and bounding box regression. It utilizes Binary Cross-Entropy (BCE) for classification and combines Distribution Focal Loss (DFL) with CIoU loss for precise bounding box predictions [96]. This modular approach optimizes both classification and localization accuracy.

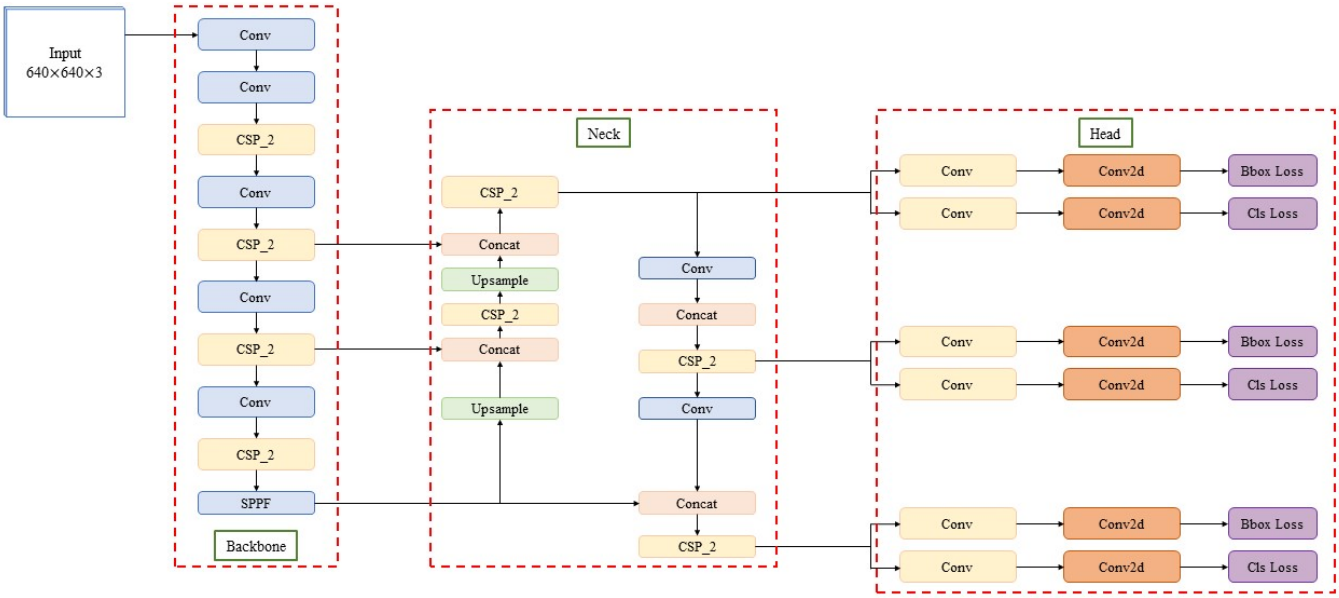


Figure 12. Architecture of Yolov8 model.

YOLOv8 incorporates the Task-Aligned Assigner, which dynamically assigns samples. This feature enhances the model's accuracy and resilience by merging classification scores with the Intersection over Union metric. This integration refines prediction quality and mitigates low-quality prediction boxes. The anchor-free approach of this model significantly contributes to precise object detection across different scales and orientations.

The training environment made use of Google Colab, a cloud-based Jupyter Notebook platform offered by Google. It utilized a T4 GPU, allowing access to 35GB of memory within the Colab environment. The setup included PyTorch version 2.1.0 and CUDA toolkit version 11.8.

5.2. Model Performance Evaluation Method

There are several metrics used to evaluate the performance of object detection models. Commonly used indicators include Accuracy, Precision, Recall, F1 Score, and Average Precision (AP). Accuracy is the ratio of correct predictions among all predictions, while Precision is the ratio of actual positives among those predicted as positive. Recall measures the ratio of actual positives correctly predicted as positive. Average Precision considers the relationship between Precision and Recall as a performance metric, providing a comprehensive evaluation of object detection models. In this study, the F1 Score, mAP, and PR (Precision–Recall) were chosen as the metrics for comparing model performance.

- Precision: As shown in Equation (1), it is defined as the ratio of true positive predictions to the total number of predicted positives and measures the model's accuracy in predicting positive instances.

$$Precision = \frac{\mathrm{TP}}{(\mathrm{TP} + \mathrm{FP})} \tag{1}$$

TP (True Positives): The number of correct positive predictions made by the model.; FP (False Positives): The number of incorrect predictions where the model predicted an object as positive when it is negative.

- Recall: As shown in Equation (2), also known as sensitivity, it is the ratio of true positive predictions to the total number of actual positive instances, assessing the model's ability to identify all relevant instances.

$$Recall = \frac{\text{TP}}{(\text{TP} + \text{FN})} \tag{2}$$

FN (False Negatives): The number of incorrect predictions where the model predicted an object as negative when it is positive.

- F1 Score: The F1 Score calculates the harmonic mean of the model's Precision and Recall; it is useful in assessing models with imbalanced class distributions or when it is crucial to balance both Precision and Recall [98]. The formula for the F1 Score is as follows:

$$F1 = 2 \times \frac{(\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})} \tag{3}$$

TP (True Positive), FP (False Positive), and FN (False Negative) are key indicators that represent how well a model's predictions match actual values, expressed as integers. These indicators are used to calculate the F1 Score, and the results of model training are often represented in a graph with the F1 Score on the y-axis and Confidence values on the x-axis. However, in this study, an F-1 score at a 0.25 confidence threshold was used to analyze that of the Tile and Crack class. Moreover, the author used three types of F-1 score—macro (Equation (3)), weighted (Equation (4)), and micro averaged (Equation (5))—to analyze the F-1 score, aiming to account for data imbalance across classes.

- Macro F1: The Macro F1 Score averages the F1 scores of all classes, providing an equal weight to each, regardless of their frequency.
- Weighted F1: The Weighted F1 Score adjusts for class imbalance by weighting each class's F1 score according to its prevalence in the dataset which is different from Macro F1 metrics.

$$F1_{weighted} = \sum_{i=1}^{N} W_i F1_i \tag{4}$$

- Micro F1: The Micro F1 Score aggregates outcomes across all classes to reflect overall accuracy, emphasizing the model's total effectiveness across all instances.

$$F1_{micro} = 2 \times \frac{(Precision_{micro} \times Recall_{micro})}{(Precision_{micro} + Recall_{micro})} \tag{5}$$

In fact, the Exception class is a very small percentage compared to Tile and Crack classes; therefore, it requires the use of a different type of F-1 score. In this study, the number of data is used for relative weight in obtaining the weighted average F-1 score.

- Confidence represents the degree of certainty in the model's predictions. Typically, object detection models produce a probability value for each prediction, indicating the level of confidence in detecting the object. In this study, both confidence and F-1 scores have been used to present comprehensive learning outcomes across all classes. The F-1 score, which represents the optimal balance between precision and recall attained by the model at the given confidence threshold, is illustrated in the following section.
- The mean average precision is an aggregated measure of performance across multiple classes or over different recall levels. The Average Precision (AP) for a single class is calculated as the area under the PR curve for that class as shown in Equation (6). It can be interpreted as the weighted mean of precisions achieved at each threshold, with the increase in recall from the previous threshold used as the weight.

$$AP = \sum_n (R_n - R_{n-1}) P_n \tag{6}$$

- where P_n and R_n are the precision and recall at the nth threshold. The mean AP (mAP) is obtained by averaging the APs across all classes as shown in Equation (7).

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \quad (7)$$

- where "N" represents the number of classes, the mean average precision (mAP) serves as a single metric summarizing the model's performance across all classes. This metric is commonly used in object detection benchmarks where the detection threshold is varied, and detections are ranked by their predicted scores. In this research, two variants of mAP were utilized: mAP(B) and mAP(M). mAP(B) indicates the best performance achieved by the model at the stage during training for a specific intersection over the Union (IoU) threshold, while mAP(M) represents the average of the mean average precision across all classes. This provides an overview of the model's overall performance across the dataset, without being skewed by exceptionally high or low scores in any particular class. IoU thresholds of 0.5 and 0.5–0.95 were utilized in this study.

The performance graph offers insights into how the model balances Precision and Recall achieving its best performance, which is crucial for real-world applications. The outcomes of training the dataset with the YOLOv8 model are depicted through metrics such as F1 Score, mAP, and PR. By comparing these metrics, the influence of image augmentation using virtual datasets on accuracy is evaluated. This assessment aids in understanding whether virtual image datasets make a meaningful contribution to the results.

5.3. Dataset Creation for Comparative Validation

To validate virtual data, a comparative analysis against real image data was conducted. Table 3 summarizes the datasets used. Two dataset groups were used: the first assessed the standalone value of virtual images and their combination with real images in a hybrid dataset. The second group evaluated accuracy differences influenced by varying mix ratios of real and virtual images.

Table 3. Dataset groups.

Dataset	Image Combination
Dataset: group 1	Real Image 253 Virtual Image 253 Real Image 150 + Virtual Image 150
Dataset: group 2	Real Image 253 + Virtual Image 800 Real Image 800 Real Image 800 + Virtual Image 800

The first dataset group aims to evaluate the value of virtual images as image data by comparing the accuracy differences between real and virtual images. It also confirmed the performance of the hybrid dataset, which combines virtual and real-site images. The results obtained from training on 253 real images (real image dataset) were compared with those from training on 253 randomly selected virtual images (virtual image dataset), along with the accuracy of the Hybrid dataset. The Hybrid dataset consists of a random assortment of 253 images from a total pool of 300 images, maintaining a one-to-one ratio. This evaluation aimed to determine whether virtual images alone could sufficiently replace real images for training deep learning models and to assess the potential for virtual images to substitute real images within the Hybrid dataset.

The second group was designed to evaluate the difference in accuracy depending on the mix ratio of real and virtual images. A dataset mixing 253 real images with 800 virtual images was prepared, and this Hybrid dataset was constituted by mixing randomly selected virtual images from the pool of 792 virtual images and 253 real images. In com-

parison, a dataset of 800 images was prepared by augmenting 253 real images. The augmentation techniques applied included flipping, rotating, shearing, blurring, adding noise, multiplying, and adjusting contrast. This allowed for the assessment of the difference in accuracy between augmenting real images and substituting the augmentation with virtual images. Furthermore, a dataset was created by mixing all images, including 800 virtual images and augmented real images, and this was compared to the two previous datasets to assess the degree of improvement in accuracy when virtual image data and augmented real images were combined. The comparative analysis of these two datasets validated the effectiveness of the virtual image data and evaluated the practical applicability.

In the study, both validation and test datasets consisted exclusively of real images. This configuration was essential to objectively assess the model's performance across various datasets. The consistency in dataset composition for validation and testing ensures that the performance metrics are comparable and reliable, highlighting the true impact of using virtual images during training.

5.4. Learning Results

5.4.1. Results of Dataset Group 1

The aforementioned datasets have undergone deep learning training with consistent parameters: 500 epochs, batch size 128, resized to 640 × 640, learning rate of 0.01, and early stopping near epoch 360. Table 4 displays the learning outcomes for Group 1, presenting object detection performance metrics: mAP, Precision, Recall, and F-1 score.

Table 4. The object detection performance of Group 1.

Dataset	mAP 50(B)	mAP 50-95(B)	mAP 50(M)	mAP 50-95(M)	Precision	Recall	F-1 Score (Confidence)
Real Image 253	0.4895	0.370	0.451	0.312	0.345	0.345	0.45 (0.609)
Virtual Image 253	0.184	0.0844	0.155	0.0714	0.199	0.199	0.2 (0.279)
Real Image 150 Virtual Image 150	0.497	0.399	0.474	0.282	0.430	0.430	0.5 (0.704)

As these results are aggregated across all classes, Tables 5 and 6 detail the results for our primary targets, Tile and Crack, with a threshold set at 0.25. While Table 4 shows the result of learning comprehensively. The Hybrid dataset showed the best performance among them. The key findings of Table 4 are as follows: (1) Except for mAP50-95(M), the hybrid data consistently performs best across other metrics. (2) It is noteworthy that the F-1 score for the hybrid dataset reached 0.704 in confidence, even surpassing the dataset composed solely of an equivalent number of real images. (3) The dataset comprising only virtual images records considerably lower scores. (4) The mAP results showed that Real Image 150 Virtual Image 150, Real Image 253, and Virtual Image 253 performed the best, in that order. These results prompt further discussion, addressed in the following section.

Table 5. Result of Tile class object detection from Group 1.

Dataset	Precision	Recall	F-1 Score (0.25)
Real Image 253	0.862	0.769	0.813
Virtual Image 253	0.4655	0.628	0.535
Real Image 150 Virtual Image 150	0.879	0.8226	0.85

Table 6. Result of Crack class object detection from Group 1.

Dataset	Precision	Recall	F-1 Score (0.25)
Real Image 253	0.356	0.256	0.2984
Virtual Image 253	0.01695	0.25	0.03175
Real Image 150 Virtual Image 150	0.3729	0.373	0.3729

The results of Tile class detection were better than that of Tile class. For detailed inquiry, the key points of Table 5 are as follows: (1) Both the real image and the hybrid datasets show high detection accuracy for tiles. (2) The recall value of the dataset containing only virtual images is respectable at 0.628, but the precision value is relatively low at 0.4655. (3) The hybrid dataset exhibits a higher recognition rate for the tiles, with an F-1 score of 0.85. Following this, Table 6 summarizes the results for the Crack class. It shows the following results: (1) Overall values of Crack detection remained low compared to that of Tile detection. (2) The hybrid dataset achieves the highest accuracy with an F-1 score of 0.3729. (3) The virtual image dataset achieves a recall value similar to the real image dataset at 0.25, but the precision value is notably low at 0.01675.

Figure 13 illustrates the learning outcomes across three datasets, each evaluated using Macro, Micro, and Weighted F-1 scores. These metrics provide insights into model performance under varying conditions of class imbalance and data representation. The Real and Hybrid datasets demonstrate improvements in F-1 scores in the order of Macro, Weighted, and Micro, indicating a consistent enhancement across broader to more specific instances. Conversely, the Virtual dataset exhibits the highest Weighted F-1 score, followed by Micro and Macro. Figure 14 provides a visual representation of the test results obtained from the learning outcomes. The leftmost image serves as the ground truth, while the subsequent images on the right represent the test outcomes. It shows that the virtual dataset exhibits a low detection rate.

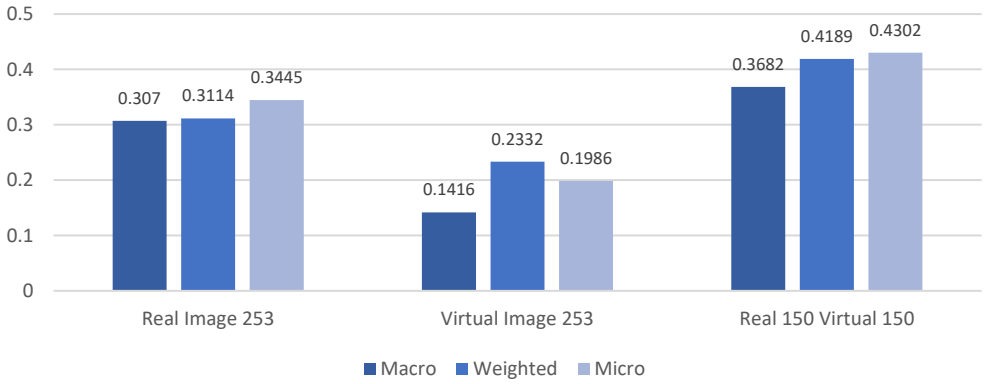
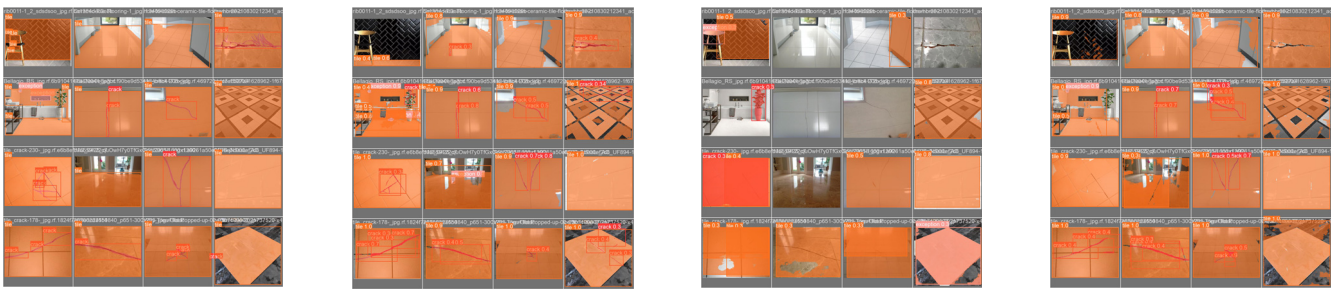


Figure 13. Result by F-1 score types (Group 1).



(**a**) Prediction test (**b**) Real253 (**c**) Virtual253 (**d**) Real150 Virtual150

Figure 14. Prediction Result (Group 1).

5.4.2. Results of Dataset Group 2

Table 7 describes the object detection performance for each dataset, with Tables 8 and 9 detailing the performance specifically for the Tile and Crack classes, respectively. All test images in these datasets were consistently used. For the result, the Real 800 and Virtual 800 datasets achieved the highest performance. The key points in Table 7 are as follows: (1) The dataset containing both 800 real and 800 virtual images exhibits the highest detection accuracy. (2) The F-1 score of 0.55 was observed for the dataset using only 800 real images, which increased to 0.57 when augmented with 800 virtual images. (3) The F-1 score for the dataset with 253 real and 800 virtual images was 0.05 lower than that of the dataset with 800 real images. (4) The precision and recall values for the dataset with 253 real and 800 virtual images at a threshold of 0.25 were 0.0323 and 0.033 higher than that of the dataset with 800 real images. (5) The mAP results showed that Real 800 Virtual 800 had the best performance for all mAP. Real 800 and Real 253 Virtual 800 ranked differently for each mAP. Real 800 achieved a higher mAP score than Real 253 Virtual 800 in mAP50(B) and mAP50-95(B).

Table 7. The object detection performance of Group 2.

Dataset	mAP 50(B)	mAP 50-95(B)	mAP 50(M)	mAP 50-95(M)	Precision	Recall	F1 score (Confidence)
Real 253 Virtual 800	0.49417	0.36563	0.47184	0.319	0.4123	0.413	0.5 (0.698)
Real 800	0.51672	0.37936	0.45243	0.30378	0.38	0.38	0.55 (0.489)
Real 800 Virtual 800	0.52284	0.42912	0.47946	0.31572	0.46	0.46	0.57 (0.627)

Table 8. Result of Tile class object detection from Group 2.

Dataset	Precision	Recall	F-1 Score (0.25)
Real 253 Virtual 800	0.862	0.926	0.893
Real 800	0.8772	0.794	0.833
Real 800 Virtual 800	0.8597	0.891	0.875

Table 9. Result of Crack class object detection from Group 2.

Dataset	Precision	Recall	F-1 Score (0.25)
Real 253 Virtual 800	0.339	0.3704	0.354
Real 800	0.424	0.27174	0.331
Real 800 Virtual 800	0.322	0.432	0.369

Table 8 presents results for the Tile class alone, and the results are as follows: (1) The Real 253 Virtual 800 dataset achieves the highest recall and F-1 values of 0.926 and 0.893, respectively. (2) The Real 800 dataset achieved the highest score in Precision with 0.8772. (3) The Real 800 Virtual 800 dataset achieved 0.891 and 0.875 for Recall and F-1 scores, which is better than the Real 800 dataset but lower than the other.

The key points of Table 9 are as follows: (1) The Real 800 dataset, which uses only real images, achieves the highest precision for the Crack class, but shows a significantly reduced recall value of 0.27. (2) The Real 800 Virtual 800 images dataset's recall value

achieved 0.432 higher than the Real 800 dataset. (3) The Real 253 Virtual 800 dataset achieved better precision than Real 800 Virtual 800 with 0.339 and its recall was higher than the Real 800 dataset with 0.3704.

Figure 15 offers a comparative analysis of the F-1 scores as Figure 13. It shows that the datasets containing virtual images achieved the highest weighted average F-1 scores followed by Micro and Macro. Conversely, datasets consisting of only real images achieved the highest on Macro but lowest on a weighted average. Following Figure 16 visually presents the prediction results obtained from training on each dataset, supporting the comparative analysis.

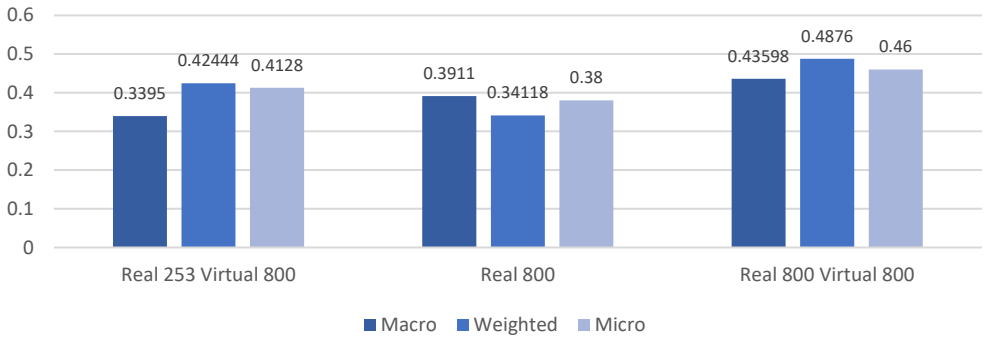


Figure 15. Results obtained using F-1 score types (Group 2).

(**a**) Real800 (**b**) Real253 Virtual800 (**c**) Real800 Virtual800

Figure 16. Prediction result (Group 2).

6. Discussion and Limitations

6.1. Discussion

The results for Group 1 in Table 4 demonstrate that the object detection rate is higher with the same number of hybrid datasets compared to real images alone, indicating that the images generated in the virtual environment are adequate and valuable as data. The mAP, precision, recall, and F-1 scores all exhibit improvement. However, it is noticeable that the accuracy is substantially low for the dataset solely using virtual images. This discrepancy is likely attributed to the domain shift phenomenon, resulting from training with a dataset of virtual images and validating with a dataset of real images.

Moreover, the comparative analysis of the F-1 scores showed that the Real and Hybrid datasets in Group 1 demonstrated improvements in F-1 scores in the order of Macro, Weighted, and Micro, indicating a consistent enhancement across broader to more specific instances. While the Virtual dataset in Group 1 exhibits the highest Weighted F-1 score, followed by Micro and Macro. This variation underscores the distinct impacts of data types on model training, reflecting the virtual dataset's limited ability to generalize across diverse conditions compared to real or hybrid data configurations. For the Group 2 datasets

combining 253 real and 800 virtual images, as well as those with 800 real and virtual images, the highest scores are observed in Weighted F-1, followed by Micro and Macro. This pattern indicates that these datasets achieve better average performance across classes, particularly when class imbalance is adjusted for in the weighting. In contrast, the dataset comprised solely of 800 real images exhibits the highest scores in Macro and Micro F-1, with Weighted F-1 being the lowest. This suggests that this dataset performs well in generalizing across all classes equally and in capturing all positive instances but may underperform in classes with fewer samples.

The difference in these patterns, compared to previous results where Real and Hybrid datasets consistently showed enhancements in the order of Macro, Weighted, and Micro, suggests that the inclusion of a larger proportion of real images tends to stabilize performance across different class distributions. This highlights the influence of real-world data in training models that are robust and less sensitive to class imbalance, as opposed to virtual data that might not capture the full spectrum of real-world variability. However, it is noticeable that the results of train solely on virtual images are markedly low.

Domain shift is a common phenomenon in studies using synthetic images. When validation is performed on synthetic images, the accuracy tends to be very high; however, when validation on real images, the accuracy drops significantly [72]. Nevertheless, this phenomenon disappears when a hybrid dataset is used, leading to improved accuracy. This demonstrates that incorporating real images into a dataset comprising virtual images can mitigate domain shift, allowing virtual images to effectively train deep learning models alongside real images. However, in order to train using only virtual images, it is necessary to address the domain shift phenomenon. This can be achieved by either minimizing the domain shift phenomenon through domain generalization or randomization or by minimizing the reality gap between the virtual image and the real image.

The accuracy improvements need to be examined in detail for each class type. For the Tile class, accuracy is generally high. Both the real image dataset and the hybrid dataset exhibit F-1 values above 0.8, along with good precision and recall values. These values surpass those in Table 4, which aggregates all classes together. These results indicate that the aggregated metrics are influenced by the Exception and Background classes. The virtual image dataset shows a low Precision value but a notably high Recall value. While the Precision values for the real image dataset and the hybrid dataset do not exhibit a significant difference, a 7% improvement in Recall suggests that the integration of virtual images leads to a notable enhancement in Recall.

Similarly, for the Crack class, the hybrid dataset displays the best results, highlighting the impact of the virtual image on the recall value. This indicates that the images generated in the virtual environment for the Crack class possess sufficient diversity to mirror real-world images, facilitating sophisticated pixel-level labeling. In summary, the augmented images generated by the virtual environment in this study can serve as training data and are effective in improving the recall value. However, the reliability of datasets consisting only of virtual images drops sharply due to the domain shift phenomenon.

The comparison in Group 2 involves three datasets: one with 253 real images mixed with 800 virtual images, another with 800 augmented real images created by augmentation of 253 real images, and a third with 1600 real images mixed with 800 virtual images.

Table 7 reveals that the dataset with only 800 real images outperforms the dataset with 253 real and 800 virtual images in mAP50(B), mAP50-90(B), and F-1 scores for all classes combined. Table 7 reveals that the dataset with only 800 real images outperforms the dataset with 253 real and 800 virtual images in mAP50(B), mAP50-90(B), and F-1 scores for all classes combined. However, the results of mAP50(M) and mAP50-90(M) were opposite. Furthermore, the Confidence threshold of the 253 real and 800 virtual datasets was 0.698, which is much higher than that of the real 800 dataset, which achieved 0.489. This suggests that incorporating virtual images can provide enhanced diversity and quality over mere augmentation, potentially leading to better model generalization and robustness. It implies that the integration of virtual images alongside real images not only compensates for

the limitations seen with augmentation techniques but also enhances the dataset's overall effectiveness in diverse scenarios. When we look more closely at the metrics that evaluate only the Tile class, the hybrid dataset performs well on all metrics. This result aligns with the findings in Group 1. This is also evident in the Crack class, where the hybrid dataset has a higher recall value, despite having a higher precision for the real image dataset.

The results across all classes demonstrate the highest values for the hybrid dataset of 800 real images and 800 virtual images. However, the Tile class does not exhibit better performance compared to the hybrid dataset of 253 real images and 800 virtual images. "Others" and "Background" classes have relatively high values, resulting in better overall metric results. However, the precision and recall metrics for the "Tile" and "Crack" classes, which are the focus of this experiment, exhibit lower values. This suggests that the detection of tile objects is not significantly impacted by the proliferation of a small number of real images. Figure 17 shows the comparison of the F-1 score of all datasets when only Tile and Crack classes are taken into account.

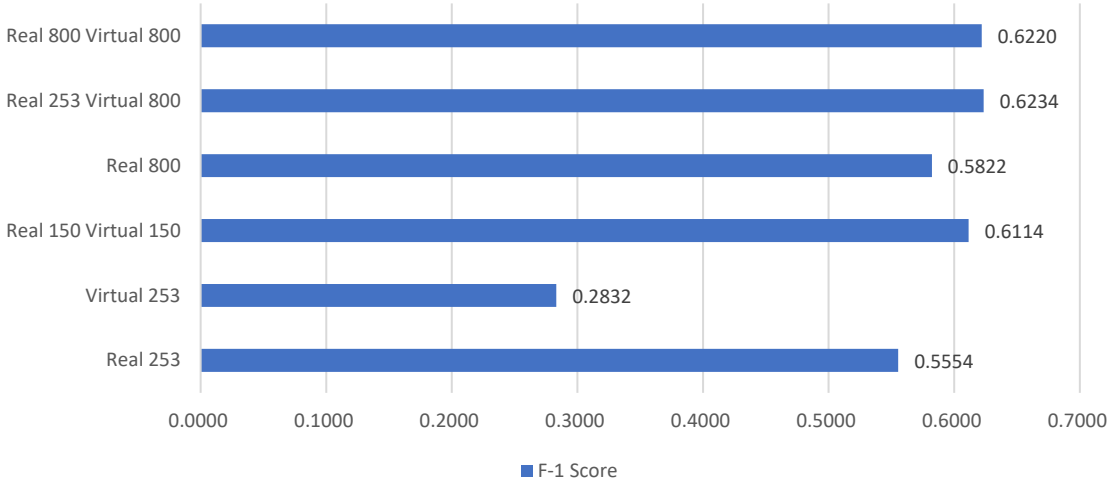


Figure 17. F-1 scores considering Tile and Crack only on the whole dataset.

Furthermore, the authors identified additional considerations for data generation to improve accuracy. In this study, the model's tile joints were created in white and bright, in accordance with the prevailing practices in new constructions. Upon analyzing the model's real image prediction result, it was observed that the tile joints in older buildings often appear darker due to the effects of aging and exposure, which influenced the accuracy of our crack detection. This discrepancy highlights the importance of adapting virtual models to include varied joint colors in order to better simulate real-world conditions. Such adjustments are crucial for enhancing model robustness and applicability across diverse building conditions.

It is worth noting that in the test results, there were instances where tiles were detected without cracks being defined as any class and appearing as empty areas. This suggests that the learning rate for the Tile class was adequate whilst the learning rate for the Crack class was inadequate. Furthermore, it is acknowledged that micro-cracks present a challenge in their detection, particularly when visibility is reduced due to greater shooting distances. Our study, therefore, emphasizes the need for enhanced image resolution and criteria including a minimum shooting distance. Micro-cracks are often undetected due to low clarity when images are captured from afar, which can significantly impact the model's ability to perform accurate defect detection. To address this, we suggest employing high-resolution cameras that can capture finer details more clearly. Furthermore, the results of a series of experiments must be evaluated in order to ascertain the minimum distance at which micro-cracks can be detected.

6.2. Limitation

Despite the strengths and efficacy of this study, there are several limitations acknowledged. The most prominent limitation is domain shift, which occurs when models trained solely on virtual images are tested on real images. This phenomenon prevents the exclusive use of virtual image datasets. To mitigate this, techniques such as GAN-based style transfer or domain adaptation can be employed to reduce the reality gap between virtual and real images. Unlike conventional applications of style transfer, which aim to make training data appear more realistic, the author is considering applying style transfer to test images to make real images appear virtual.

This innovative approach has the potential to optimize the use of virtual images for data augmentation, thereby improving model generalization. Another concern is the time-consuming process of manually annotating real images at the pixel level, which took several hours to label the real images. Currently, there are several attempts to automatically annotate images based on deep learning techniques. The most popular method is an automatic annotation system using unsupervised learning [99]. This method is based on clustering techniques to distinguish the characteristics of images and thus distinguish classes. There have been various studies based on this method, but it is still difficult to apply [100]. Moreover, the application of unsupervised learning to objects that occur randomly rather than in a regular pattern, such as cracks in tiles, is more challenging. Therefore, further research is required to accomplish a fully automated image annotation system.

The automated labeling system was applied to virtual images, and a random selection was manually reviewed. However, as the number of images and the sample size for random checks grows, it becomes crucial to adopt an objective evaluation method. One potential approach is to manually label sample images and then compare them with the system's output using Intersection over Union (IoU) metrics. Furthermore, the current study's emphasis on tile cracks restricts its wider applicability. The claim that the study demonstrates effective methods for all types of defects based on only one subject is weak.

To bolster the validity of this study, the methodology should be expanded to encompass various types of defects. For instance, in order to gain further insight into the tile defects, it would be beneficial to conduct experiments involving additional defects such as those related to falling off and tilting. Furthermore, it would be advantageous to apply this approach to defects in other trades in order to ensure broad applicability. Nonetheless, this research marks a significant advancement in automated defect detection in construction. It proposes a methodology that uses virtual image augmentation to improve model accuracy. By addressing these limitations and exploring suggested enhancements, the study's methodology could be refined. This would contribute to the advancement of automated, intelligent construction management systems and propel the field towards more accurate and autonomous quality management in construction projects.

7. Conclusions

The construction industry is increasingly acknowledging the necessity of automated detection models for identifying defects. However, to ensure the effectiveness of such models, achieving high accuracy is crucial, which in turn demands a substantial and diverse dataset. Yet, gathering such data directly from construction sites is exceedingly time-consuming and laborious, and maintaining its diversity and quality poses a significant challenge. To address these obstacles, this study introduces a novel hybrid approach involving the creation of virtual 3D models and the generation of virtual image data through automated rendering and annotation systems. This methodology commences with the collection of basic field images, followed by the development of a hierarchical classification system based on these images. This strategy guarantees data diversity, while the subsequent establishment of an automated system enhances convenience and efficiency. Moreover, through the utilization of segmentation and rendering mask images, this method minimizes the labor required for labeling for segmentation purposes. The efficacy of the virtual image data produced by this approach has been validated.

When employing an equal number of images, the F-1 score of the hybrid dataset, which combined virtual and real images in a 1:1 ratio, exhibited a 4.4% increase at 0.4329 compared to the 0.4154 F-1 score achieved solely using real images. This outcome substantiates the effectiveness of utilizing virtual image data and underscores the meaningfulness of incorporating virtual data. Furthermore, the dataset formed by augmenting real images yielded an F-1 score of 0.4499, whereas the hybrid dataset, amalgamating augmented real and virtual images, scored 0.4990 in the F-1 score, marking a 10% enhancement. This underscores that leveraging virtual image data significantly augments data and thereby improves accuracy, establishing its efficacy akin to real data. These findings validate the effectiveness of the proposed methodology. However, when exclusively relying on virtual image data, the F-1 score was notably low. It is presumed that challenges in prediction arose when there were disparities in shooting distances between virtual and real environments. While virtual images are created with high resolution and clarity, enhancing the distinction between defective and non-defective areas, real-world images often suffer from variable sharpness due to random shooting distances. This can result in less detailed defect depiction, leading to decreased accuracy in predictions. To mitigate this, we suggest the implementation of standardized shooting distances for real images, mirroring the conditions used for generating virtual images. Furthermore, adjusting the focus parameters within the virtual environment to simulate real-world imaging conditions could bridge the gap in image quality and enhance the model's predictive accuracy.

While the primary focus of this study was on tile cracks, the methodology holds relevance for addressing various types of defects across different construction contexts. By crafting and modeling datasets specific to each defect type and suitable for the respective work phase, a multitude of image data can be generated. This data can then be utilized to advance the development of automated detection models for a range of defects that were previously difficult to identify. Nonetheless, in instances where the complexity of certain defects necessitates intricate modeling, further refinement will be necessary for future research efforts. While this study primarily focuses on improving tile crack detection from a data perspective, it is clear that further advancements could also come from enhancing the machine learning model itself. For more robust crack detection, future research might consider refining the model's architecture, optimizing loss functions, or tweaking training parameters. Such modifications could tailor the model more precisely for recognizing subtle differences in crack patterns, potentially leading to significant improvements in detection accuracy. Overall, this methodology has the potential to enhance the accuracy of comprehensive defect management systems in the future.

Author Contributions: Conceptualization, S.-M.C. and H.-S.C.; methodology, S.-M.C.; software, S.-M.C.; validation, S.-M.C. and H.-S.C.; formal analysis, S.-M.C. and S.J.; investigation, S.-M.C. and S.J.; resources, H.-S.C. and S.J.; data curation, S.-M.C.; writing—original draft preparation, S.-M.C.; writing—review and editing, H.-S.C.; visualization, S.-M.C.; supervision, H.-S.C.; project administration, S.-M.C.; funding acquisition, H.-S.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Ajou University research fund (S2023G000100466) and the research fund from AP 10 program (S2024G000100041).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset is available on request from the authors.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Maksym, A. Business Strategy Paradigm Shift: Building Resilience in a Disrupted Economy after the COVID-19 Pandemic. Order No. 29211349. Master's Thesis, Webster University, Webster Groves, MO, USA, 2021. Available online: https://www.proquest.com/dissertations-theses/business-strategy-paradigm-shift-building/docview/2671699692/se-2 (accessed on 20 June 2024).
2. Luo, H.; Xiong, C.; Fang, W.; Love, P.E.D.; Zhang, B.; Ouyang, X. Convolutional neural networks: Computer vision-based workforce activity assessment in construction. *Autom. Constr.* **2018**, *94*, 282–289. [CrossRef]
3. Yang, J.; Shi, Z.; Wu, Z. Vision-based action recognition of construction workers using dense trajectories. *Adv. Eng. Inform.* **2016**, *30*, 327–336. [CrossRef]
4. Barbosa, F.; Woetzel, J.; Mischke, J.; Ribeirinho, M.J.; Sridhar, M.; Parsons, M.; Bertram, N.; Brown, S. Reinventing Construction Through a Productivity Revolution. 2017. Available online: https://www.mckinsey.com/industries/capital-projects-and-infrastructure/our-insights/reinventing-construction-through-a-productivity-revolution (accessed on 20 June 2024).
5. Ilyas, M.; Khaw, H.Y.; Selvaraj, N.M.; Jin, Y.; Zhao, X.; Cheah, C.C. Robot-Assisted Object Detection for Construction Automation: Data and Information-Driven Approach. *IEEE/ASME Trans. Mechatron.* **2021**, *26*, 2845–2856. [CrossRef]
6. Busta, H. KPMG Report: Construction Industry Slow to Adopt New Technology. 14 September 2016. Available online: https://www.constructiondive.com/news/kpmg-report-construction-industry-slow-to-adopt-new-technology/426268/ (accessed on 20 June 2024).
7. Manyika, J.; Ramaswamy, S.; Khanna, S.; Sarrazin, H.; Pinkus, G.; Sethupathy, G.; Yaffe, A.; Digital America: A Tale of the Haves and Have-Mores. McKinsey Global Institute 2015, pp. 1–120. Available online: https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/digital-america-a-tale-of-the-haves-and-have-mores (accessed on 20 June 2024).
8. Construction Workers Mutual Aid Association. *Monthly Statistics: Construction Progress Payment and Construction Workforce Insights Report (December 2023)*; Construction Workers Mutual Aid Association: Seoul, Republic of Korea, 2023.
9. Sungjin, K.; Soowon, C. Castro-Lacouture Daniel Dynamic Modeling for Analyzing Impacts of Skilled Labor Shortage on Construction Project Management. *J. Manag. Eng.* **2020**, *36*, 04019035. [CrossRef]
10. Alomari, O.M. Identification and categorization of building defects. *Civ. Eng. Archit.* **2022**, *10*, 438–446. [CrossRef]
11. Abou Shaar, B. Adaptable Three Dimensional System for Building Inspection Management. Master's Thesis, University of Waterloo, Waterloo, ON, Canada, 2012. Available online: http://hdl.handle.net/10012/6923 (accessed on 20 June 2024).
12. Zhou, Y.; Ji, A.; Zhang, L. Sewer defect detection from 3D point clouds using a transformer-based deep learning model. *Autom. Constr.* **2022**, *136*, 104163. [CrossRef]
13. Perez, H.; Tah, J.H. Deep learning smartphone application for real-time detection of defects in buildings. *Sensors* **2021**, *28*, e2751. [CrossRef]
14. Choi, M.; Kim, S.; Kim, S. Semi-automated visualization method for visual inspection of buildings on BIM using 3D point cloud. *J. Build. Eng.* **2024**, *81*, 108017. [CrossRef]
15. Xu, Y.; Li, D.; Xie, Q.; Wu, Q.; Wang, J. Automatic defect detection and segmentation of tunnel surface using modified Mask R-CNN. *Measurement* **2021**, *178*, 109316. [CrossRef]
16. Luo, H.; Lin, L.; Chen, K.; Antwi-Afari, M.F.; Chen, L. Digital technology for quality management in construction: A review and future research directions. *Dev. Built Environ.* **2022**, *12*, 100087. [CrossRef]
17. Kim, H.; Liu, X.; Ahn, E.; Shin, M.; Shin, S.W.; Sim, S. Performance assessment method for crack repair in concrete using PZT-based electromechanical impedance technique. *NDT E Int.* **2019**, *104*, 90–97. [CrossRef]
18. Liu, Y.F.; Cho, S.; Spencer, B.F., Jr.; Fan, J.S. Concrete Crack Assessment Using Digital Image Processing and 3D Scene Reconstruction. *J. Comput. Civ. Eng.* **2016**, *30*, 04014124. [CrossRef]
19. Gao, Y.; Mosalam, K.M. PEER Hub ImageNet: A large-scale multiattribute benchmark data set of structural images. *J. Struct. Eng.* **2020**, *146*, 04020198. [CrossRef]
20. Mostafa, K.; Hegazy, T. Review of image-based analysis and applications in construction. *Autom. Constr.* **2021**, *122*, 103516. [CrossRef]
21. Han, K.K.; Golparvar-Fard, M. Potential of big visual data and building information modeling for construction performance analytics: An exploratory study. *Autom. Constr.* **2017**, *73*, 184–198. [CrossRef]
22. Paneru, S.; Jeelani, I. Computer vision applications in construction: Current state, opportunities & challenges. *Autom. Constr.* **2021**, *132*, 103940. [CrossRef]
23. Fang, W.; Ding, L.; Love, P.E.D.; Luo, H.; Li, H.; Peña-Mora, F.; Zhong, B.; Zhou, C. Computer vision applications in construction safety assurance. *Autom. Constr.* **2020**, *110*, 103013. [CrossRef]
24. Wang, T.; Gan, V.J.L. Automated joint 3D reconstruction and visual inspection for buildings using computer vision and transfer learning. *Autom. Constr.* **2023**, *149*, 104810. [CrossRef]
25. Perez, H.; Tah, J.H.; Mosavi, A. Deep learning for detecting building defects using convolutional neural networks. *Sensors* **2019**, *19*, 3556. [CrossRef]
26. Xu, S.; Wang, J.; Wang, X.; Shou, W. Computer vision techniques in construction, operation and maintenance phases of civil assets: A critical review, ISARC. In Proceedings of the International Symposium on Automation and Robotics in Construction, Banff, AB, Canada, 21–24 May 2019; IAARC Publications: Cambridge, UK, 2019; pp. 672–679. [CrossRef]

27. Xu, S.; Wang, J.; Shou, W.; Ngo, T.; Sadick, A.; Wang, X. Computer vision techniques in construction: A critical review. *Arch. Comput. Methods Eng.* **2021**, *28*, 3383–3397. [CrossRef]
28. Chai, Y.T.; Wang, T. Evaluation and decision-making framework for concrete surface quality based on computer vision and ontology. *Eng. Constr. Archit. Manag.* **2022**, *30*, 4881–4913. [CrossRef]
29. Spencer, B.F., Jr.; Hoskere, V.; Narazaki, Y. Advances in computer vision-based civil infrastructure inspection and monitoring. *Engineering* **2019**, *5*, 199–222. [CrossRef]
30. Fang, W.; Love, P.E.D.; Luo, H.; Ding, L. Computer vision for behaviour-based safety in construction: A review and future directions. *Adv. Eng. Inform.* **2020**, *43*, 100980. [CrossRef]
31. Hamdan, A.; Taraben, J.; Helmrich, M.; Mansperger, T.; Morgenthal, G.; Scherer, R.J. A semantic modeling approach for the automated detection and interpretation of structural damage. *Autom. Constr.* **2021**, *128*, 103739. [CrossRef]
32. Gao, Y.; Kong, B.; Mosalam, K.M. Deep leaf-bootstrapping generative adversarial network for structural image data augmentation. *Comput.-Aided Civ. Infrastruct. Eng.* **2019**, *34*, 755–773. [CrossRef]
33. D'Addario, J. New Survey Finds British Businesses Are Reluctant to Proactively Share Data. 2020, Open Data Institute, 29. Available online: https://theodi.org/article/new-survey-finds-just-27-of-british-businesses-are-sharing-data/ (accessed on 20 June 2024).
34. Amarù, S.; Marelli, D.; Ciocca, G.; Schettini, R. DALib: A Curated Repository of Libraries for Data Augmentation in Computer Vision. *J. Imaging* **2023**, *9*, 232. [CrossRef] [PubMed]
35. Frid-Adar, M.; Klang, E.; Amitai, M.; Goldberger, J.; Greenspan, H. Synthetic data augmentation using GAN for improved liver lesion classification. In Proceedings of the 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018), Washington, DC, USA, 4–7 April 2018; pp. 289–293. [CrossRef]
36. Creswell, A.; White, T.; Dumoulin, V.; Arulkumaran, K.; Sengupta, B.; Bharath, A.A. Generative adversarial networks: An overview. *IEEE Signal Process. Mag.* **2018**, *35*, 53–65. [CrossRef]
37. Wu, A.N.; Stouffs, R.; Biljecki, F. Generative Adversarial Networks in the built environment: A comprehensive review of the application of GANs across data types and scales. *Build. Environ.* **2022**, *223*, 109477. [CrossRef]
38. Kim, D.H.; Lee, D.; Lee, H.J.; Min, Y.G.; Park, I.; Cho, H. Analysis of importance by defect type in apartment con-struction. *J. Korea Inst. Build. Constr.* **2020**, *20*, 357–365. [CrossRef]
39. Lu, Q.; Lin, J.; Luo, L.; Zhang, Y.; Zhu, W. A supervised approach for automated surface defect detection in ceramic tile quality control. *Adv. Eng. Inform.* **2022**, *53*, 101692. [CrossRef]
40. Huynh, N.T. An approach for classifying ceramic tile defects based on a two-dimensional Genetic CNN algorithm. *Neural Comput. Appl.* **2024**, *36*, 385–397. [CrossRef]
41. Li, Y.; Fang, J. Detection of Surface Defects of Magnetic Tiles Based on Improved YOLOv5. *J. Sens.* **2023**, *2023*, 2466107. [CrossRef]
42. Wang, H.; Jin, P.; Wang, J.; Wu, J.; Li, D.; Cheng, H. Research on Ceramic Tile Defect Detection Based on Enhanced YOLOv5. In Proceedings of the 2023 International Conference on the Cognitive Computing and Complex Data (ICCD), Huaian, China, 21–22 October 2023; pp. 78–83. [CrossRef]
43. Kovilpillai, J.J.A.; Jayanthy, S. An optimized deep learning approach to detect and classify defective tiles in production line for efficient industrial quality control. *Neural Comput. Appl.* **2023**, *35*, 11089–11108. [CrossRef]
44. Cao, X.; Yao, B.; Chen, B.; Wang, Y. Multi-defect detection for magnetic tile based on SE-U-Net. In Proceedings of the 2020 IEEE International Symposium on Product Compliance Engineering-Asia (ISPCE-CN), Chongqing, China, 6–8 November 2020; pp. 1–6. [CrossRef]
45. Sioma, A. Automated control of surface defects on ceramic tiles using 3D image analysis. *Materials* **2020**, *13*, 1250. [CrossRef]
46. Cao, X.; Chen, B.; He, W. Unsupervised Defect Segmentation of Magnetic Tile Based on Attention Enhanced Flexible U-Net. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 5011110. [CrossRef]
47. Liu, T.; He, Z.; Lin, Z.; Cao, G.Z.; Su, W.; Xie, S. An adaptive image segmentation network for surface defect detection. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *35*, 8510–8523. [CrossRef] [PubMed]
48. Cao, T.; Song, K.; Xu, L.; Feng, H.; Yan, Y.; Guo, J. Balanced multi-scale target score network for ceramic tile surface defect detection. *Measurement* **2024**, *224*, 113914. [CrossRef]
49. Stephen, O.; Maduh, U.J.; Sain, M. A machine learning method for detection of surface defects on ceramic tiles using convolutional neural networks. *Electronics* **2021**, *11*, 55. [CrossRef]
50. Wan, G.; Fang, H.; Wang, D.; Yan, J.; Xie, B. Ceramic tile surface defect detection based on deep learning. *Ceram. Int.* **2022**, *48*, 11085–11093. [CrossRef]
51. Zhu, Z.; Zhu, P.; Zeng, J.; Qian, X. A Surface Fatal Defect Detection Method for Magnetic Tiles based on Semantic Segmentation and Object Detection: IEEE ITAIC (ISSN: 2693-2865). In Proceedings of the 2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), Chongqing, China, 17–19 June 2022; pp. 2580–2586. [CrossRef]
52. Dong, G.; Sun, S.; Wu, N.; Chen, X.; Huang, P.; Wang, Z. A rapid detection method for the surface defects of mosaic ceramic tiles. *Ceram. Int.* **2022**, *48*, 15462–15469. [CrossRef]
53. Zhu, H.; Hwang, B.G.; Ngo, J.; Tan, J.P.S. Applications of smart technologies in construction project management. *J. Constr. Eng. Manag.* **2022**, *148*, 04022010. [CrossRef]
54. Lu, Y.; Duanmu, L.; Zhai, Z.J.; Wang, Z. Application and improvement of Canny edge-detection algorithm for exterior wall hollowing detection using infrared thermal images. *Energy Build.* **2022**, *274*, 112421. [CrossRef]

55. Luo, Q.; Gao, B.; Woo, W.L.; Yang, Y. Temporal and spatial deep learning network for infrared thermal defect detection. *NDT E Int.* **2019**, *108*, 102164. [CrossRef]
56. Ichi, E.; Dorafshan, S. Effectiveness of infrared thermography for delamination detection in reinforced concrete bridge decks. *Autom. Constr.* **2022**, *142*, 104523. [CrossRef]
57. Chen, C.; Xiao, B.; Zhang, Y.; Zhu, Z. Automatic vision-based calculation of excavator earthmoving productivity using ze-ro-shot learning activity recognition. *Autom. Constr.* **2023**, *146*, 104702. [CrossRef]
58. Lee, K.; Hong, G.; Sael, L.; Lee, S.; Kim, H.Y. MultiDefectNet: Multi-class defect detection of building façade based on deep convolutional neural network. *Sustainability* **2020**, *12*, 9785. [CrossRef]
59. Dais, D.; Bal, I.E.; Smyrou, E.; Sarhosis, V. Automatic crack classification and segmentation on masonry surfaces using convolutional neural networks and transfer learning. *Autom. Constr.* **2021**, *125*, 103606. [CrossRef]
60. Wang, Y.; Han, Y.; Wang, C.; Song, S.; Tian, Q.; Huang, G. Computation-efficient deep learning for computer vision: A survey. In *Cybernetics and Intelligence*; IEEE: Piscataway, NJ, USA, 2024. [CrossRef]
61. Høye, T.T.; Ärje, J.; Bjerge, K.; Hansen, O.L.; Iosifidis, A.; Leese, F.; Mann, H.M.; Meissner, K.; Melvad, C.; Raitoharju, J. Deep learning and computer vision will transform entomology. *Proc. Natl. Acad. Sci. USA* **2021**, *118*, e2002545117. [CrossRef]
62. Brunetti, A.; Buongiorno, D.; Trotta, G.F.; Bevilacqua, V. Computer vision and deep learning techniques for pedestrian de-tection and tracking: A survey. *Neurocomputing* **2018**, *300*, 17–33. [CrossRef]
63. Bao, Y.; Tang, Z.; Li, H.; Zhang, Y. Computer vision and deep learning–based data anomaly detection method for structural health monitoring. *Struct. Health Monit.* **2019**, *18*, 401–421. [CrossRef]
64. Halevy, A.; Norvig, P.; Pereira, F. The unreasonable effectiveness of data. *IEEE Intell. Syst.* **2009**, *24*, 8–12. [CrossRef]
65. Arora, N. Image Augmentation Using Generative Adversarial Networks. *CASS Stud.* **2020**, *4*, 2.
66. Shorten, C.; Khoshgoftaar, T.M. A survey on image data augmentation for deep learning. *J. Big Data* **2019**, *6*, 1–48. [CrossRef]
67. Shamsolmoali, P.; Zareapoor, M.; Granger, E.; Zhou, H.; Wang, R.; Celebi, M.E.; Yang, J. Image synthesis with adversarial networks: A comprehensive survey and case studies. *Inf. Fusion* **2021**, *72*, 126–146. [CrossRef]
68. Alqahtani, H.; Kavakli-Thorne, M.; Kumar, G. Applications of generative adversarial networks (gans): An updated review. *Arch. Comput. Methods Eng.* **2021**, *28*, 525–552. [CrossRef]
69. de Melo, C.M.; Torralba, A.; Guibas, L.; DiCarlo, J.; Chellappa, R.; Hodgins, J. Next-generation deep learning based on simulators and synthetic data. *Trends Cogn. Sci.* **2022**, *26*, 174–187. [CrossRef]
70. Qin, Z.; Liu, Z.; Zhu, P.; Xue, Y. A GAN-based image synthesis method for skin lesion classification. *Comput. Methods Programs Biomed.* **2020**, *195*, 105568. [CrossRef]
71. Sandfort, V.; Yan, K.; Pickhardt, P.J.; Summers, R.M. Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks. *Sci. Rep.* **2019**, *9*, 16884. [CrossRef]
72. Lei, Y.; Harms, J.; Wang, T.; Liu, Y.; Shu, H.; Jani, A.B.; Curran, W.J.; Mao, H.; Liu, T.; Yang, X. MRI-only based synthetic CT generation using dense cycle consistent generative adversarial networks. *Med. Phys.* **2019**, *46*, 3565–3581. [CrossRef]
73. Lu, Y.; Chen, D.; Olaniyi, E.; Huang, Y. Generative adversarial networks (GANs) for image augmentation in agriculture: A systematic review. *Comput. Electron. Agric.* **2022**, *200*, 107208. [CrossRef]
74. Dewi, C.; Chen, R.; Liu, Y.; Tai, S. Synthetic Data generation using DCGAN for improved traffic sign recognition. *Neural Comput. Appl.* **2022**, *34*, 21465–21480. [CrossRef]
75. Li, S.; Zhao, X. High-resolution concrete damage image synthesis using conditional generative adversarial network. *Autom. Constr.* **2023**, *147*, 104739. [CrossRef]
76. Bang, S.; Baek, F.; Park, S.; Kim, W.; Kim, H. Image augmentation to improve construction resource detection using generative adversarial networks, cut-and-paste, and image transformation techniques. *Autom. Constr.* **2020**, *115*, 103198. [CrossRef]
77. Goodfellow, T.S.; Zaremba, W.; Ian, V.C. Improved techniques for training gans. *Adv. Neural Inf. Process. Syst.* **2016**. [CrossRef]
78. Soltani, M.M.; Zhu, Z.; Hammad, A. Automated annotation for visual recognition of construction resources using synthetic images. *Autom. Constr.* **2016**, *62*, 14–23. [CrossRef]
79. Hwang, J.; Kim, J.; Chi, S. Site-optimized training image database development using web-crawled and synthetic images. *Autom. Constr.* **2023**, *151*, 104886. [CrossRef]
80. Kim, J.; Kim, D.; Lee, S.; Chi, S. Hybrid DNN training using both synthetic and real construction images to overcome training data shortage. *Autom. Constr.* **2023**, *149*, 104771. [CrossRef]
81. Lee, H.; Jeon, J.; Lee, D.; Park, C.; Kim, J.; Lee, D. Game engine-driven synthetic data generation for computer vision-based safety monitoring of construction workers. *Autom. Constr.* **2023**, *155*, 105060. [CrossRef]
82. Barrera-Animas, A.Y.; Delgado, J.M.D. Generating real-world-like labelled synthetic datasets for construction site applications. *Autom. Constr.* **2023**, *151*, 104850. [CrossRef]
83. Assadzadeh, A.; Arashpour, M.; Brilakis, I.; Ngo, T.; Konstantinou, E. Vision-based excavator pose estimation using synthetically generated datasets with domain randomization. *Autom. Constr.* **2022**, *134*, 104089. [CrossRef]
84. Roberts, M.; Ramapuram, J.; Ranjan, A.; Kumar, A.; Bautista, M.A.; Paczan, N.; Webb, R.; Susskind, J.M. Hypersim: A photo-realistic synthetic dataset for holistic indoor scene understanding. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 10912–10922. [CrossRef]
85. Schieber, H.; Demir, K.C.; Kleinbeck, C.; Yang, S.H.; Roth, D. Indoor synthetic data generation: A systematic review. *Comput. Vis. Image Underst.* **2024**, *2024*, 103907. [CrossRef]

86. Ying, H.; Sacks, R.; Degani, A. Synthetic image data generation using BIM and computer graphics for building scene understanding. *Autom. Constr.* **2023**, *154*, 105016. [CrossRef]
87. Hong, Y.; Park, S.; Kim, H.; Kim, H. Synthetic data generation using building information models. *Autom. Constr.* **2021**, *130*, 103871. [CrossRef]
88. Rampini, L.; Re Cecconi, F. Synthetic images generation for semantic understanding in facility management. *Constr. Innov.* **2024**, *24*, 33–48. [CrossRef]
89. Neuhausen, M.; Herbers, P.; König, M. Using synthetic data to improve and evaluate the tracking performance of construction workers on site. *Appl. Sci.* **2020**, *10*, 4948. [CrossRef]
90. Mailhe, C.; Ammar, A.; Chinesta, F. On the use of synthetic images in deep learning for defect recognition in industrial infrastructures. In Proceedings of the 2023 6th International Conference on Machine Vision and Applications, Singapore, 10–12 March 2023; pp. 81–87. [CrossRef]
91. Siu, C.; Wang, M.; Cheng, J.C. A framework for synthetic image generation and augmentation for improving automatic sewer pipe defect detection. *Autom. Constr.* **2022**, *137*, 104213. [CrossRef]
92. Qiu, S.; Cai, B.; Wang, W.; Wang, J.; Zaheer, Q.; Liu, X.; Hu, W.; Peng, J. Automated detection of railway defective fasteners based on YOLOv8-FAM and synthetic data using style transfer. *Autom. Constr.* **2024**, *162*, 105363. [CrossRef]
93. Bai, R.; Wang, M.; Zhang, Z.; Lu, J.; Shen, F. Automated Construction Site Monitoring Based on Improved YOLOv8-Seg Instance Segmentation Algorithm. *IEEE Access* **2023**, *11*, 139082–139096. [CrossRef]
94. Yang, T.; Zhou, S.; Xu, A.; Ye, J.; Yin, J. An approach for plant leaf image segmentation based on YOLOV8 and the improved DEEPLABV3. *Plants* **2023**, *12*, 3438. [CrossRef]
95. Wang, G.; Chen, Y.; An, P.; Hong, H.; Hu, J.; Huang, T. UAV-YOLOv8: A small-object-detection model based on improved YOLOv8 for UAV aerial photography scenarios. *Sensor* **2023**, *23*, 7190. [CrossRef]
96. Yang, G.; Wang, J.; Nie, Z.; Yang, H.; Yu, S. A lightweight YOLOv8 tomato detection algorithm combining feature enhancement and attention. *Agronomy* **2023**, *13*, 1824. [CrossRef]
97. Wang, X.; Gao, H.; Jia, Z.; Li, Z. BL-YOLOv8: An improved road defect detection model based on YOLOv8. *Sensor* **2023**, *23*, 8361. [CrossRef]
98. Powers, D.M. Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv* **2020**, arXiv:2010.16061. [CrossRef]
99. Novozámský, A.; Vit, D.; Šroubek, F.; Franc, J.; Krbálek, M.; Bílkova, Z.; Zitová, B. Automated Object Labeling For Cnn-Based Image Segmentation. In Proceedings of the 2020 IEEE International Conference on Image Processing (ICIP), Abu Dhabi, United Arab Emirates, 25–28 October 2020; pp. 2036–2040. [CrossRef]
100. Adnan, M.M.; Rahim, M.S.M.; Rehman, A.; Mehmood, Z.; Saba, T.; Naqvi, R.A. Automatic Image Annotation Based on Deep Learning Models: A Systematic Review and Future Challenges. *IEEE Access* **2021**, *9*, 50253–50264. [CrossRef]