DEVELOPMENT OF A ROBOTIC CONTROL SYSTEM FOR AUTOMATED BUILDING CRACK INSPECTION

SIWEI CHANG

PhD

The Hong Kong Polytechnic University

2023

The Hong Kong Polytechnic University

Department of Building and Real Estate

Development of a Robotic Control System for Automated Building Crack Inspection

Siwei Chang

A thesis submitted in partial fulfilment of the requirements

for the degree of Doctor of Philosophy

January 2023

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

Siwei Chang

Abstract

To monitor the quality and health of structures and make sure they are constructed in accordance with regulatory requirements, construction inspection is essential. In most cases, professional inspectors are engaged to inspect the quality defects using visual inspection or measurement devices such as rulers. However, with the fast development of construction industry, the drawbacks of conventional inspection techniques—such as a shortage of skilled labor, high costs, and poor efficiency and accuracy—become increasingly serious. The challenges grow much worse when inspecting building cracks because cracks occur the most frequently. They can exist in any type of building component, such as slabs, walls, or beams, as well as during any stage of construction, such as when a building is being built or demolished. The urgent practical needs, therefore, leave the motivations and opportunities for advanced building crack inspection techniques.

To alleviate the aforementioned concerns, computer vision techniques, such as the convolutional neural network (CNN), are increasingly integrated to achieve the automated building crack inspection. In the process of using CNN to inspect cracks, the images should be captured first to build the datasets. The datasets are then imported into the pretrained CNN model to predict and demonstrate whether there are cracks in the images. To improve the speed and accuracy of CNNs, researchers have been focusing on

developing or modifying their architectures. Although CNNs do contribute to automatic crack inspection to some extent, they are less efficient and inconsistent when compared to the fully automated inspection techniques.

This research developed a robotic control system for the automated building crack inspection with considering the mentioned limitation into account. The robotic control system is fully automated, flexible, robust, and user-friendly in comparison to the current computer vision-based crack inspection method. Controlled by the developed robotic control system, the inspection robot can assist or even replace manual works by automatically, remotely, smoothly, and continuously inspecting building cracks. To build the robotic control system, the following research was explored: 1) The development trend of construction inspection robotics was investigated to target the supporting technologies for the development of the robotic inspection system. 2) A lightweight CNN model was designed for the development of robotic vision. 3) A fuzzy logic controller enabled wall follower algorithm was designed for the robotic navigation. 4) A web user interface was designed for the robotic visualization. A series simulation and on-site validation were carried out to validate the feasibility. It has been proven that the developed robotic control system can be successfully employed in robot platforms to conduct building crack inspection works, including video stream capture, CNN-based crack inspection, autonomous navigation in building environments, and the demonstration of inspection outcomes, without human intervention. Consequently, to reduce the reliance on skilled inspectors and increasing productivity and accuracy of building crack inspection.

Publications

*An asterisk * indicates corresponding author.*

Journal Papers (Published)

1. **Chang, S.**, Siu, M. F. F., Li, H., & Luo, X. (2022). Evolution pathways of robotic technologies and applications in construction. Advanced Engineering Informatics, 51, 101529. https://doi.org/10.1016/j.aei.2022.101529
2. Law, K. K., **Chang, S.***, & Siu, M. F. F. (2022). Factors Influencing Adoption of Construction Robotics in Hong Kong's Industry: A Multistakeholder Perspective. Journal of Management in Engineering, 38(2), 04021096. https://doi.org/10.1061/(asce)me.1943-5479.0001011
3. Kwok, T. W., **Chang, S.***, & Li, H. (2022). Factors affecting unitized curtain wall system adoption for Hong Kong's high-rise residential buildings: a multi-stakeholder perspective. Engineering, Construction and Architectural Management. https://doi.org/10.1108/ECAM-04-2022-0359

Journal Papers (Under review)

1. **Siwei Chang.**, Jiyi Chen, Ming Fung Francis Siu., & Heng Li. (2022). Robotic Technology in Construction Inspection: Historical Review and Development Trend. Expert Systems with Applications. Manuscript ID: ESWA-D-22-06580 (Under Review).

2. **Siwei Chang.**, Heng Li., & Xin Fang. (2022).A Lightweight Convolutional Neural Network for Automated Crack Inspection. Engineering Applications of Artificial Intelligence. Manuscript ID: EAAI-22-4984 (Under Review).

3. **Siwei Chang.**, Ming Fung Francis Siu., & Heng Li. (2022). Development of a fuzzy logic controller for autonomous navigation of building inspection robots in unknown environments. Journal of Computing in Civil Engineering. Manuscript ID: CPENG-5060R1 (Under Review).

4. Haitao Wu., Heng Li., Hung Lin-Chi., ZhenYu Peng., **Siwei Chang**., Yue wu. (2022). Thermal Image-based Hand Signal Recognition for Safe Worker-Robot Collaboration in the Construction Industry: A Feasible Study. Advanced Engineering Informatics. Manuscript ID: ADVEI-D-22-01890 (Under Review).

Book Chapter (Published)

1. **Chang, S.**, & Siu, M. F. F. (2022). Computer Vision-Based Techniques for Quality Inspection of Concrete Building Structures. https://doi.org/10.5772/intechopen.104405

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisor, Professor Heng Li. It is my great honor to be his Ph.D. student. From the beginning to the end, Professor Li has always shown unreserved guidance and support for my research. He shares with me all the valuable materials related to my research topic, he guides me on how to write a good research paper, and he also recommends excellent researchers to communicate with and discuss technical problems. His passion for research greatly inspired me. Besides, he cares and treats students with sincerity. He is full of support and fatherly encouragement every time I meet difficulties. I am so lucky to have such a good supervisor, and I really appreciate all his contributions, support, time, and patience.

Secondly, I would like to thank the Research Grants Council (RGC) of Hong Kong, the Hong Kong Polytechnic University, the Faculty of Construction and Environment, and the Department of Building and Real Estate for offering and supporting me in pursuing my PhD study.

Meanwhile, I would like to thank my colleagues, Mr. Kai Qi, Ms. Qing Yang Zhao, Ms. Rong Yan Li, and Mr. Bo Wen Zheng, for comforting me with their kindness and accompanying me whenever I am under great stress. They remind me how colorful and joyful a Ph.D. life can be. I also thank my colleagues in the Smart Construction Lab and everyone who helped me.

Last but not least, I would like to thank my family, who always show me their endless love, care, support, and encouragement.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

CITF	Construction Innovation and Technology Fund
BIM	Building Information Modelling
CV	Computer Vision
CNN	Convolutional Neural Network
SLAM	Simultaneous Localization and Mapping
SSH	Socket Shell
GUI	Graphical User Interface
PASS	Building Performance Assessment Scoring System
MBIS	Mandatory Building Inspection Scheme
MWIS	Mandatory Window Inspection Scheme
NDT	Non-Destructive Testing
VT	Visual Testing
PTP	Point-To-Point
VR	Virtual Reality
UAV	Unmanned Aerial Vehicles
AI	Artificial Intelligence
FLC	Fuzzy Logic Controller
NLTK	Natural Language Toolkit
WoS	Web of Science
LDA	Latent Dirichlet Allocation
NDT	Nondestructive Testing
ROS	Robot Operating System

PID	Proportional Integral Derivative
ER	Electrical Resistivity
IE	Impact-Echo
Adam	Adaptive Moment Estimation
YOLOv3	You Look Only Once v3
MFs	Membership Functions
HTML	Hypertext Markup Language
RWD	Responsive Web Design
CSS	Cascading Style Sheets
IE	Internet Explorer

List of Symbols

$\boldsymbol{D}$	Literature set
$\boldsymbol{W}$	Each word that consists of paper d
$\boldsymbol{T}$	The set of generated topics
$\boldsymbol{P}_{tj}$	Probability of each topic to each literature
$\boldsymbol{n}_{w}$	The number of common words of literature d and topic t
$\boldsymbol{N}_{w}$	The total number of words in literature d
$\boldsymbol{P}_{wi}$	The probability of each word to each topic
$\boldsymbol{N}_{wi}$	The total number of words in topic t_j, literature d and topic t
$\boldsymbol{f}_{wi}$	The occurrence of word W_i, topic t_j appears in the total word set W
$\boldsymbol{\alpha}$	The hyperparameter that determines the distributions of the independent variables
$\boldsymbol{x}_{i}$	The probability of topic to literature, or word to topic
$\boldsymbol{C}$	Label of the input
$\boldsymbol{X}_{[j]}$	Outputs of x_0 to x_j
$\boldsymbol{W}_{ic}$	The weight of the *i-th* convolutional layer
$\boldsymbol{F}_{hi}$	The height of the convolutional filters
$\boldsymbol{F}_{wi}$	The width of the convolutional filters
$\boldsymbol{F}_{ni}$	The number of the convolutional filters
$\boldsymbol{I}_{i\text{-}1}$	The number of feature map inputs in the *i-1-th* convolutional layer

W_{if}	The weight of the *i-th* fully connected layer
F_{ii}	The number of inputs in the *i-th* fully connected layer
F_{oi}	The number of outputs in the *i-th* fully connected layer
O_i	The output dimension in the *i-th* layer
I_i	The input dimension in the *i-th* layer
D_{Ii}	The convolutional filter size in the *i-th* layer
P_i	The padding pixel in the *i-th* layer
S_i	The convolutional filter stride in the *i-th* layer
x_b, y_b	The heading directions of the robot
α	The orientations of the robot
V_l	The velocity of the left wheels
Vr	The velocity of the right wheels
V	The linear velocity
ω	The angular velocity
μ_A	Fuzzy degree
v^*	The crisp linear velocity
$\mu_{(vi)}$	The fuzzy degree of the membership function in the speed fuzzy set
V_i	The centroid position of the *i-th* membership function
ω^*	The crisp linear velocity
$\mu_{(\omega i)}$	The fuzzy degree of the *i-th* membership function in the rotation fuzzy set
ω_i	The centroid position of the *i-th* membership

Chapter1: Introduction

1.1 Robotics in construction

Construction robotics is currently receiving more and more attention in addressing the emerging challenges of the traditional construction process. By employing robotic platforms for construction activities, certain improvements can be envisioned, including: 1) greater working efficiency. Constantly working on a repetitive cycle can be demonstrated. 2) Better working quality. By executing tasks with mathematical programming and electronic sensors, precise construction can be guaranteed. 3) Enhanced working security. Robotic system can be utilized to conduct hazard construction works, such as quality inspection in cantilevered balconies of high-rise buildings, without human intervention.

Therefore, scholars, entrepreneurs, and the government all express favorable views to embrace this cutting-edge technology (You et al., 2018). For example, the Hong Kong Construction Industry Council and Development Bureau have proposed establishing a HK$ 1 billion Construction Innovation and Technology Fund (CITF) to encourage enterprises and practitioners to adopt innovative constructive technologies such as Building Information Modeling (BIM), machines and robots, and modular integrated (Citf, 2020). Regarding entrepreneurs, a growing number of organizations, such as Fastbrick Robotics, Shimizu Corporation, and the Country Garden, have committed to the development of construction robotics (Brehm, 2019, Heiming et al., 2020, New York University, 2020, Wagner et al., 2020, Dörfler, K, et al., 2016, Pritschow et al., 1996). From an academic perspective, the development of various construction robotics, such as the wearable robotic exoskeleton (Cho et

al., 2018, Yu et al., 2018, Liu et al., 2020), which assists workers in lifting heavy objects, the gantry-type robotic system to realize automated beam assembling (Chu et al., 2013), and the underwater robotic track to level rubble on the seabed for port construction (T.S. Kim, et al., 2014), has gradually become the research focus.

1.2 Research scope

Among various categories of construction robotics, this research narrowed down the research scope to the development of building crack inspection robotics for the following reasons.

1) Building crack inspection plays an important role in ensuring the safety, economy, and long-term viability of construction activities. Building cracks, often caused by foundation movements or excessive external loads, occur more frequently than other defects (Billah, et al., 2020). As reported by BRE Group (BRE, 2021), the cracks with a width of 5mm to 25mm may generate severe damages to building structures. These structural flaws have a significant impact on the serviceability and stability of buildings (Chitte et al., 2018). For example, a 40-year-old oceanfront condo building in Florida collapsed on June 27, 2021, because of the neglect of wall cracks. As noticed, the cracked or crumbling concrete, the interior cracks, and the cracks at the corners of windows and doors are the significant signs of this tragedy. Therefore, it is essential to inspect building cracks carefully and thoroughly.

2) The conventional building crack inspection method is inefficient. Commonly, building cracks are inspected by professional inspectors. The inspectors examine the cracks by walking along the buildings and marking them with their own eyes. Because building cracks can appear in diverse building components, such as walls,

stairs, and columns, as well as at every stage of construction, such as the under or post-construction stage, the categories and numbers of cracks are massive, and incomplete and inaccurate manual inspection happens often. Meanwhile, as more young people are drawn to high-tech industries, the shortage of experienced inspectors is becoming severe. As a result, hiring qualified inspectors is growing more expensive. The cost of professional liability insurance alone might reach $1,400 per person each year. These emerging problems motivate the advancement of building crack inspection techniques.

3) Automated and efficient building crack inspection work is possible with construction robotics. Being controlled by the mathematical programming and detecting surroundings with electrical sensors, the robot can accurately, completely, and autonomously imitate the inspection behaviors of “walking along buildings” and “checking cracks.” Due to the extensive needs and interests of the construction industry, it is also possible to foresee lower investment and quick returns when investing in the development of building crack inspection robotics.

1.3 Research problem statement

To develop the expected robotic functions for building crack inspection, research was conducted in the areas of robotic vision, autonomous navigation, and visualization. To identify the supporting technologies for the development of the robotic control system, the development trend of construction inspection robotics was first investigated from both robotic mechanical and controlling technologies.

Research problems of targeting the supporting technologies for the development of building crack inspection robotics. Because identifying a development trend

indicates patterns that emerge at different times, the construction inspection robotics development trend was investigated to target the most applicable supporting technologies. The majority of previous studies that describe the development trend of construction robotics were identified through analyzing literatur e data since academic research publications offer insightful information for researchers and industry practitioners (Darko and Chan, 2016). For instance, to investigate the trend of construction robotics from the application viewpoint, a systematic review of 52 papers was undertaken (Gharbia et al., 2020). Mi Pan et al. (Pan et al., 2020) anticipated that the future deployment of construction robotics is significantly related with technological and social advancements within the economic and political ecosystem using the scenario approach and a critical literature analysis. In the meantime, Shiyao Cai et al. (Cai et al., 2021) used CiteSpace to identify the supporting technologies for the development of construction robotics, including RFID and computer vision. They provided CiteSpace the bibliographic data of 5522 papers. Although the findings promote the advancement of construction robotics to some extent, the following gaps still exist:1) The review that focuses on the robotics technologies is still lacking. 2) The discussion of the robotic building crack inspection is given less attention. Therefore, in order to identify the supporting robotic technologies for the development of building crack inspection robots, its development trend was first discussed from both the perspective of robotic mechanisms and controlling technologies.

Research problems of robotic vision for the building crack inspection robot. With the boom of computer vision (CV) techniques, the convolutional neural network (CNN), a CV algorithm, attracts the most attention to realize automated inspection

and enable computers instead of humans to visually examine building cracks. To start incorporating the CNN technique into construction inspection, the pre-trained CNNs have been deployed directly on practical inspection projects. For example, automatically inspecting bridge cracks by processing captured images using ResNet-18 (Lee et al., 2018, Deng et al., 2019). Compared with the innovations in CNN designs, the data bank, including both photographic images and infrared thermal images (Yang et al., 2019), is more valuable at that time. Recently, a growing number of academics have begun to focus on developing new CNN architectures apart from building datasets. Since widening and deepening CNNs is thought to be an effective technique to increase prediction accuracy, a number of the modified CNNs were developed with deep and wide designs (Shin et al., 2016). For example, the CNNs designed based on the 16-layer model VGG-16 (Ahmed, 2019) and the 51-layer ResNet-101 for road crack inspection (Dong et al., 2021). Although the proposed CNNs indeed contribute to the automated crack inspection, nearly all the designed deep CNNs need to be processed using powerful GPUs, such as the Nvidia GTX 1080 Ti (Li and Zhao, 2019) to compute millions or billions of parameters. As such, it is inconvenient for a number of robotic platforms to implement CNNs because most of the robot motor controllers, such as the Raspberry Pi, are powered by CPUs instead of GPUs. Therefore, to make CNN applicable for the majority of robotic platforms, lightweight CNNs, that can be executed in various robotic micro-processers, are needed.

Research problems of robotic navigation for the building crack inspection robot. One of the most challenging problems in operating mobile robotics is navigation (Pandey et al., 2017). While much effort has carried into developing computer

vision algorithms for the robotic vision of building inspection robots, little has focused on designing navigation strategies. As a result, most of the robotic platforms are required to be manually controlled to the inspection spots before inspecting building cracks. The motion of the teleoperated building inspection robot, for example, is controlled by operators using joysticks and virtual reality (Tang and Yamada 2011). To realize the autonomous navigation of construction robots, the simultaneous localization and mapping (SLAM) technique (Durrant and Bailey 2006) has begun to be employed. However, due to the size, constant change, and complexity of terrains present on construction sites, such as ramps, sand piles, and excavation trenches, SLAM's higher computational cost and lower map updating efficiency can easily lead to the system getting stuck. Therefore, a local navigation strategy that does not heavily rely on map construction is necessary to control the building crack inspection robot to travel safely and autonomously in dynamic surroundings.

Research problems of robotic visualization for the building crack inspection robot. Developing robotic visualization is essential for presenting robotic programming to users. The inspection results processed by the host robots are typically shown on the screens of the master computer using the socket shell (SSH), a network communication protocol allowing IP address-based communication between the master and host computers. The "SSH-Y" or "SSH-X" commands are recommended to display the outcomes in a graphical window. Although displaying the inspection results via an SSH connection benefits, the display lag issue cannot be avoided due to the slow transmission speed. Rather than using the SSH protocol, previous studies have developed a variety of graphical user interfaces (GUIs) to

fluently demonstrate the crack inspection results, either the crack images or the inspection video streams, in real-time (Lim, et al., 2011, Ryew et al., 2000, Yuan et al., 2022). The GUIs are indeed multifunctional platforms to provide effective interaction between users and the inspection robots. The following key problems, however, need to be taken into consideration: 1) The GUI requires a considerable amount of storage space, which makes it challenging for CPU-driven robotic platforms to operate smoothly. 2) The GUI isn't flexible enough. On desktops or portable electronics including mobile phones, a GUI need to be installed first. It is challenging to develop adaptive GUIs that automatically adjust to the device screens at the same time. Therefore, it is worthwhile to develop a web user interface to visualize building crack inspection results fluidly, constantly, in real time, and with flexibility.

To sum up, the research problems that need to be explored to develop the building crack inspection robotics are: 1) In order to guide the design of the robotic control system, it is necessary to investigate the supporting development technologies. 2) A lightweight CNN needs to be developed for robotic vision. 3) To drive the inspection robots autonomously in the unknown environment, a navigation strategy needs to be developed. 4) A web-user interface needs to be developed to visualize the inspection outcomes for users.

1.4 Research aim and objectives

The present research aims to develop a robotic control system for fully automated building crack inspection. To fulfill the research purpose, the following research objectives were carried out specifically:

1) To investigate supporting technologies for the development of building crack inspection robots.

2) To develop a lightweight CNN for the robotic vision.

3) To develop a fuzzy logic controller-enabled wall-following algorithm for robotic autonomous navigation.

4) To develop a web user interface for the robotic visualization.

1.5 Research significance

On a theoretical level, the current study first identified and classified historical robotic technologies in construction inspection. The identified topics cover a broad scope, including both robotic mechanisms, vision, and navigation algorithms. The advantages, disadvantages, suitable application areas, research emphasis over the period, and future trends of each robotic technology were also summarized. Second, for the development of lightweight CNN architectures, a lightweight CNN model as well as the concept of reducing CNN weight were shared. Thirdly, an effective autonomous navigation technique for driving the crack inspection robots was introduced using a robust fuzzy logic design which improves the wall-following behavior. Finally, a feasible technique for the smooth and continuous robotic visualization was described.

On a practical level, the proposed robotic control system, which integrates robotic vision, navigation, and visualization, provides the industry a fully automated robotic inspection strategy for routine building crack inspection. This system can be easily coded in a variety of robotics hardware platforms to control them to autonomously

conduct building crack inspection works. In addition, the benefits of construction automation, such as high efficiency, high accuracy, and labor savings, can be envisioned via using crack inspection robotics to support or even replace manual inspections.

1.6 Overview of the thesis

Seven chapters fill up the remainder of the thesis. A literature review is presented in Chapter 2 to illustrate the need for developing building crack inspection robotics by introducing the historical of construction robotics, the manual crack inspection process, and the current automatic crack inspection methods. The research outline, which is introduced in Chapter 3, describes the general research logic and processes. The four specific research objectives are realized by introducing the specific research concepts, designs, techniques, results, simulations, and on-site validations from Chapter 4 to Chapter 7. Finally, Chapter 8 presents the conclusion, discussion, and recommendations for potential research topics.

Chapter2: Literature Review[1]

This section examines existing literature on automatic building crack inspection technology in Section 2.1 and robotic technology in construction in Section 2.2 to demonstrate overview knowledge and emphasize the need of developing robotic control systems for building crack inspection.

2.1 Automatic building crack inspection technology

During the crack inspection process, the professional inspectors are hired to inspect cracks and assess a score to the building's quality in accordance with Building Performance Assessment Scoring System (PASS) for new buildings, and Mandatory Building Inspection Scheme (MBIS) and Mandatory Window Inspection Scheme (MWIS) for existing buildings. To inspect the building defects, the non-destructive testing (NDT) has been employed most frequent (Chakraborty et al., 2019). Various NDT techniques, such as eddy current testing (Hamia et al., 2014, Yuan et al., 2021, Omer, 2020), impact-echo testing (Yu et al., 2021, Hashimoto et al., 2020), and infrared thermography testing (Li et al., 2020, Liu et al., 2021, Puthiyaveettl et al., 2021), have been applied on-site. Among them, the visual testing (VT) technique, serves as the primary testing method, is seen as the most widely used NDT technique in examining cracks due to its cost-effectiveness

[1] This chapter is based on a published study and being reproduced with the permission of Elsevier.

Chang, S., Siu, M. F. F., Li, H., & Luo, X. (2022). Evolution pathways of robotic technologies and applications in construction. Advanced Engineering Informatics, 51, 101529. https://doi.org/10.1016/j.aei.2022.101529

Chang, S., & Siu, M. F. F. (2022). Computer Vision-Based Techniques for Quality Inspection of Concrete Building Structures. https://doi.org/10.5772/intechopen.104405

and convenience. For example, using VT to inspect cracks on the floor surface or the Ottoman buildings (Zoidis et al., 2013, Kilic et al., 2015). Although VT is an easy and low-cost NDT technique, it has been gradually updated and replaced by the automated inspection technique to avoid the shortcomings of human errors on the testing accuracy and efficiency (Alzubaidi et al., 2021).

At the beginning, researchers recommended to employ UAVs to first automatically capture the building images. The captured images are then examined by the inspectors in the offices instead of on-sites. It has been proved that using the proposed technique assist in improving inspection efficiency and minimizing safety hazards (Cha et al., 2017, Tong et al., 2018). However, it is still time-consuming and labor intensive to manually identify cracks from thousands of images, neither in front of the computers nor inspecting in construction sites. To solve this problem, computer vision technique has been continuously implemented to realize the automatic crack analyzing from captured images.

“Computer vision” is defined as an interdisciplinary field that enables computers to recognize and interpret environments from digital images or videos (Huang et al., 1996). By automatically processing images and videos, computer vision-based inspection technologies enable efficient, accurate, and low-cost crack inspection. Various techniques in the computer vision field, such as semantic segmentation and object inspection, have been developed and applied to date (Feng et al., 2019). Among them, image classification considered the most basic computer vision techniques. The motivation of image classification is to identify the categories of input images. Different from human recognition, an image is firstly presented as

three-dimensional array of numbers to computer. The value of each number ranges from 0 (black) to 255 (white). An example is shown in **Figure 2.1**. The crack image is 256 pixels wide, 256 pixels tall, and has three color channels (RBG). Therefore, this image generates 256×256×3 = 196608 input numbers. The input array is then computed using computer vision algorithms to transform the numbers to a specific label that belongs to an assigned set of categories. One of the computer vision algorithms: CNN has become dominant in image classification tasks (Yamashita et al., 2018). CNN is a form of deep learning model for computing grid-shaped data. The central idea of CNN is to identify the image classification by capturing its features using filters. The features are then output to a specific classification by a trained weight and biases matrix. The input pixels are transformed to an output label through three main modules in CNN: convolution, pooling, and fully connected layer. The convolution and pooling layers are used to extract image features. The fully connected layer is used to determine the weight and biases matrix and to map the extracted features into specific labels.

Although employing CNNs for automated building crack inspection has attracted the most interest, it lacks continuity and flexibility and is inconvenient for on-site implementation. The images or videos are required to be obtained first and then computed using pretrained CNN models in fixed computers. Motivated by this research gap, the present study focused on integrating CNNs into robotic platforms that achieve fully automated inspection process.

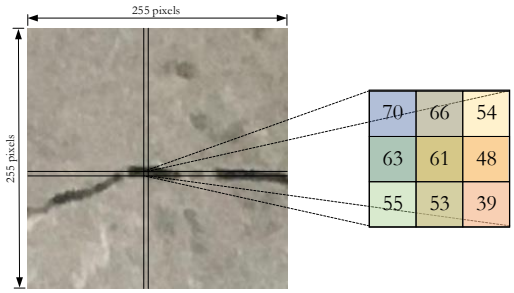


Figure 2. 1 An example of the input number array

To summarize, rather than focusing solely on computer vision techniques, the current research aimed to develop robotic technologies for building crack inspection in order to fill the research gap of achieving a fully automated inspection process.

2.2 Robotic technology in construction

The development of construction robotic technologies can be traced back to 1980s. To explore the development patterns, the present research reviewed and analyzed four million linguistic terms in 581 related papers, which were selected after two-step searching (Chang et al., 2022). By clustering the occurrence probabilities of research topics using the k-means algorithm, four evolutionary stages of construction robotics: Stage I (1983-1999), Stage II (2000-2008), Stage III (2009-2015) and Stage IV (2016-2021) can be found, as shown in **Figure 2.2**.

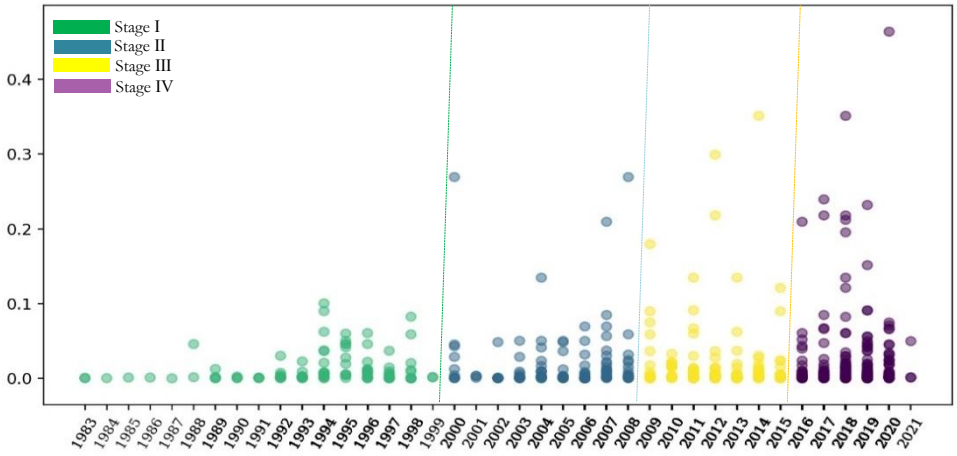


Figure 2. 2 Evolutionary stages of construction robotics

Between Stages I and IV, research topics related to construction robotics became increasingly popular in construction domains. Notably, sample numbers increased particularly rapidly after Stage III, showing that needs for automatic construction emerged strongly after 2015. As the time span of each evolutionary stage became shorter, the technology changed more rapidly, and its applications in practice became more frequent. The first stage can be seen as the introductory stage according to the definition of industry life cycle (Klepper, 1997). Because the main aim in the introductory stage was to create new and unique construction robots as much as possible, the occurrence probability (refers to the y-axis of **Figure 2.2**.) of each robotic topic is relatively low while the number of the topics is relatively high. A feature in this stage is that the entire market was highly fragmented, with no specific development standards. As such, the competition between different affiliates is scarce or non-existent and the early stage was the most innovative stage. Differently, stage II to stage IV were the growth stages, which aimed to enhance and optimize the existing inventions. The growth rate in developing construction robotic increased rapidly from 24.1% to 64.7%. In a growth stage, market

competition becomes fierce, the most valuable robotic technologies remain, and the less valuable are phased out. The prototypes that have high occurrence become more apparent year by year. The most valued prototypes show up among the reference samples, helping to form developing standards.

In Stage I, 99 out of 581 samples were grouped. The robotic mechanical related and the robotic controlling related papers appear 49 and 48 times, respectively, reflecting that there was no distinguished research interest between 1983 and 1999. Most of the prototypes, designed in a large-scale (such as, frame size was 6m×6m×6m), have been implemented to the arc welding works during this stage. These welding robot mechanical configurations were presented as the gantry platform (frame systems), which consists of welding torch, frame system, and weld line as shown in **Figure 2.3** (Yagishita et al., 1983). Guided by the electrode weld lines, the robots could move along the three axes, X, Y, and Z. The equipped vision, arc weaving and touch sensors along with the point-to-point (PTP) control assist in automatically identifying, locating, and tracking welding seams. Developed by Apple Computer Inc., 1987, HyperCard programming was the primary programming language used during this stage.

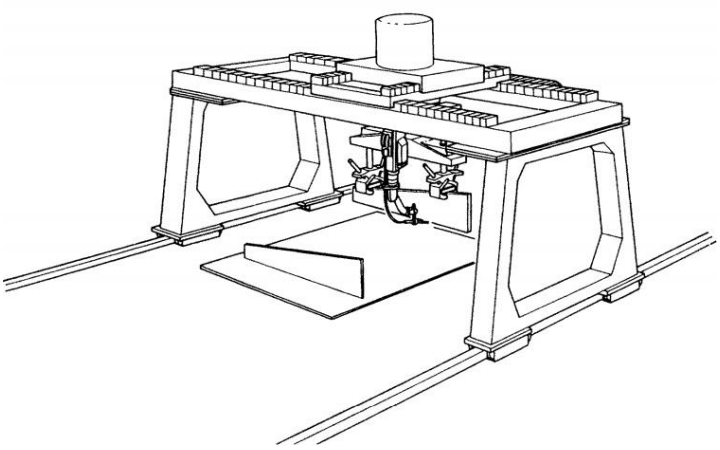
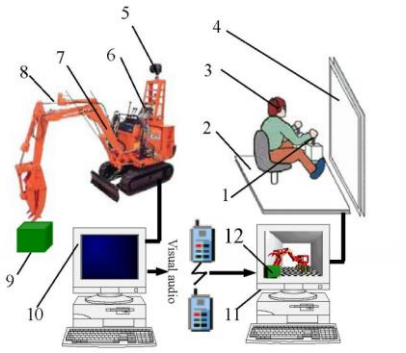


Figure 2. 3 Example of large-scale welding robot in 1980s (Yagishita et al., 1983)

During Stage II, the number of construction robotics-related studies increased from 99 to 112. During this stage, the research topics of robotic hardware design attracted more attention than the controlling methods with 61 papers focused on the robotic mechanical engineering topic and 47 papers focused on the robotic controlling topic. To protect human workers from hazardous working environment (such as, disaster restoration and rescue works) and to ensure worker safety (Kawashima et al., 2004, Sasaki et al., 2004), the proposed robotic systems were mostly used for "remote" construction. The main functionalities include grasping heavy objects (such as rocks, concrete blocks), excavation, and materials transportation. The 6-DOF robotic arms and humanoid robots, which were installed on the construction machinery, were the two typical mechanical provisions of the teleoperated robots (Feng et al., 2006, Cui et al., 2016). Most robot arms were pneumatically actuated for reasons of portability (Sasaki et al., 2004). Lightweight materials (such as, rubber) were used for the robotic arms. Some of the humanoid robots, covered by the protective clothing, could perform outside work under rainy and dusty conditions. Most teleoperated robots were controlled by the operator using joysticks, as shown in **Figure 2.4**

(Yokoi et al., 2003). More specifically, the operator would first use the joystick to input the target position signal, and further signals were then transmitted to the robotics arms through wireless LAN boards and a PC. During this period, sensor systems mainly included force, torque and pressure sensors. These sensors were mounted on the wrists of the robotic arms, and on each wrist and foot of the humanoid robots. The position sensors and DC servo motors were connected under the joysticks to estimate the reaction forces experienced by each arm cylinder. Inverse kinematics was applied to control robot motions in response to input sensor signals.

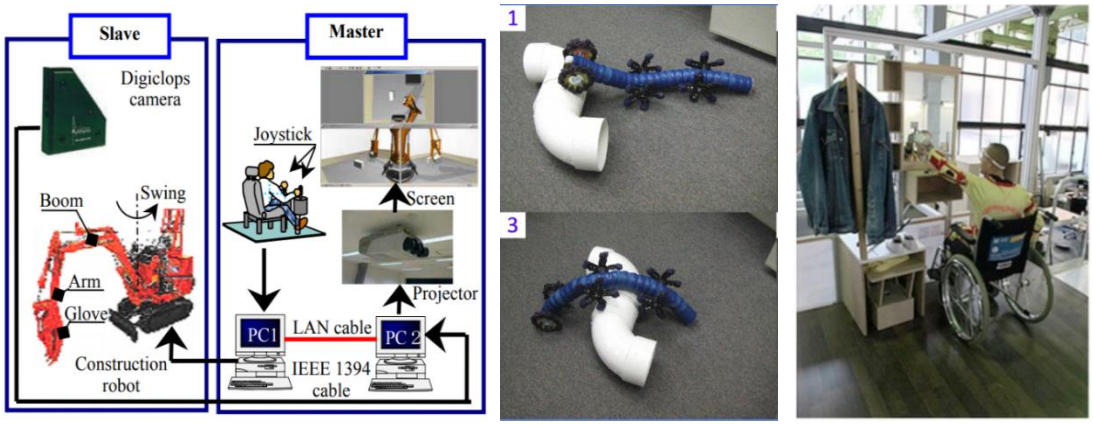


1-operating handle ; 2-6DOF motion base ; 3-earphone ; 4-3D scene ; 5-3-eye camera ; 6-accelerator sensor ; 7-force, torque, displacement sensors ; 8-hydraulic cylinder ; 9-object ;

Figure 2. 4 Example of using joysticks to tele-operate robot (Yokoi et al., 2003)

During Stage III, the number of construction robotics-related research increased from 112 to 139. Different to the former two stages, 52% of the research topics involved robotic controlling and 41% involved mechanical engineering. Robots for teleoperated construction still appeared the most frequently. The operators controlled the intelligent vehicles using a master-slave control system (Yamada et al., 2009). An example of the master-slave system is shown in **Figure 2.5**. The master system is composed of joysticks, mobile base (mounted in remote operation room), control box, and a vision screen. The slave system is composed of robotic

construction machinery, and related sensors. The teleoperated robots were improved by the receipt of more adequate site information (such as operation path, obstacles), the valid data volume was extended, making the robots more accurate. Meanwhile, the virtual reality technology was employed most to build and display virtual models of remote construction space (Tang et al., 2010). With the help of 3D images, captured by the stereo cameras, and the VR (virtual reality) techniques, the remote environment can be accurately represented, and the operators can identify the position of objects and avoid obstacles more accurately and efficiently. Notably, worm-like robots and intelligent furniture system also appeared frequently, as shown in **Figure 2.6** (Siles. et al., 2009, Georgoulas et al., 2014). Worm-like robots were developed to undertake inspection works (such as, pipes inspection). The intelligent furniture systems, consisting of multi-automated furniture and mobile robots, were mostly used to assist the elderly and disabled people.

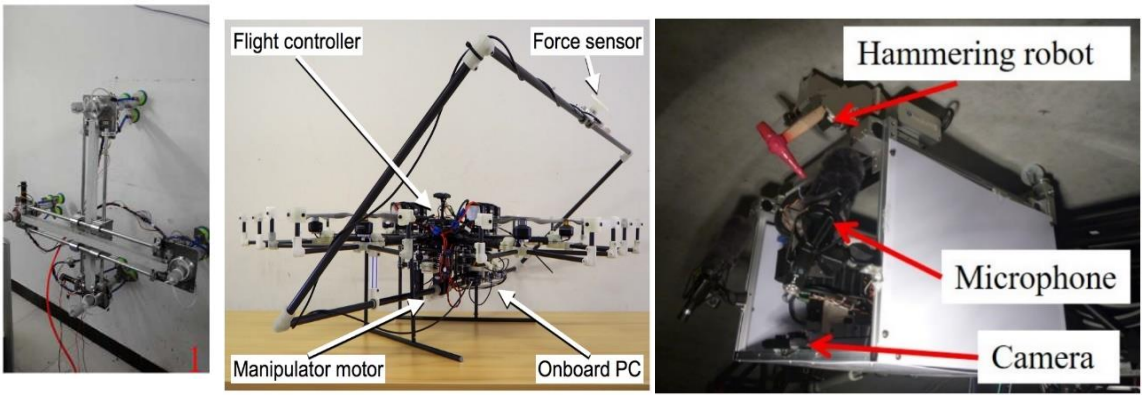


Master-slave system | Worm-like robot | Intelligent furniture

Figure 2. 5 Example of master-slave system, worm-like robot and the intelligent furniture (Yamada et al., 2009)

During Stage IV, the further development of construction robots has emerged as a very strong activity. The number of construction robotics-related studies increased sharply from 139 to 229, and the growth rate is almost 2.7 times that of the preceding stage. 62.4% of studies focussed on robotic controlling technology. The building

inspection robots at the stage have become the mainstream. To prevent buildings' deterioration, the functions of the inspection robots include detecting and reporting spalling, non-evenness, and inclinations, especially the cracks. Compared with human workers, inspection robots can work efficiently, making fewer errors. The Unmanned Aerial Vehicles (UAV) are used to inspect the slabs or the external building walls (Ikeda et al., 2017). The omni-directional wheel mobile and climbing robots have been well developed to search uneven building surfaces. Most climbing robots move with vacuum cups (Cui et al., 2016), shown in **Figure 2.6**. To imitate a worker conducting a concrete quality test using a test hammer, robotic arms were equipped with UAVs and climbing robots execute the hammer actions (Ikeda et al., 2017, Takahashi et al., 2018), shown in **Figure 2.6.** During Stage IV, the robots have often been equipped with lidar scanner and camera sensor systems. Lidar scanners generate 3D point data and cameras generate colour data for attaching to point clouds (Bolourian et al., 2020). These point data provide more accurate location information in foggy and low light conditions. Machine learning algorithms such as the deep reinforcement learning method, have been used to compute the 3D point data. Crucially, mobile robots for special-shaped masonry work and 3D printing robots also appeared frequently and related studies have increased rapidly (Hoffmann et al., 2020).



Climbing robot with vacuum cups

UAVs with manipulator

Climbing robot with hummer

Figure 2. 6 Example of climbing robots, manipulator – UAVs, and hummer climbing robot (Ikeda et al., 2017)

In summary, construction robotics research has been much investment from the 1980s to the 2020s. Research endeavours focused on robotic mechanical "architecture" initially but is now more preoccupied with exploring controlling algorithms.

The mechanical design of construction robots, from the 1980s to the 2000s were of large size and fixed in one place. They were difficult to relocate and reuse in different site locations. Therefore, from the 2000s to 2010s, movable construction robots were developed. In this period, the self-controlled vehicle and robot-controlled vehicle became mainstream activity. These robots undertake such tasks on site as excavation and the grasping of heavy objects. Robot-controlled vehicles were designed as humanoid robots and robotic arms were mounted on the vehicle operation room. The humanoid robots and robotic arms can be easily disassembled and reassembled on different sites. These robots can also be controlled by joysticks and remote operators, thus the better to avoid injuries. Between 2010 and 2020, robots became smaller and more diverse in appearance. The mobile emerged with a wheeled mobile robot and UAVs became popular. These small mobile robots are

fully automated and can be used in both outdoor and indoor construction environments. Meanwhile, cable-driven parallel robots were developed for printing large-scale building components.

In the early stages of developing sensory systems and control methods for robots, the feedback information was mainly given using touch, force, pressure, and position sensor. Currently, the smart robots are also equipped with vision sensors (such as stereo cameras), laser scanners, and lidars. These updated sensory systems allow the robot to sense the surrounding environment, in order to perform operations more accurately and efficiently.

In the early stages, industrial robot control methods such as PTP (point-to-point) was commonly used. However, these methods were less efficient for movable robots and larger datasets. Recently, AI (artificial intelligence) techniques have been introduced to learn from big point-cloud data and image data. The machine learning based algorithms, such as deep reinforcement learning algorithm, are well used for dynamically controlling robots in unfamiliar environments. With sufficient valid information and intelligent control algorithms, construction robots operate more smoothly and dynamically in a complex site environment.

As for the technology applications, arc welding robots were well researched during Stage I. After that, remote construction robots were well researched in Stage II and Stage III. These VR-based, master and slave controlled teleoperated robots became relatively mature in Stage III. Nowadays, inspection robots research is much studied in relation to building crack inspection. Therefore, it is worth to studying the robotic control system for building crack inspection to meet the targeted research trend and

demands. A wheeled mobile robot, equipped with lidar and a camera, was employed to test the developed control system. Meanwhile, the developed robotic control system is expected to be freely coded in diverse robotic platforms.

Chapter3: Research Outline

This chapter presents an overview of the research body. In this research, a robotic control system was developed for fully automated building crack inspection.

According to the definition, the robotic control system is developed for controlling the robot's movement and realizing the expected functions of the robot. To develop the robotic control system for building crack inspection, three fundamental functions need to be achieved: 1) Capturing the photographs and inspecting the cracks. 2) Autonomous navigation in building environments. 3) Visualizing the inspection outcomes for the users. Therefore, this research put emphasis on developing a computer vision-based automated inspection algorithm, a local autonomous navigation strategy, and a web-user interface to activate the robotic vision, navigation, and visualization, respectively.

Specifically, the first step (Chapter 4) explored the development trend of construction inspection robotics to target the supporting technologies for the development of the robotic control system. The development trend was identified by analyzing the fruitful linguistic topics related to construction inspection robotics. The linguistic topics were extracted and generated using the natural language toolkit and the topic modeling technique. The needed technologies were then determined by summarizing the advantages, disadvantages, prevalence, and trend of the linguistic topics.

Guided by the supporting technologies, the second step (Chapter 5) is to design a lightweight CNN architecture that can be successfully implemented on diverse robotic platforms, including those powered by CPU processors. The network depth

was designed based on the very deep CNN architecture, VGG-16, because deeper CNN yields more accurate prediction. The network weight was decreased by inserting several 4×4 convolution filters in the convolution process. The performance of the designed lightweight CNN was evaluated using the confusion matrix and highlighted by comparison with the state-of-the-art CNNs. An on-site validation was conducted to test the feasibility of the developed lightweight CNN.

The third step (Chapter 6) is to develop an autonomous navigation method for controlling robot's movements. To save the computation cost of map construction, the local navigation strategy, a fuzzy logic controller (FLC) enabled wall-following algorithm, was designed. Navigated by the designed FLC, the robot is capable of continuously following the building components to conduct detailed crack scanning as well as avoiding forehead obstacles. The developed navigation strategy was validated in both simulation and on-site environments.

The last step (Chapter 7) is to develop a web-user interface to demonstrate the inspection outcomes from the robot processor to the users' screen. With the help of the web-user interface, smooth, continuous, and real-time visualization can be ensured. The web-user interface was developed using the HTML framework and formatted using the CSS language. JavaScript plugins were coded accordingly to realize the functions of: 1) capturing the video stream, 2) inspecting cracks with the developed lightweight CNN, and 3) saving the inspection videos. To provide a clear interpretation of the research works, the research outline diagram is shown in **Figure 3.1**.

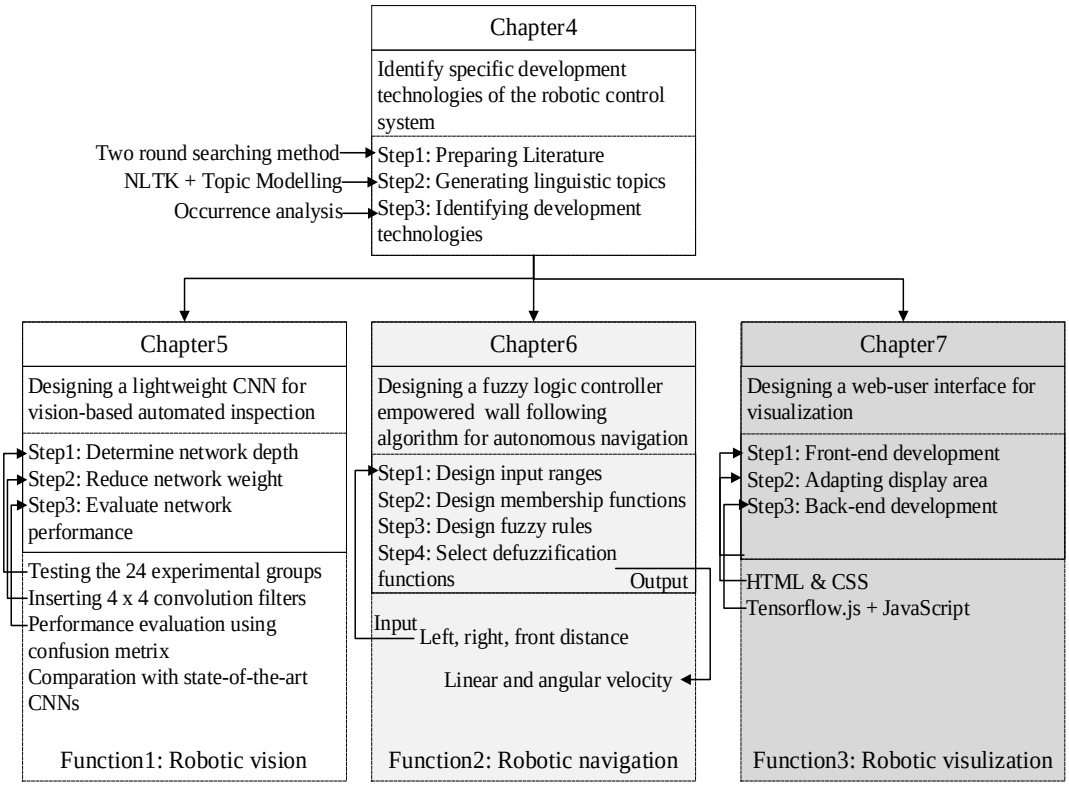


Figure 3. 1 Research outline

Chapter4: Explore the Development Trend of Construction Inspection Robotic

4.1 Introduction

The supporting technologies for developing the robotic control system were identified in this section. The robotic control system is expected to enable the mobility and operation of the robotic functions, including video stream capture, crack inspection, autonomous navigation, and inspection results visualization. To target the supporting technologies for the expected functions, the development trend of construction inspection robotics was explored by automatically extracting, classifying, and interpreting the historical technological topics, as well as their benefits, drawbacks, and prevalence. Topic modeling and a natural language toolkit were employed as a semi-automated analytical approach to gather the technological topics in a thorough and objective manner.

Literature-based scientific knowledge has been widely extracted and used as supporting data to identify technological breakthroughs (Gharbia et al., 2020, Gharbia et al., 2019, Cai et al., 2019). Therefore, scientific literature, regarded as the root for technological possibilities, creations, designs, and assessment criteria (Brooks, 1994), was used to extract the technological topics. To obtain a broad scope of knowledge, the identified topics cover the entire field of construction inspection robotics. Meanwhile, the supporting robotic technologies for the construction inspection robots also applicable to the building crack inspection robots since building crack inspection is a subcategory of construction inspection.

To be specific, there are three main processes conducted in this chapter. First,

research studies that are related to construction inspection robotics were prepared using a two-round searching strategy. The selected papers were then preprocessed using the Natural Language Toolkit (NLTK) to retain essential linguistic information of each paper. Next, the cleaned papers were input to topic modeling to automatically generate and group technological topics and their representative words. Finally, the generated technological topics were interpreted and categorized by referring to the meanings of representative words. The advantages, disadvantages of each topic were provided. The research emphasis over the period and future trends were also shared. Research methods, contents, outcomes, and links of each process were simplified in **Figure 4.1**. Details of each process are shown in sections 4.2 to 4.4, respectively.

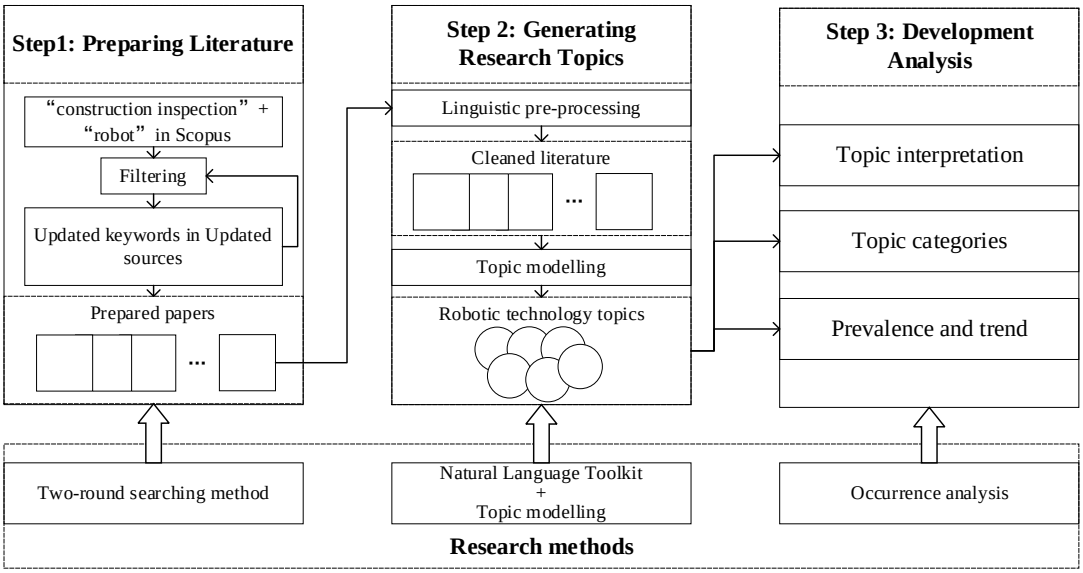


Figure 4. 1 Research framework

4.2 Data collection

In the literature collection stage, a two-round searching method was proposed and employed to identify literature related to construction inspection robotics as complete as possible.

4.2.1 Literature searching in the first round

In the first searching round, the search keywords were subjectively determined based on their relevance to construction inspection robotics research. The search sources are the most general literature databases: PubMed, Google Scholar, Scopus, and Web of Science (WoS) (Li et al., 2010). Among them, Scopus were selected because: Scopus index the articles from natural sciences and engineering fields (Mongeon and Paul Hus, 2016) and Scopus indexes the largest number of journals than the other three databases (Falagas et al., 2008).

The keywords "robot" and "construction inspection" were used as the search keywords in the first searching round. These search keywords were selected subjectively based on their high occurrence in relevant industry reports and previous studies. The searching field was limited to "engineering" to efficiently target robotic technologies employed in construction inspection applications. Literature types, such as review, conference review, book, were excluded because they mainly present summaries or insights instead of specific and detailed information. Finally, 351 papers were obtained in the first round after using the searching query: "(TITLE-ABS-KEY (robot) AND TITLE-ABS-KEY (construction inspection)) AND (LIMIT-TO (SUBJAREA, "ENGI")) AND (LIMIT-TO (DOCTYPE, "cp") OR LIMIT-TO (DOCTYPE, "ar") OR LIMIT-TO (DOCTYPE, "ch"))".

The obtained papers were filtered by intensively reading their titles and abstracts because the main topics are more likely to appear in these two sections. The filter criteria are: 1) research that is not relevant to construction applications was excluded. For example, a mobile robotic system developed for fruit-harvesting (Zhou et al.,

2015), 2) research that was not focused on developing robotic technologies were excluded. For example, (Xiang et al., 2020) discussed challenges and breakthroughs in tunnel construction instead of introducing robotic technology. Doing so, 185 papers that only focused on developing or implementing robotic technologies for construction inspection remained.

4.2.2 Literature searching in the second round

To find hidden research works, both the search keywords and sources were updated in the second searching round. The updated keywords were obtained after analyzing the co-occurrence keywords mapping. Research keywords with high co-occurrence are considered because they have higher importance weights in discovering meaningful knowledge (Radhakrishnan et al., 2017). The co-occurrence mapping was obtained using VOSviewer. VOSviewer is a commonly used software to perform the bibliometric information of scientific publications (Yu et al., 2020). With the help of inserted natural language processing algorithms, the keywords in each article can be extracted automatically in VOSviewer. In that way, hidden keywords that are related to construction inspection robotics can be completely identified.

To obtain sufficient updated keywords, the minimum number of co-occurrences in VOSviewer was set at 4. The keywords, such as, “construction industry” were neglected because such terms are too general, which may lead to excessive and invalid searching. **Figure 4.2** presents the co-occurrence map generated by processing the 185 papers in VOSviewer. It can be observed that except for the used keywords of “inspection robots”, “construction”, hidden keywords of “mobile

robots", "machine design", "concrete construction" demonstrated a relatively high co-occurrence in papers related to construction inspection robotics. Therefore, the search keywords, updated as "mobile robot and inspection", "machine design and inspection robot", "concrete construction and inspection robot", were searched and filtered again to identify hidden literature.

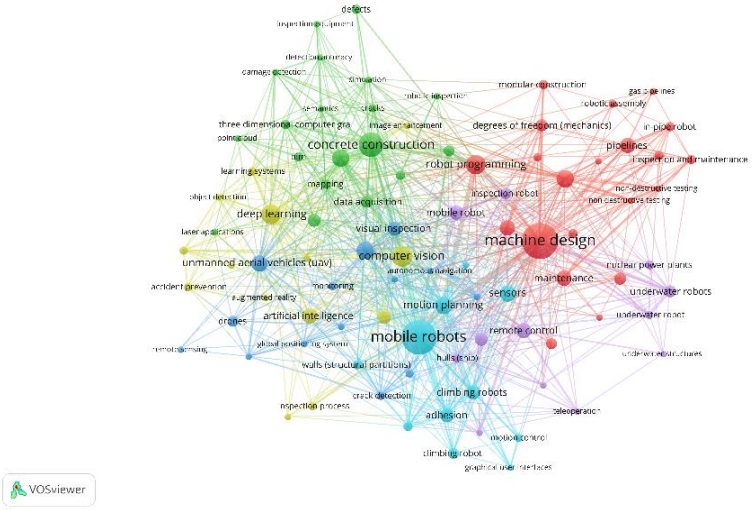


Figure 4. 2 Keywords co-occurrence map generated from VOSviewer – Round1

When searching the updated keywords in Scopus, a substantial number of unrelated papers were provided. To efficiently target the relevant papers, the publications that indexes the filtered papers were employed as the updated sources instead of Scopus. Specifically, the publications with high relevance were selected. The number of filtered papers that were indexed in the publications was used to rank their relevance. The journals or conferences indexing more filtered papers were chosen as the updated search sources. Narrowing the search area helps to keep the most relevant papers at a fast speed, which improves the efficiency of the filtering process.

Specifically, the 185 papers were indexed in 90 different publications. Among them, *International Symposium on Automation and Robotics in Construction*, *IEEE*

International Conference on Intelligent Robots and Systems, *Automation in Construction*, *Industrial Robot*, *Proceedings - IEEE International Conference on Robotics and Automation*, appeared nine, eight, six, and seven times, respectively, while most of the publications only appeared once. Therefore, these five publications were chosen as the updated sources.

When searching the updated keywords in the updated sources, 64 hidden papers were identified after filtering. **Figure 4.3** shows the keywords co-occurrence map in the second round. It can be observed that except for the keywords identified in the first and second round, "computer vision" appeared frequently, and the five publications mentioned above still showed high occurrence. Therefore, the new keywords were searched again in the five publications, which generated five more papers.

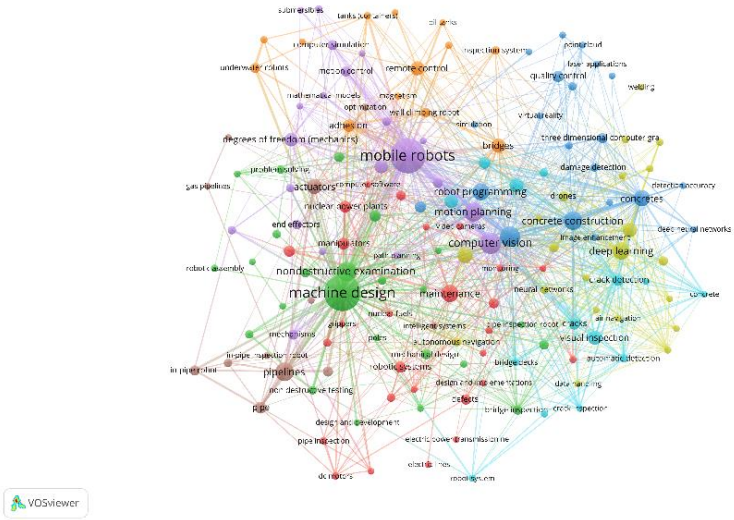


Figure 4. 3 Keywords co-occurrence map generated from VOSviewer – Round2

In summary, 254 (185+64+5) papers that related to construction inspection robotics were prepared for further analysis. When processing the 254 papers in VOSviewer again, no more new keywords that have high occurrence were presented, which

ensures the integrity of the prepared literature.

4.2.3 Topic generation

The Latent Dirichlet Allocation (LDA), the most commonly used TM algorithm (Blei, Ng, and Jordan 2003), was employed to automatically generate, group the research topics from the 254 papers as well as the representative words for each topic.

LDA is considered as a generative statistical model that contributes to clustering a set of documents with similar features. Normally, the features of textual data can be recognized by their representative topics and keywords. In line with this idea, LDA assumes that the corpus is characterized by a mixture of latent topics while the topics are generated by their representing words. In the topic generating process, the documents (literature dataset) and the linguistic words (preprocessing literature using NLTK) are first obtained as the known parameters. The initial latent topics are then assigned randomly by the system. After training the distributions of words to topics and the distributions of topics to documents within several epochs, the determined topic can be obtained when the distributions tend to converge.

LDA outputs the probabilities of the words that represent a generated topic and the topics that represent a document in the whole corpus. The meanings of the generated topics are manually labeled, referring to their representative words with the highest occurrence probability.

Specifically, D represents the entire literature set, $D = \{d_1, d_2, d_3, \cdots, d_{254}\}$, d represents each selected paper in the literature set. W represents the set of

different words in set D, $W = \{w_1, w_2, w_3, \cdots, w_i\}$, W represents each word that consist of paper d. T represents the set of generated topics, $T = \{t_1, t_2, t_3, \cdots, t_j\}$. Equations 1 and 2 can be used to compute the probability of each topic to each literature, as well as the probability of each word to each topic, using the values of these parameters.

$$p_{t_j} = \frac{n_w, w \in (d \cap T)}{N_w, w \in d} \quad (1)$$

Here, p_{t_j} is the probability of each topic to each literature, n_w is the number of common word of literature d and topic t, N_w is the total number of words in literature d

$$p_{w_i} = \frac{f_{w_i}}{N_{w_i}} \quad (2)$$

Here, p_{w_i} is the probability of each word to each topic, N_{w_i} is the total number of words in topic t_j, literature d and topic t, f_{w_i} is the occurrence of word w_i in topic t_j appears in the total word set W.

ϑ_d and η_t represent the sets of p_{t_j} and p_{w_i}, respectively. Both ϑ_d and η_t are multinomial distributions, obeying the Dirichlet distribution (Fabius, 1973). The Dirichlet distribution is expressed in equation 3, where α is the hyperparameter that determines the distributions of the independent variables x_i. The probability of topic to literature, or word to topic, is represented by x_i in this research, and the hyperparameters that influence the distribution of these probabilities are represented

by α and β .

$$f(x_1, \ldots, x_k; \alpha_1, \ldots, \alpha_k) = \frac{1}{B(\alpha)} \prod_{i=1}^{K} x_i^{\alpha_i - 1}$$
$$B(\alpha) = \frac{\prod_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)} \tag{3}$$

Equation 4 can be used to express the connection between $P(W|D)$, $P(T|D)$, and $P(W|T)$. During the training process, $P(W|D)$ is a given condition, and each literature can be distributed with the relevant topic by continuously adjusting $P(T|D)$ and $P(W|T)$, in other words, α and β using Gibbs sampling (Alhamazawi and Yu, 2012). The meanings of the topics are labeled based on their representative words. When Gibbs sampling converges, the optimal α and β are determined.

$$P(W|D) = \sum_{T} P(T|D) P(W|T) \tag{4}$$

Here, $P(W|D)$ is the probability of word to literature, $P(T|D)$ is the probability of topic to literature p_{t_j} , $P(W|T)$ is the probability of word to topic p_{w_i} .

It is essential to determine the best topic number before training the LDA model. If the initial number of topics is inappropriate, the LDA model may become too coarse to yield reliable classifications (Mazzei et al., 2021). The papers selected cannot be summarized completely if the number of topics is too small. Some of the important ideas will most likely be ignored. The generated topics will be too fragmented if the number of topics is too large. It is possible to come up with topics that have less

relevance. In this research, the two most commonly used methods, the Perplexity score and the Coherence score, were used to determine the topic number. Perplexity is a type of log-likelihood function used to demonstrate how well a model fits the data. The perplexity score decreases gradually if the LDA model is successfully fitted (Spreeuwers, 2014). By measuring the degree of semantic similarity, topic coherence supports the exclusive investigation of topics (Syed and Spruit., 2017). With a higher coherence score, the generated topics are more specific. Therefore, the optimal topic number appears when the perplexity score is decreasing, and the coherence score reaches its highest.

4.2.4 Data preprocessing

To ensure an accurate computation process of the LDA model, the data pre-processing is needed to remain only the essential contents and structure the input texts. In this research, data pre-processing was implemented in the following steps:

1) Remove empty space, punctuations, numbers. Because punctuation, such as "! @#$ percent," numerals, such as "12345," and empty space have no special meanings for topic interpretation, they were deleted first to enhance computation performance.

2) Retain only nouns, verbs. Normally, nouns and verbs contribute the most to understanding the meaning of phrases. Words like "automatically" (adverb) are used to describe nouns and verbs. Therefore, only nouns and verbs were retained to target first most information. Some nouns and verbs, such as "abstract, conclusion, summary, therefore, fig, figure, table, author" are irrelevant to the topic meanings, they were also removed.

3) Lemmatization and stemmer. Lemmatization and stemmer are important processes to normalize text words. To meet the grammatical requirements, the words in papers are presented as different forms. For example, ‘inspection’, ‘inspects’, ‘inspected’, “inspector”. Lemmatization contributes to normalize the inflectional forms and derivationally forms of words to dictionary form. For example, after lemmatization, both the word ‘inspects’ and ‘inspected’ is converted into same form ‘inspect’ for the interpretation of computers. Stemmer is used to convert inflected words to their word base. For example, “cats”, “catlike”, and “catty” are normalized as “cat” after stemmer. Doing so, the same words that presented with different forms can be correctly recognized and input to the LDA model. WordNetLemmatizer from the NLTK library in the Python programming language was used to implement lemmatization process.

4) Lowercase. Because computer considers the same words that presented with uppercase and lowercase as two different variables. For example, computer interprets ‘Robotics’ and ‘robotics’ as different words although they contain the same information. Therefore, to reduce the distribution error of words and topics, the whole contents that processed after the first two steps were lowercased.

5) Remove stopwords. Stopwords like we, you, are, is, am, there is unnecessary for LDA processing. Therefore, stopwords were removed accordingly using stopwords library from NLTK in Python.

All the papers were preprocessed following the above five steps. After preprocessing, the clean texts were used for the LDA computation. **Figure 4.4** demonstrates an example of the preprocessed text.

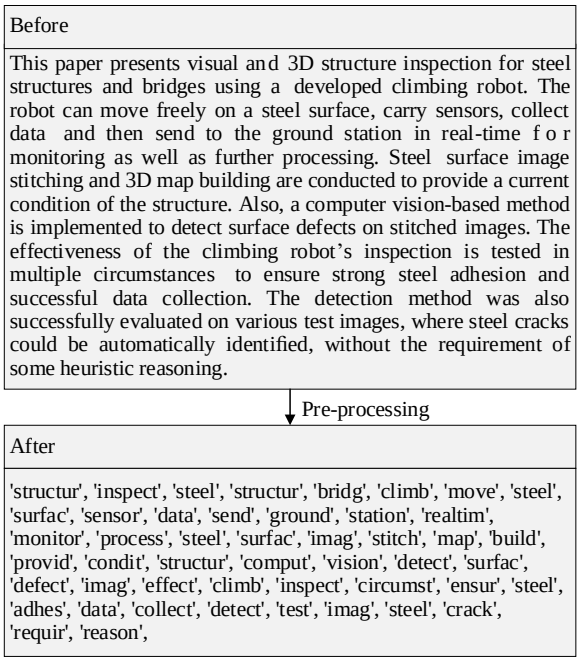


Figure 4. 4 Example of text pre-processing

4.3 Identified Topics

4.3.1 Number of topics

As described in section 3.2, both the perplexity score and the coherence score were referenced to estimate the appropriate number of topics. **Figure 4.5** shows the variation of perplexity and coherence of the LDA model when the number of topics changes from 1 to 20. The number of topics was set in a range of 1 to 20, referencing the findings of (Park, Hong and Kim, 2017, Yang, Chang and Choi, 2018, Kim and Rhee, 2016).

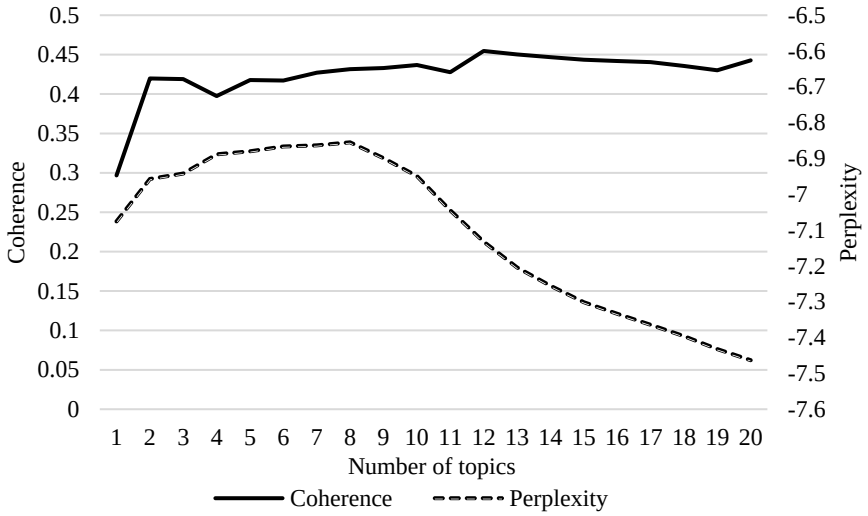


Figure 4. 5 Variation of perplexity and coherence

In **Figure 4.5**, x-axis represents the number of topics, while the y-axis represents the perplexity and coherence scores. When the number of topics rose from one to eight, the perplexity score increased consistently from -7.07 to -6.85. Upon that eighth topic, the elbow appeared, and the perplexity score started to decrease from -6.85 to -7.46, which indicates that the LDA model began to fit when more than eight topics were generated. Meanwhile, the highest coherence score occurred when the number of topics reached twelve. Therefore, the optimal number of topics was determined as twelve.

4.3.2 Topics and their interpretation

The generated twelve topics were listed in **Table 4.1**. The topics were automatically generated based on the principles of the LDA model introduced in section 3.2 and coded using the Python language and NLTM and Genism package. To interpret the topics, the top 15 words that are associated with each topic were generated and examined. The occurrence probability of each word was also obtained. The higher the probability, the more relevant they are to the topic.

Table 4. 1 Generated research topics and their representative words

Topics	Representative words and occurrence probability
Topic0	0.143*"frame" + 0.036*"transduc" + 0.031*"camera" + 0.028*"construct" + 0.027*"calibr" + 0.025*"build" + 0.024*"marker" + 0.020*"base" + 0.018*"corridor" + 0.017*"refer" + 0.010*"sequenc" + 0.010*"video" + 0.009*"registr" + 0.009*"panorama" + 0.008*"site"
Topic1	0.039*"pipe" + 0.028*"defect" + 0.017*"inspect" + 0.017*"detect" + 0.017*"model" + 0.016*"pipelin" + 0.016*"imag" + 0.014*"system" + 0.013*"condit" + 0.013*"process" + 0.012*"water" + 0.009*"sewer" + 0.009*"construct" + 0.008*"studi" + 0.008*"techniqu"
Topic2	0.017*"control" + 0.016*"point" + 0.016*"obstacl" + 0.014*"model" + 0.014*"motion" + 0.014*"posit" + 0.012*"track" + 0.011*"shape" + 0.011*"environ" + 0.011*"sensor" + 0.011*"navig" + 0.010*"forc" + 0.010*"time" + 0.009*"steel" + 0.008*"path"
Topic3	0.065*"hammer" + 0.052*"sound" + 0.036*"test" + 0.034*"inspect" + 0.030*"frequenc" + 0.027*"delamin" + 0.022*"tile" + 0.020*"concret" + 0.017*"devic" + 0.015*"surfac" + 0.014*"nois" + 0.014*"structur" + 0.013*"sampl" + 0.013*"signal" + 0.012*"defect"
Topic4	0.033*"control" + 0.031*"system" + 0.011*"oper" + 0.010*"posit" + 0.010*"modul" + 0.008*"model" + 0.008*"joint" + 0.008*"forc" + 0.008*"design" + 0.007*"actuat" + 0.007*"structur" + 0.007*"time" + 0.007*"simul" + 0.007*"motion" + 0.006*"power"
Topic5	0.035*"point" + 0.023*"imag" + 0.020*"camera" + 0.017*"model" + 0.015*"valu" + 0.015*"line" + 0.014*"algorithm" + 0.012*"system" + 0.011*"distanc" + 0.011*"environ" + 0.011*"error" + 0.010*"featur" + 0.010*"local" + 0.010*"measur" + 0.009*"sensor"
Topic6	0.149*"wheel" + 0.077*"forc" + 0.031*"friction" + 0.030*"torqu" + 0.022*"corner" + 0.016*"adhes" + 0.015*"valu" + 0.015*"front" + 0.014*"servo" + 0.011*"rubber" + 0.010*"tilt" + 0.010*"calcul" + 0.010*"traction" + 0.010*"locomot" + 0.009*"rear"'
Topic7	0.020*"system" + 0.015*"inspect" + 0.014*"vehicl" + 0.013*"environ" + 0.013*"plan" + 0.013*"camera" + 0.012*"path" + 0.011*"oper" + 0.010*"test" + 0.010*"process" + 0.009*"construct" + 0.009*"strategi" + 0.008*"area" + 0.008*"devic" + 0.008*"measur"
Topic8	0.030*"surfac" + 0.028*"inspect" + 0.028*"climb" + 0.019*"system" + 0.016*"forc" + 0.016*"wall" + 0.012*"control" + 0.012*"design" + 0.012*"adhes" + 0.012*"sensor" + 0.011*"pressur" + 0.011*"structur" + 0.010*"steel" + 0.010*"test" + 0.007*"mechan"
Topic9	0.113*"pipe" + 0.055*"pipelin" + 0.054*"drive" + 0.044*"mechan" + 0.043*"wheel" + 0.039*"inspect" + 0.026*"diamet" + 0.021*"forc" + 0.020*"motor" + 0.018*"unit" + 0.016*"modul" + 0.015*"steer" + 0.014*"vehicl" + 0.014*"angl" + 0.008*"cabl"
Topic10	0.062*"imag" + 0.038*"detect" + 0.038*"crack" + 0.019*"featur" + 0.018*"network" + 0.013*"train" + 0.012*"learn" + 0.011*"layer" + 0.011*"surfac" + 0.011*"system" + 0.010*"process" + 0.009*"perform" + 0.009*"model" + 0.008*"pixel" + 0.008*"accuraci"
Topic11	0.050*"inspect" + 0.037*"bridg" + 0.029*"system" + 0.020*"tunnel" + 0.019*"crack" + 0.014*"imag" + 0.013*"point" + 0.013*"detect" + 0.011*"measur" + 0.011*"camera" + 0.011*"sensor" + 0.009*"posit" + 0.008*"cloud" + 0.008*"structur" + 0.007*"model"

The twelve topics were then manually interpreted based on the meanings of their representative words with high probability. Generally, the existing research for construction inspection robotics can be divided into three categories: studies into robotic mechanism designs, robotic inspection techniques, and robotic navigation techniques. Categories and subcategories of the research topics are shown in **Figure 4.6.**

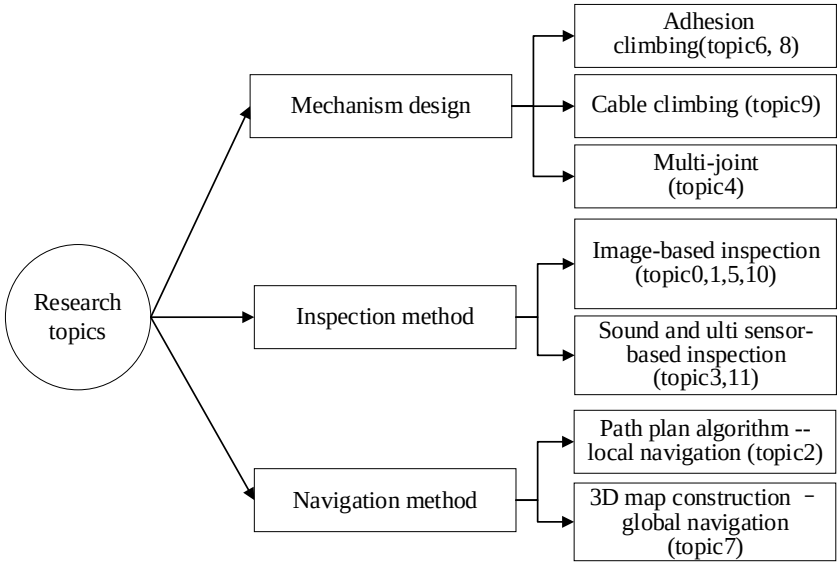


Figure 4. 6 Categories of construction inspection robotics research topics

To target the development technologies for robotics control system, the topics related to "inspection method" (topic0,1,3,5,10,11) and "navigation method" (topic2 and 7) that contribute to controlling the robot to realize expected functions and movements of the robot were interpreted in a detailed manner.

(1) Image-based robotic inspection method

Referring to the representative words, such as "imag", "calibr", "camera", "feature", studies of topic0, and topic5 introduce employing image processing for robotic inspection. Referring to representative words such as "imag", "feature", "network",

“train”, studies of topic1 and topic0 focused on inspections using machine learning method. Both image processing and machine learning inspections demonstrate quality defects to users by processing photographs captured by cameras mounted on robotic systems. To be specific, the inputs are images or videos, and the outputs are image characteristics, such as object segmentations or classifications. To achieve this, the input images are first converted to a numerical pixel matrix for computer interpretation. Steps for processing pixel matrices in image processing and machine learning are different.

Fundamental steps of image processing include: 1) image enhancement and restoration, such as grayscale, deblurring, noise removal, and illumination correction, which adjusts input images for efficient computing. 2) Wavelet transformation, which is used to separate data from an image. For example, wall cracks and backgrounds can be separated through wavelet transformation. 3) Image segmentation, which divides and labels an image into different segments. 4) Morphological, which removes imperfections in the image segments. 5) Representation and description, which outputs the image segments in raw pixel data, such as the boundary of a crack region.

The most commonly employed approach for machine learning-based robotic inspection is convolutional neural networks (CNN). Convolutional, pooling, and fully connected layers are the three basic layers of CNN. A convolutional layer is used to extract image features. By sliding the entire picture with an s-curve, filters, n×n matrix, are used to generate feature maps of an image. The effects of extraction depend on the number and size of filters. A feature map is typically 3×3 in size, and

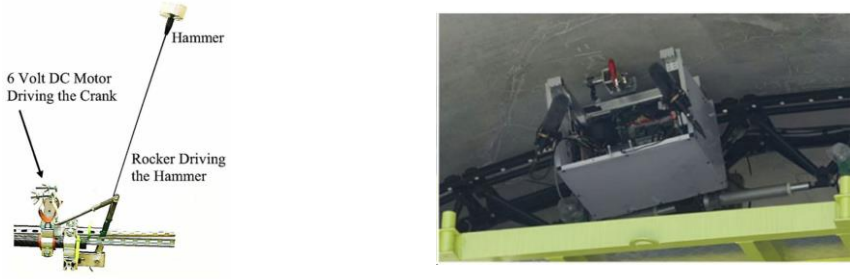
more feature maps result in more detailed object identification. By lowering the dimension of feature maps, pooling layers aid in saving computing costs. Finally, feature maps are identified using a fully connected layer trained by comparing the output labels to the actual labels. Deep learning networks are employed in the fully connected process. Reporting object classifications, such as presenting a label of crack, is the essential function of CNN. In spite of showing labels, an increasing number of improved networks also provide locations, or even sizes, regions of defects.

Both image processing and machine learning can present defects in an image. Because cracks appear most frequently and are difficult to identify, these two methods are mainly used to inspect cracks on building surfaces. Machine learning techniques attract more attention compared with image processing because: 1) The machine learning approach identifies defects through training rather than programming, making the process faster and simpler. 2) CNN models can be used for many different tasks because they are easy to retrain with new datasets.

(2) Sound and multi-sensor based robotic inspection method

According to the representative words, such as “hammer”, “sound”, “test”, “point”, “cloud”, studies of topic3 and topic11 focused on developing robotic inspection methods by processing information from hammers and multi-sensors. With the fast development of nondestructive testing (NDT), robotic inspection technologies are no longer limited to inspecting visible defects. As guided by NDT, by observing signal variations, such as hammer sounds, the shape of an electronic wavelet, vibration frequency, invisible defects, such as concrete delamination, or crack depth,

can be identified. For example, when striking concrete surfaces with hammers, the undamaged concrete reflects clear sound while delamination or void concrete reflects hollow or muted sound (Watanabe, et al., 2015). Therefore, in spite of integrating cameras on robotic systems to inspect visible defects, today's construction inspection robots have begun to equip multi-sensors, including hammers, lasers, electrical resistivity, impact-echo, and ultrasonic surface waves sensors to test concrete inner structures (**Figure 4.7**).



Moreu et al. 2018 Kasahara et al. 2018

Figure 4. 7 Example of the hammer robotics

(3) Research of robotic navigation techniques

Referring to the representative words, such as "mode", "obstacle", "motion", "posit", "path", studies of topic2 looked into path planning research for controlling robot motions. Referring to the representative words, such as "path", "construct", "environ", "area", studies of topic7 focused on constructing environmental maps for generating robot navigation paths. These two topics can be considered the branches of robotic navigation techniques. Most of the navigation strategies are used to control unmanned aerial vehicles (UAV) and autonomous vehicles to travel autonomously and safely in building environments such as indoor or outdoor buildings, pipelines, pavements, and bridge surfaces.

Robotics navigation can be divided to global and local navigation. Global

navigation (topic7), also known as off-line navigation, requires maps that are given beforehand and incorporate environmental knowledge. The most developed and extensively used method for robotic global navigation is simultaneous localization and mapping (SLAM). However, because it only offers 2D maps, the conventional SLAM method is insufficient for construction inspection applications. For instance, while navigating a UAV to inspect a high-rise structure, 3D location data is necessary. Therefore, an increasing number of research are beginning to focus on the construction of 3D environmental maps for construction inspection robot navigation. Extracting geometry information from building information modeling (BIM) is considered an efficient way to obtain position data of building components. To be specific, a precise BIM model needs to be provided first and then transferred to robotic control systems such as Robot Operating System (ROS) and Unity. Although integrating BIM for autonomous navigation is a feasible option, there are several limitations: 1) The majority of research manually extract geometry data from BIM models. Software that uses BIM models to directly control robots is scarce. 2) It is time-consuming to build environmental maps of complex environments. 3) When it comes to changing building scenarios, it is ineffective.

For local navigation (topic2), the robots make path plans in real-time by receiving and computing position information from external sensors, such as GPS, lidar, or laser. Mathematical models, such as proportional–integral–derivative (PID) control and fuzzy logic controller, are widely used to compute the input location data and output angular and velocity commands. Compared with global navigation, local navigation is more suitable for complex and dynamic environments. The majority of local navigation algorithms, however, rely on the precision of location sensors.

Once the sensors are unable to receive information from the surroundings, the robot may possibly lose control.

The advantages, disadvantages, and suitable application areas of each developed robotic technology are summarized below.

Table 4. 2 Summary of development technologies for robotic control system

Robotic technologies			ADV	DA	Suitable application areas
Inspection technology	Image -based	Image processing	-- Accurate	Time consuming	Surface cracks (including surfaces of building, bridge, tunnel, pavement, pipeline)
		Machine learning	--Faster --simpler	Costly	
	Sound and multi-sensor based		Detect invisible defects, such as concrete delaminat ion, crack depth	--Heavy --Costly	Concrete inner structures (including building, bridge, tunnel)
Navigation technology	Global navigation		Accurate	Not suitable for complex and changing environment	Control UAV and autonomous vehicles in indoors, outdoors, pipelines, pavement, bridge surface.
	Local navigation		Dynamic	Sensitive to sensor inputs	

4.3.3 Trends of topics

Around 1989, robotic mechanism was first developed for construction inspection activities. The first inspection robot was a multi-joint snake-like robot develop for pipeline inspection (Journa et al., 1989). Following that, research interest has begun

to increase in the integration of robotics—either robotics mechanisms or robotics control algorithms—into construction inspection activities. The publication trend of construction inspection robotics innovations from 1989 to 2021 is depicted in **Figure 4.8.**

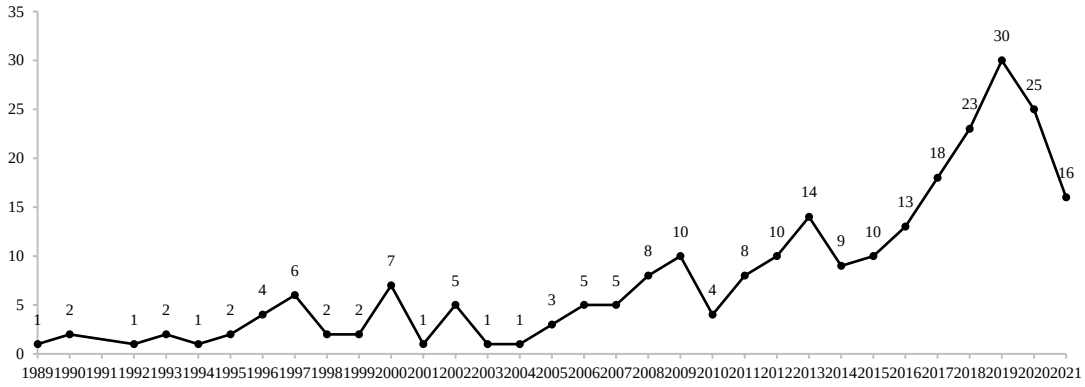


Figure 4. 8 Trend of construction inspection robotics-related research

It can be observed that the trend of construction inspection robotics research shows an increase in general. Less than 10 published papers per year were centered on construction inspection robotics before 2013, indicating a relatively low level of research interest. At the beginning, a large number of laborers were attracted to the construction industry since it was regarded as the country's pillar industry and showed high demand. Therefore, there was a less need to invest in robotics into construction inspection at that time. The hiring of manual inspectors was satisfactory in terms of cost-effectiveness.

With a maximum of 30 papers published each year, construction inspection robotics related research began to rapidly expand from the year 2014 and peaked in 2019. Since 2014, there has been an increased demand for inspection robotics, mostly because of the challenging of manpower shortages (Press releases, 2015). Due to the difficulty in recruiting skilled inspectors and the rising expense of hiring them,

investors have started to prefer to deploy robotics, which has been developed to automate manual operations (Herm et al., 2022), to assist or even replace manual inspectors.

Although the number of published studies related to construction inspection robotics has decreased slightly after 2019, the market investment is increasing. It can be indicated that the construction inspection robotics is still attracting intentions, the emphasis in this stage shifts from researching for on-site experiments and implementation. The outperformed prototypes of construction inspection robots have gradually been applied in practice. It is predicted that the global market for inspection and maintenance robots will increase at a CAGR of 14.1% between 2021 and 2028 (Market Shapshot, 2022). The main players including Eddyfi Technologies, Gecko Robotics, Inc., Honeybee Robotics (Globenewswire, 2021). In summary, despite the decline in research acceptability, it is still a good chance to study construction inspection robotics because: 1) industry has started to embrace construction inspection robotics. 2) the market of construction inspection robotics isn't fully matured yet. Even though the investment is rising, there is little proof that the customers are satisfied with the products. Existing technologies still need to be enhanced in order to accelerate their on-site application and better satisfy practical demands. The criteria for accepting construction inspection robotics-related studies have, however, tightened at current stage. Scientific literature may only accept shared ideas that are significantly novel and contribute to guiding technological advancements.

Before 2014, 64% of studies were devoted to developing robotic mechanisms. 65%

of the mechanisms were designed as multi-joint robotics(topic4). 73% of the multi-joint robotics were developed for pipeline inspection. Mechanism of vacuum and pneumatic adhesion (topic8), magnet adhesion (topic6), and cable-driven mechanism (topic9) account for 18.3%, 10%, 6.7%, respectively. Different from pipeline inspection robots, the majority of adhesion and cable robotics were designed to inspect the surfaces of both internal and exterior walls in high-rise buildings. It revealed that the majority of earlier studies emphasized on developing robotics mechanisms. The most widely used prototype was the multi-joint device, invented for pipeline inspection.

Differently, 66.3% of research focused on building robotic control systems at the moment. The occurrence variation of works on image-processing methods (topics0 and 5), machine learning methods (topics 1 and 10), navigation methods (topics2 and 7) and multi-sensor methods (topics3 and 11) are depicted in **Figure 4.9.**

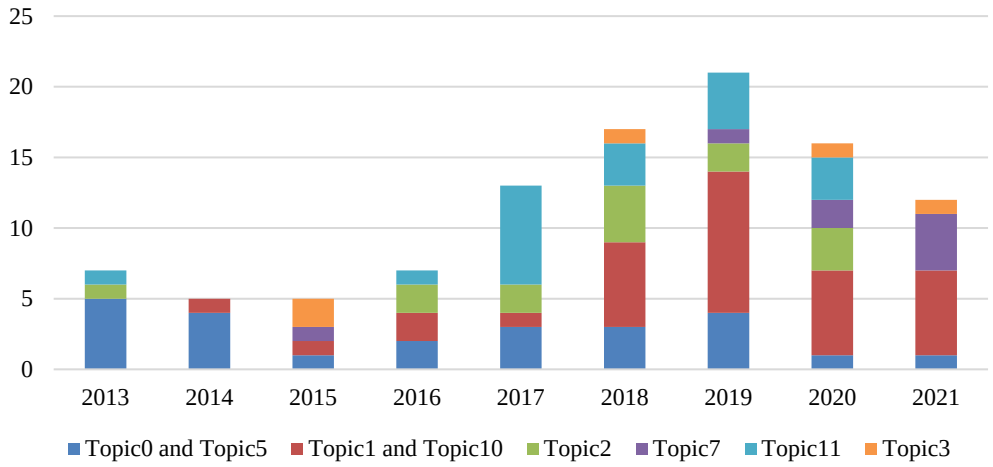


Figure 4. 9 Research trend of robotic control algorithms

It can be observed that the popularity of developing machine learning algorithms for robotic inspection exploded in 2019 and since then kept on growing. The majority of machine learning networks are designed to visually examine defects,

particularly cracks, on the surfaces of structures including buildings, bridges, tunnels, pavements, and pipelines. Deep CNNs, both self-developed and existing, such as VGG16 (Simonyan and Zisserman, 2014), MobilenetV2 (Sandler, et al., 2018), were most commonly used for construction inspection robotics. The results in lie with the findings of (Shanmugam, et al., 2022). Deep networks are likely to yield more precise inspection. After 2020, CNNs were not only used to identify defects but also to show their attributions, such as localizations, employing algorithms such as Faster-RCNN (Ren et al., 2015) and YOLO (Redmon et al., 2016). Studies on the use of CNNs to inspect defects in specific scenarios, including underwater, also demonstrate an increase. It can be indicated that: 1) simpler and faster robotic visual inspection techniques for crack inspection are needed nowadays. 2) Multiple attributions of defects, such as locations, areas, heights, and widths, should be provided by the designed CNNs. 3) Robotic inspection is becoming more and more necessary in special places, such as underwater.

Studies of navigation techniques for robotic control can be considered a “safe topic,” which appears the second most frequently after 2015. Before 2015, the majority of navigation algorithms were employed to control the motion of robotic arms using PID or kinematic matrix. An increasing number of navigation techniques, both local (FLC system) and global (SLAM, and BIM-based 3D map construction) is now developed to control the stability of flying or wheeled mechanisms, such as UAVs and autonomous vehicles.

The adoption of multi sensor-based techniques peaked in 2017 and then gradually decreased after that. Before 2017, NDT sensors for structure defect inspection, such

as electrical resistivity (ER) and impact-echo (IE) sensors, were commonly employed. However, there is an increasingly prevalent for robotic systems nowadays to be equipped with sensor groups, such as lasers and cameras, to provide integrated and continuous information for visual inspection. The explanations may be that the majority of construction inspection items can be visually detected, and it provides consistency and immediate effects (Ledford et al., 2018).

In summary, developed construction inspection robotics is now being widely promoted and applied by industry. Only research with a high level of novelty and contributions have a chance to be accepted now due to the increased popularity of construction inspection robotics. **Table 4.3** shows the research emphasis and their trends in the domains of robotic construction inspection over period.

Before 2014, pipeline inspection using multi-joint robotic mechanisms was the primary study focus. Currently, mechanisms that move more stable, freely, and easily controlled, such as UAVs and wheeled autonomous vehicles, gained more preference. Therefore, navigation algorithms, such as the FLC system and BIM model-based 3D map constructions, are encouraged to control the stability of flying or wheeled mechanisms. The developed robotic visual inspection algorithms were evenly employed to typical applications, such as pipelines, bridges, pavements, tunnels, and indoor and outdoor building. Among them, algorithms for surface cracks gained the most popularity. Simpler and faster CNNs that provide multi-attributions of defects, especially in hard-to-reach scenarios such as underwater, need to be improved nowadays to effectively adapt construction inspection robots for practical implementation.

Table 4. 3 Research emphasis over the period and future trend

Robotic technologies		Research emphasis (old)	Research emphasis (new)	Trend
Mechanism		Multi-joint robots for pipeline inspection	UAVs and wheeled autonomous vehicles for typical scenarios, including building indoors and outdoors, bridges, pavements, tunnels, pipelines	--Stable --Freely --Easy to control
Inspection technology	Image processing	Deep CNNs	CNNs provide diverse functions, such as .defect localization	--Simpler and faster --Provide multi-attributions of defects --Inspect defects in hard-to-reach locations
	Sound and multi-sensor based	NDT sensors, such as ER, IE	Visual sensors groups, such as lasers and cameras	Provide density and continuous visual data
Navigation technology		Motion control for robotic arms, such as PID	FLC system or BIM model-based 3D construction for mobile robotics navigation	Local and global algorithms to control UAVs and wheeled autonomous vehicles.

4.4 Summary

The supporting technologies for developing the robotic control system are discussed in this chapter. In general, supporting technologies include: 1) vision-based inspection technology, which refers to robotic control technique for robotic vision, and 2) navigation technology, which refers to robotic control technique for robotic autonomous navigation.

As for the vision-based inspection technology, automated computer vision

techniques, especially CNNs, are gaining popularity since the majority of quality defects can be visually inspected, which delivers consistent and prompt responses. A great number of the developed robotic vision algorithms have been implemented to inspect cracks in the surfaces of construction components. To meet practical requirements and ensure they can be utilized on robotic platforms, the visual algorithms must be simpler and faster to use. Therefore, the lightweight CNN architecture design technology was investigated in order to develop the vision module of the robotic control systems.

To achieve the autonomous navigation of mobile robotic platforms, local and global techniques, such as the FLC system and BIM-based 3D map construction, are encouraged. The local navigation approach was the focus for developing the robotic control system in the present research since it is more stable and adaptable to complex and changing building environments. For instance, the angular and velocity commands that control the movements of the robots can be generated and adjusted dynamically by computing the real-time sensor data using a fuzzy logic controller.

In addition, the visualization technology that commands the robot to visualize the inspected cracks to the users' screens was also investigated in order to achieve the robotic visualization.

The next three chapters of Chapters 5 to 7 describe how the development of the lightweight CNN, FLC system, and robotic visualization was conducted in detail.

Chapter5: Develop a Lightweight CNN for Robot Vision

5.1 Introduction

This chapter aims to design a lightweight CNN that can be successfully implemented on various robotic platforms without being limited to those powered by GPUs. To design the lightweight CNN architecture, 24 hyperparameter groups of convolution and pooling blocks were first established based on VGG-16 and trained to determine the network depth. Then, to reduce the network weight, 4×4 convolution filters with 1 or 2 strides were inserted into the adjacent convolution layers in each block. As a result, the proposed lightweight CNN maintains only 8.96 million weights. Evaluated by the confusion matrix, the lightweight CNN shows a good performance. The F1-scores reached 96.8% and 92.4% in the training and testing datasets, respectively. The performance of the proposed lightweight CNN, the state-of-the-art CNNs developed for crack inspection, as well as the pretrained lightweight CNNs, were compared to highlight the effectiveness. The mobile robotic platform Turtlebot 3, which is powered by a CPU processor, was employed to validate the feasibility. Details of lightweight CNNs, the designing process, the validation process, and a summary are shown in sections 5.2 to 5.5, respectively.

5.2 Lightweight CNNs

CNN algorithms have been continuously embraced in the construction domain since 2018 to achieve automated crack inspection in various construction scenarios, such as buildings, roads, and infrastructures (Ali et al., 2022). It has been proven that by introducing the visual-based automatic technique to the conventional inspection

process, the problems of time-consuming, subjective, and heavily dependent on skilled inspectors can be effectively avoided (Wang et al., 2022). Because of the obvious merits and the ease of applicability, both industry and government have shown high interest and supportive altitudes towards CNNs, which enables the topic to become popular in the research field.

As mentioned, although the existed CNN designs for building crack inspection indeed contribute to enhancing the performance to some extent, nearly all the designed deep CNNs need to be realized using powerful GPUs, such as the Nvidia GTX 1080 Ti (Li and Zhao, 2019) to compute millions or billions of parameters. Because of the computation complexity, complex CNN models, such as VGG-16 based models (Fang et al., 2019), may generate severe response delays on common computers (Bouguettaya et al., 2019).

As such, it is inconvenient for a number of real-life cases to implement CNNs into robotic platforms. Most of the inspection robots are controlled by the micro-processors that powered by CPUs instead of GPUs. Therefore, to make it applicable for inspection robotic platforms, lightweight CNNs, that can be executed in various CPU devices and not being limited to those powered by GPU, are needed. Meanwhile, developing lightweight CNNs has become a trend at the current stage, which aims to widen CNN implementation (Xue and Ren, 2021). After reducing the computing hyperparameters, fast response and real-time object inspection can be effectively ensured by the lightweight CNNs.

According to (Howard et al., 2017), the ways of obtaining lightweight CNNs can be divided into two categories: 1) compressing pretrained CNNs, such as (Nikouei,

et al., 2018), 2) directly training small networks, such as (Anvarjon et al., 2020). Among them, the idea of reducing weight by modifying the convolution process has been deemed the most efficient. For example, using the "depth-wise and point-wise" convolution in MobileNet (Howard et al., 2017), which has 15191 citation counts, or "using the 1×1 convolution filters instead of the 3×3 filters" in SqueezeNet (Iandola et al., 2016), which has 6551 citation counts. Lightweight CNNs contain only a small number of computing parameters, which substantially increases the programming efficiency. For example, when training with the same datasets, only 4.2 million parameters remine in the MobileNet-224, while 138 million parameters exist in the VGG-16. Because of the fast operation speed, the lightweight CNNs have been continuously employed in real-life object inspection applications (Sae-Lim et al., 2019, Bai et al., 2021, Kamel et al., 2020), including the crack inspection tasks (Li et al., 2018, Dais, 2021). However, it has also been proven that the inspection accuracy of the lightweight CNNs may be lower than that of deep CNN architectures in some cases because of the incomplete extraction of the image features (Taresh et al., 2021). In short, it is worth exploring ways to balance CNN's weight and accuracy.

5.3 Methods

Because deep architecture performs better with accuracy (Simonyan et al., 2014), the very deep CNN, VGG-16, was referenced as the base model to design the depth of the lightweight CNN. Although CNNs with deeper architectures are more accurate, the 16-layer depth was not directly employed because CNN's weights also increase along with the depth. For example, the number of convolutional filters in

VGG-16's first layer is 64, while the number increases to 128 in the fourth layer, which generates 1,728 and 147,456 weights in the first and fourth layers, respectively. Therefore, it is better to determine a suitable depth instead of employing the 16-layers directly. To do so, experimental groups that contain different convolutional and pooling blocks of VGG-16 were established and compared. The solutions to reduce the weights and memory were then presented.

5.3.1 Determine network depth

(1) Setup experimental groups

In VGG-16, the convolutional and pooling layers are presented as five separate blocks. The first and second blocks contain two convolutional layers each, while the following three layers contain three convolutional layers, respectively. The filter number in each convolutional block is 64,128, 256, 512, 512, respectively. The filter size in each convolutional layer is all 3×3. The convolutional stride is fixed to 1 pixel. The max-pooling layers follow the last convolutional layer in each block. All the pooling filter size is 2×2, with stride 2. Three fully connected layers follow the last convolutional and pooling block. The first two fully-connected layers contain 4096 neurons, activated by the ReLU function. The last fully-connected layer contains 1000 channels and is activated by the softmax function.

Based on the mentioned designs of VGG-16's convolutional and pooling blocks, four experimental groups were established. Specifically, the number of contained convolutional and pooling blocks increases gradually among the experimental groups. For example, the first experimental group contains the VGG-16's first convolutional and pooling blocks, and the second experimental group contains the

first and second blocks. Details of the four experiment groups are shown in **Table 5.1**. In addition, according to VGG-16, the convolutional filter size was normalized as 3×3 with stride 1. The spatial padding technique was applied to predict the pixels at the edges as completely as possible. The pooling filter size was set as 2×2 with a stride of 2. The max-pooling method was used. To decrease the number of computation parameters, different from VGG-16, two fully connected layers were designed, which have 1024 neurons in each layer. Neurons were activated using the Softmax function. The performance of the designed fully-connected layers has been proven in previous CNNs, such as AlexNet (Krizhevsky, et al., 2017) and LeNet (Lecun et al., 1998).

Table 5. 1 Convolutional and Pooling blocks in the experimental group

Experimental group	Convolutional layers	Pooling layers
CP_1	2 blocks (2+2 convolutional layers), filter number: 64x64(2) +128x128(2)	2 pooling layers following the final convolutional layer in each block
CP_2	3 blocks, including 2+2+3 convolutional layers filter number: 64x64(2) +128x128(2) +256x256(3)	3 pooling layers following the final convolutional layer in each block
CP_3	4 blocks, including 2+2+3+3 convolutional layers, filter number: 64x64(2) +128x128(2) +256x256(3) +512x512(3)	4 pooling layers following the final convolutional layer in each block
CP_4	5 blocks, including 2+2+3+3+3 layers filter number:	5 pooling layers following the final convolutional layer in each block

64x64(2) +128x128(2) +256x256(3) +512x512(3) +512x512(3)

CP1 to CP4 were then trained individually to examine their performance. In the training process, the learning rate was set from 0.1 to 0.0001 according to VGG-16. To maintain a proper variance and avoid extremely steep gradients, the experimental groups were trained with two normal batch sizes, 16 and 64. To preserve the information at the image edges, spatial padding was applied in the convolutional layers. The ReLU function was selected as the activation function for convolutional, pooling, and fully-connected layers to increase nonlinearity and improve efficiency in backpropagation. The dropout technique was also applied. In summary, the following hyperparameters within the given ranges were trained:

a. The number of convolutional layers (C): C1(2 blocks, including 2+2 convolutional layers); C2 (3 blocks, including 2+2+3 convolutional layers); C3 (4 blocks, including 2+2+3+3 convolutional layers); C4 (5 blocks, including 2+2+3+3+3 layers.)

b. The number of convolutional filters (Fc): range in the set {64x64, 128x128, 256x256, 512x512}

c. The number of pooling filters (Fp): range in the set {2,3,4,5}

d. Learning rate(L): range in the set {0.001, 0.0001, 0.00001}

e. Batch size (B): range in the set {16,64}

By combining the training parameters, the four experimental groups were detailed into 24 sub-groups using the grid searching method (Gan, 2014), seen in **Figure 5.1**.

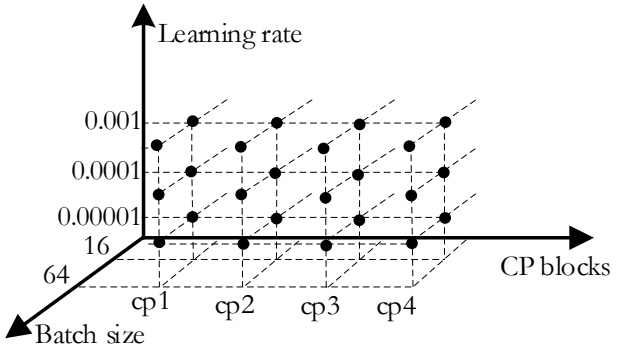


Figure 5. 1 Sub-groups of the four experimental groups

(2) Selection criteria

The efficient loss function, CrossEntropyLoss function (calculated with Equation (1)) (Cao et al., 2018), was employed to first examine the convergence ability of the experimental groups. The groups that successfully converged were then validated with the testing dataset to test whether the overfitting and underfitting problems existed. Finally, the hyperparameter set reached lowest training loss with the fastest speed and without overfitting or underfitting problems was selected as the most appropriate design of convolutional, pooling, and fully-connected layers. The calculation process was programmed in Python and the Pytorch package because of its high computing efficiency (Zhou et al., 2020).

$$\text{loss}(x, c) = -x[c] + \log\left(\sum_{j=0}^{C-1} \exp(x[j])\right) \quad (1)$$

Here: C means the label of the input. $x = [x_0, x_{C-1}]$, x[j] means the outputs of x_0 to x_j.

(iii) Training and testing dataset

The dataset was collected from the world's most significant machine learning and data science community: Kaggle. The Kaggle community offers thousands of public datasets, covering a wide research range, such as transportation, medical, and

construction (Bojer et al., 2021). The images in this research were collected by searching for the keyword “crack inspection” in the Kaggle datasets and competition modules. Finally, 40000 images in the “Surface Crack Inspection” dataset contributed by Çağlar Fırat Özgenel (Özgenel and Çağlar, 2018), 56070 images in the “SDNET2018” dataset provided by Dorafshan (Dorafshan et al., 2018), were selected because of the high integrity and image clarity. Overall, a database of 96070 images was prepared. The images contain various building surfaces with and without cracks. The 96070 images were uninformed as 224 × 224 pixel resolutions with RGB channels. The cracks in the images are as narrow as 0.06mm and as wide as 25mm. The dataset also carried a variety of obstructions, such as shadows, surface roughness, scaling. With the help of different backgrounds and various obstructions, the network robustness can be effectively ensured. Some samples of the images are shown in **Figure 5.2**. The dataset was composed of 30626 and 65444 concrete surface images with and without cracks. The database was randomly divided into training and testing datasets at a ratio of 70/30. **Table 5.2** presents the details of the prepared database.

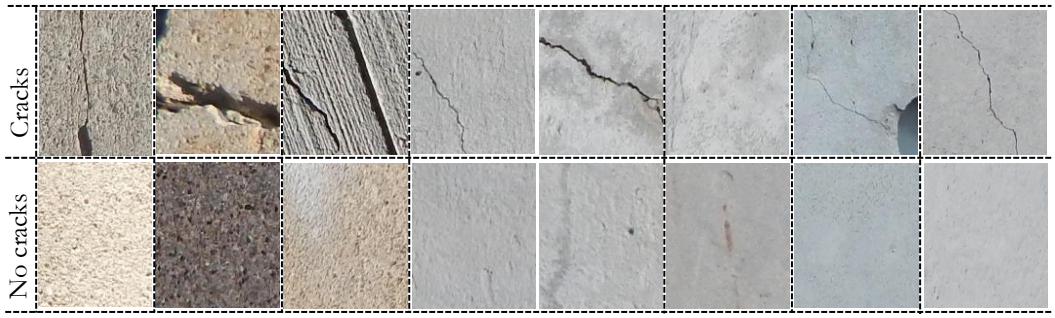


Figure 5. 2 Examples of cracked and non-cracked concrete surfaces

Table 5. 2 Database details

Total database		Training dataset	Test dataset
Total number of images	96070	67249	28821

Number of cracked images	30626	21463	9163
Number of non-cracked images	65444	45786	19658
Image pixels	224×224×3		

All the training and testing processes were conducted using the Kaggle kernels. This virtual environment was equipped with the NVIDIA Tesla K80, including a dual GPU design, 24GB of GDDR5 memory. That way, it enabled faster training and testing by 5–10 times faster than a CPU-only system. The Adaptive Moment Estimation (Adam) algorithm was used for stochastic optimization. 67249 training images were fed into each hyperparameter set. The training epochs were initially set at 30.

5.3.2 Reduce network weight

This research intended to reduce the network weights and memory by lowering the number of computational parameters. The computation parameters exist in both convolutional and fully-connected layers. There are no computing parameters in the pooling process because it is employed to reduce filter size instead of learning features (Tang, et al., 2019). The number of computing parameters in the convolutional and fully-connected layers is calculated using Equations (2) and (3), respectively. As shown in the equations, the network weights can be effectively reduced by decreasing the number and size of convolutional filters or the number of input tensors of the fully connected layers. This research chose to remine the designs of the convolutional filters to ensure the performance of extracting feature maps (Agrawal and Mittal, 2020). Instead, the number of input neurons of the fully connected layers was reduced.

$$W_{ic} = F_{h_i} \times F_{w_i} \times I_{i-1} \times F_{n_i} \quad (2)$$

Here: W_{ic} is the weight of the $i-th$ convolutional layer. F_{h_i}, F_{w_i}, and F_{n_i} are the height, width, and number of the convolutional filters in the $i-th$ convolutional layer, respectively. I_{i-1} is the number of feature map inputs in the $i-1-th$ convolutional layer.

$$W_{if} = F_{n_i} \times F_{m_i} \quad (3)$$

Here: W_{if} is the weight of the $i-th$ fully connected layer. F_{i_i}, F_{o_i} are the number of inputs and outputs in the $i-th$ fully connected layer.

The inputs of the fully-connected process can be reduced by gradually decreasing the input dimensions in each convolutional layer. To do so, the max pooling layers were first inserted between the adjacent convolutional layers, which can be seen as the simplest way. However, the results showed that the models tend to fail to converge in the training process when using too many pooling layers. The reason may be that pooling operators only consider the maximum or average pixels within a pooling area (Ma et al., 2018), and it is possible to ignore part of the pixel information, such as the discerning features. Therefore, the 4×4 convolutional filters with 1 pixel padding and strides of 1 or 2 were employed to reduce the image dimensions and keep the image features as complete as possible. The idea is based on the relationship between convolutional parameters and input dimensions, as shown in Equation (4). It is worth noticing that by employing the 4×4 convolutional filters and 2 strides, the input dimensions can be reduced to half in any convolutional

layer design. For example, when the input dimension is 224×224, the output dimension can be reduced to 112×112 (((224-4+2×1)/2)+1), which can demonstrate the same effect of the pooling process.

$$O_i = \frac{(I_i - D_{I_i} + 2P_i)}{S_i} + 1 \quad (4)$$

Here: O_i is the output dimension in the $i-th$ layer. I_i is the input dimension in the $i-th$ layer, D_{I_i} is the convolutional filter size in the $i-th$ layer, P_i is the padding pixel in the $i-th$ layer, S_i is the convolutional filter stride in the $i-th$ layer.

5.3.3 Evaluate network performance

To validate the robustness and efficiency of the proposed lightweight architecture, its performance was further evaluated using the confusion matrix (Townsend, 1971), shown in **Figure 5.3**. In the confusion matrix, TP means the number of images that are predicted as the surface with cracks, and it is also the cracked surface in reality. TN means the number of images that are predicted as no-cracked images, and it is also the non-cracked surface in reality. FP means the number of images predicted as surfaces with cracks, but the actual classification is surface without cracks. FN means the images are predicted as a surface without cracks, but the actual classification is a surface with cracks. Therefore, all the correct predictions were shown in the diagonal of the confusion matrix. The bigger the values in the diagonal, the more images that are predicted correctly. The networks' accuracy, precision, error, recall, and F1-score were then computed as follows using TP, FN, FP, and TN:

(i) $\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} ? 100$

(ii) $\text{Precision} = \frac{TP}{TP + FP} ? 100$

(iii) $\text{Error} = \frac{FP + FN}{TP + TN + FP + FN} ? 100$

(iv) $\text{Recall} = \frac{TP}{TP + FN} ? 100$

(v) $\text{F1-score} = 2 创 \frac{precision' recall}{precision + recall} 100$

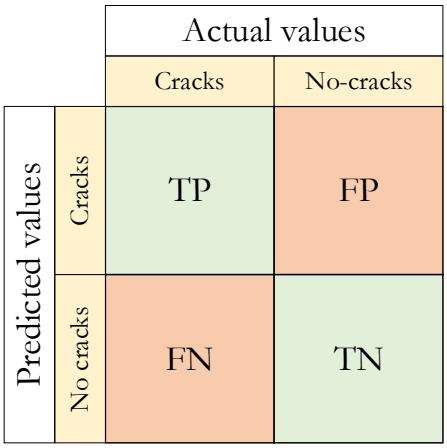


Figure 5. 3 Confusion matrix

5.4 Results

5.4.1 Network Depth

Training loss of the 24 experiment groups within 30 epochs is shown in **Table 5.3**. As mentioned, the network depth increased gradually from CP1 architectures to CP4 architectures. When increasing the network depth from CP1 to CP2, the minimum training loss decreased from 0.350841 to 0.335218. However, when expanding the network depth from CP2 to CP3 and CP4, the minimum training loss increased from 0.3335218 to 0.355781 and 0.357323. The results showed that although a deeper network is more accurate for most cases, it is possible for CNNs to perform slower or fail converge when simply increasing network depth.

Table 5. 3 Training loss of the designed hyperparameter sets

ID	Convolutional and Pooling blocks	Learning rate	Batch size	Average loss		
				Min	Max	Average
CP1_1	64x64(2) + Pooling +128x128(2) + Pooling	0.001	64	0.632339	0.632502	0.632426
CP1_2		0.001	16	0.632408	0.632455	0.632420
CP1_3		0.0001	64	0.350841	0.446381	0.375746
CP1_4		0.0001	16	0.352687	0.46959	0.383440
CP1_5		0.00001	64	0.385688	0.494764	0.404034
CP1_6		0.00001	16	0.377871	0.457704	0.396053
CP2_1	64x64(2) + Pooling +128x128(2) + Pooling +256x256(3) + Pooling	0.001	64	0.632339	0.632473	0.632405
CP2_2		0.001	16	0.632408	0.632500	0.632421
CP2_3		0.0001	64	0.335218	0.502493	0.369339
CP2_4		0.0001	16	0.631363	0.632423	0.632383
CP2_5		0.00001	64	0.626970	0.632837	0.632183
CP2_6		0.00001	16	0.359159	0.488842	0.390187
CP3_1	64x64(2) + Pooling +128x128(2) + Pooling +256x256(3) + Pooling +512x512(3) + Pooling	0.001	64	0.632309	0.632517	0.632425
CP3_2		0.001	16	0.632408	0.63245	0.632422
CP3_3		0.0001	64	0.626753	0.632532	0.631788
CP3_4		0.0001	16	0.524263	0.632423	0.625862
CP3_5		0.00001	64	0.371104	0.527704	0.398979
CP3_6		0.00001	16	0.355781	0.488589	0.384983
CP4_1	64x64(2) + Pooling +128x128(2) + Pooling +256x256(3) + Pooling +512x512(3) + Pooling +512x512(3) + Pooling	0.001	64	0.632339	0.632504	0.632421
CP4_2		0.001	16	0.632408	0.632487	0.632421
CP4_3		0.0001	64	0.357323	0.595143	0.390924
CP4_4		0.0001	16	0.632408	0.632473	0.632420
CP4_5		0.00001	64	0.368884	0.534750	0.393635
CP4_6		0.00001	16	0.360630	0.492472	0.382530

Figure 5.4 shows the comparison of the convergence performances of the

experimental groups that converged successfully. The groups that failed to decrease training losses, such as CP1_1, were excluded. The results showed that CP1_3 performed with minimum training loss in the first and second epochs. In the 3 to 6 epochs, CP4_6 showed the best convergence ability. After the sixth epoch, CP1_3 and CP2_3 showed the best convergence performance, and CP2_3 converged faster than CP1_3 and reached the lowest training loss. Therefore, the CP2_3 network, with a 9-layer depth, was initially determined as the optimal network depth.

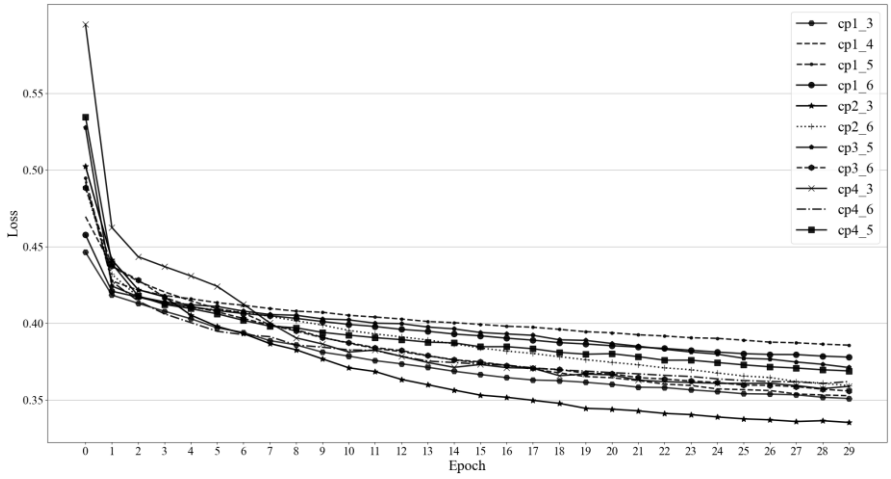


Figure 5. 4 Convergence comparison of the designed hyperparameter sets

To see whether the training loss of CP2_3 reached minimum, the CP2_3 was trained again with more epochs. The result showed that there is no obvious change in the average training loss of CP2_3 when increased epochs from 30 to 60. CP2_3 reached the optimal performance at the 30 epochs. The average training loss of CP2_3 reached 0.33 at epoch 30 then fluctuated at 0.33 from 31 to 60 epochs.

To see whether there is underfitting or overfitting of the 9-layer depth, its loss and accuracy in both training and testing datasets are shown in **Figure 5.5.** It can be observed that there is a steady variation trend of loss and accuracy in the training process. The training loss decreased steadily from 0.48 to 0.33 at the 25th epoch

and then maintained the same. Similarly, the training accuracy increased steadily from 0.82 to 0.97. The variations in the tendencies of the loss and accuracy in the testing process are consistent with the training process. It proves that there is no overfitting or underfitting problem with the CP2_3 model. The testing loss decreased from 0.42 to 0.31, while the testing accuracy increased from 0.88 to 0.99. Therefore, the depth of the designed lightweight CNN was initially determined as 9-layer. Details of the 9-layer architecture are shown in **Figure 5.6**.

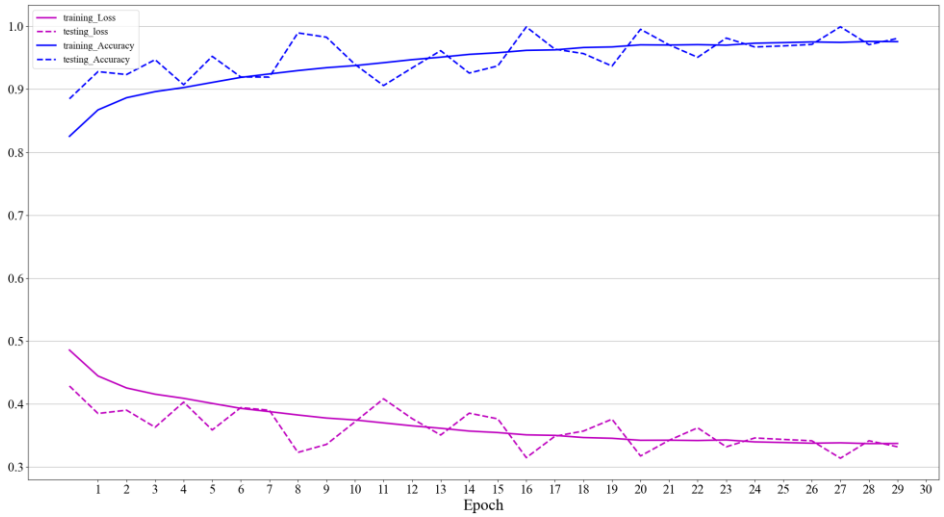


Figure 5. 5 Performance of CP2_3 hyperparameter set

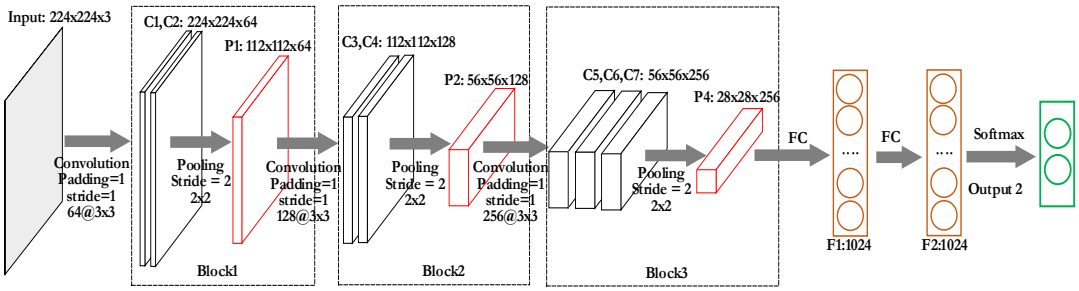


Figure 5. 6 Architecture of the 9-layer CNN

5.4.2 Network Weight

Although the 9-layer depth reached the lowest loss with a fast converge speed and no underfitting or overfitting problem, its weight is even greater than the entire

VGG-16 network. Calculated based on Equation 3, the weights of the 9-layer model and VGG-16 are 208,305,856 and 134,256,320, respectively. After inserting four 4×4 convolutional layers between the adjacent convolutional layers in the three convolution and pooling blocks, the 9-layer model was converted to a 13-layer model, and its weight was sharply decreased. **Table 5.4** lists the weight and memory of the VGG-16, the 9-layer CNN, and the lightweight 13-layer CNN when training the same datasets. The results show that the lightweight 13-layer CNN performs the least computation parameters and memory. The weight and memory of the 13-layer CNN are nearly 23 and 15 times smaller than the 9-layer model and the VGG 16 model, respectively.

Table 5. 4 Comparison of VGG-16, the 9-layer model, and the 13-layer model

Model architecture	Weight	Memory
VGG-16	134,256,320	421M
9-layer CNN	208,305,856	790M
13-layer CNN	8,961,728	24.8M

The architecture of the improved 13-layer model is shown in **Figure 5.7.** The 4×4 convolutional filters are surrounded by one padding layer. The first and the second blocks were inserted with one 4×4 filters move with two strides, respectively. The third block was inserted with two 4×4 filters, the first one moves with 2 strides and the last one moves with 1 stride. By doing so, the inputs of the fully-connected layer can be reduced from 28×28 to 3×3 without employing extra pooling layers, thereby decreasing model weights and memory.

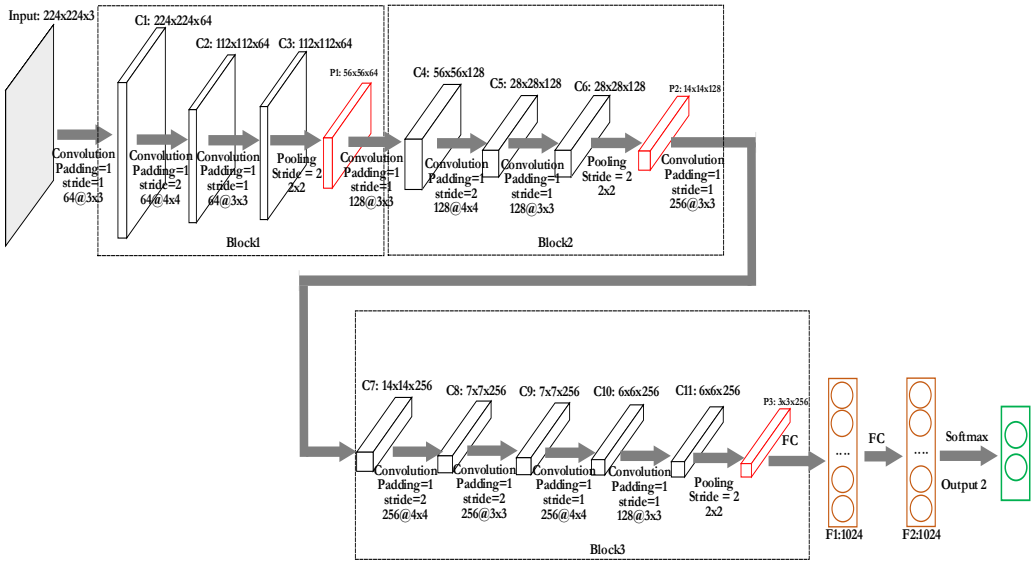


Figure 5. 7 Architecture of the lightweight 13-layer CNN

Table 5.5 shows the comparison of training loss and accuracy, testing accuracy of the 9-layer and the lightweight 13-layer architectures. The results show that although the input dimensions were reduced, the improved lightweight architecture outperforms the 9-layer model in both training and testing datasets. The maximum training and testing accuracy of the 13-layer CNN reached 0.979 and 1.000, respectively, while its average training loss reached 0.355, which is 0.015 lower than the 9-layer model.

Table 5. 5 Comparison of loss and accuracy of the lightweight 13-layer and 9-layer CNN

Model architecture	Accuracy				Loss	
	Training accuracy		Testing accuracy		Training loss	
	Average	Max	Average	Max	Average	Min
9-layer model	0.942	0.976	0.952	0.999	0.370	0.337
13-layer lightweight model	0.956	0.979	0.953	1.000	0.355	0.334

5.4.3 Network Performance

The proposed lightweight 13-layer CNN can be proved with outstanding performance by the confusion matrix, shown in **Table 5.6**. First, the lightweight model performs with high accuracy and a lower error rate. The accuracy reached 98.0% and 95.3% in the training and testing datasets, respectively, and the error rate reached 2.0% and 4.7%, respectively, which outperforms most of the existing models for crack inspection. For example, CNN design in (Cha et al., 2017). In addition, the precision reached 98.8% and 94.1% in the training and testing datasets, respectively. It proves the designed lightweight CNN has high accuracy in both crack and no crack scenarios, which is higher than most of the CNNs. For example, CNNs in (Dung, 2019).

The recall in the training and the testing dataset reached 94.9% and 90.8%, respectively, which outperforms most existing CNNs, for example, 3% higher than the CNN design in (Zhang et al., 2016). The high recall score indicates the accurate inspection of cracked surfaces. For the training dataset, 21463 images are the images of cracked surfaces. Among them, 98.8% of the images were correctly detected as surfaces with cracks. Among the 9133 crack images in the testing dataset, 94.1% of them were correctly detected. Meanwhile, the F1-score in the training dataset and the testing dataset reached 96.8% and 92.4%, respectively, which indicates the high robustness of the lightweight CNN design. The F1-score of most proposed CNNs for crack inspection is rarely higher than 90%. For example, F1-scores in (Yang et al., 2018, Dung, 2019, Zhang et al., 2017), and the 9-layer CNN in this research are 79.95%, 88.6%, 89.6%, and 90.13%, respectively.

Table 5. 6 Values of confusion matrix and the evaluation criteria

Dataset	Accuracy	Precision	Error	Recall	F1-score

Training	98.0%	98.8%	2.0%	94.9%	96.8%
Testing	95.3%	94.1%	4.7%	90.8%	92.4 %

5.4.4 Comparison with state-of-the-art CNNs

To validate the advantages of the proposed 13-layer lightweight CNN, its weight and performance were compared with the state-of-the-art CNNs that developed for crack inspection. Each selected CNN is representative. Their citation counts are no less than 290. **Table 5.7** shows the weight, accuracy, and F1 score of different CNNs. It can be observed that the proposed 13-layer CNN is lighter than most of the existing CNN and with higher precision, recall, and robustness. The precision of the 13-layer CNN reached 94%, which is at least 3% higher than the previous CNNs. The recall score of the proposed lightweight CNN reached 91%, which is similar to the findings in (Liu et al., 2019), while its weight is nearly 3.5 times lighter. The precision score is also 12%, 3%, and 2% higher than other CNNs. The F1-score of the proposed lightweight CNN reached 92%, which is at least 2% higher. The high F1-score proves the strong robustness.

It should be noted that the weight of the 13-layer CNN is nearly two times greater than the CNN proposed in (Dung, 2019), although it presents a better performance. In Dung's research, the CNN was designed as a U-net, which contains only part of the convolutional and pooling layers in VGG-16, and the fully-connected layers were discarded. Because the majority of computation parameters exist in the fully-connected layers, which were eliminated in the U-net, the U-net provides lighter weight. Although the U-net indeed provides an idea to design lightweight CNNs, the model does not accurate as well as the proposed CNN. In addition, the U-net cannot be trained directly. Pre-trained weights are needed. For example, the pre-

trained weights of VGG-16 on the ImageNet dataset (Russakovsky et al., 2015) were used to train the crack image datasets, which indicates that the U-net may be sensitive to the prepared datasets.

Table 5. 7 Weight and performance comparation with existed crack inspection CNNs

CNN design	Citation	Weight	Precision	Recall	F1 score
Liu et al., 2019	290	31,311,321	90%	91%	90%
Yang et al., 2018	345	138,860,224	82%	79%	80%
Zhang et al., 2017	570	9,922,141	90%	88%	87%
Dung, 2019	510	4,122,304	91%	89%	90%
Present	--	8,961,728	94%	91%	92%

To validate the advantages in terms of accuracy, the performance of the 13-layer CNN was compared with popular lightweight CNNs. Performance comparison of the present 13-layer CNN and the ResNet18 and MobileNet-V2 was summarized in **Table 5.8** according to the findings of (Ethisham et al., 2022). The comparison can be deemed rational because the quality and pattern of the images used in Ethisham's research are nearly the same as the images used in the present study. It can be observed that the accuracy, precision, recall, and F1-score of the MobileNet-V2 model are 15.1%, 27.3%, 1.7%, and 10.2% lower than the proposed 13-layer CNN, although it is 5.4M lighter. The proposed CNN also outperforms the ResNet18 model in terms of accuracy, precision, and F1-score. Although the 13-layer model's recall score is slightly lower than ResNet18, it is 2.8 times lighter than ResNet18.

To avoid the effects of the number of images in the training and testing datasets, the

MobileNet-V1 model was further trained and tested using the prepared dataset in the present study. The results show that although the weight and memory of MobileNet-V1 are lower than the 13-layer model, the accuracy, precision, and F1-score of the proposed 13-layer CNN are higher. It should be noted that the MobileNet-V1, trained with the present datasets, shows a high precision score, which shows its high efficiency in predicting cracked surfaces. However, the MobileNet-V1 model may be inefficient for the real-life scenarios that are surrounded by both cracked and uncracked surfaces, as indicated by the low recall score. The findings indicate that using 4×4 convolutional filters with 1 or 2 strides is an effective way to reduce CNN weight.

Table 5. 8 Weight and performance comparison with existed lightweight CNNs

CNN design	Dataset	Weight	Memory	Accuracy	Precision	Recall	F1-score
ResNet18	(Ethisham et al., 2022)	11.7M	44M	80.2%	64.1%	93.5%	85.0%
MobileNet-V2		3.5M	13M	80.5%	66.8%	89.1%	82.2%
MobileNet-V1	Present study	3.5M	13M	92.6%	98.0%	78.2%	87.1%
Present study		8.9M	25M	95.3%	94.1%	90.8%	92.4 %

5.4.5 Validation in Robotics

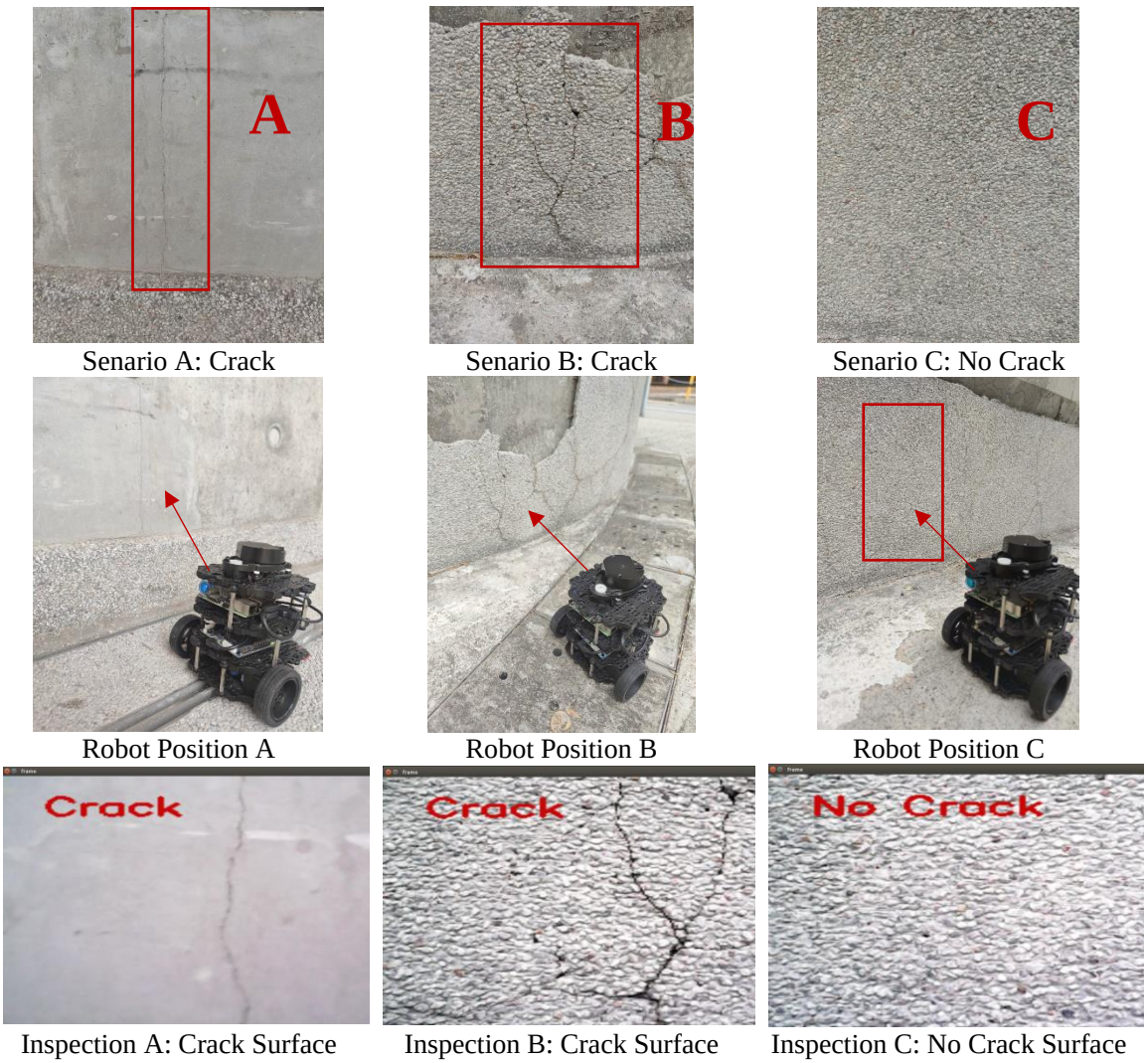
On-site validation was conducted at the Hong Kong Polytechnic University to examine the feasibility of the proposed 13-layer CNN in the robotic platforms. A mobile robot, Turtlebot3 burger, was employed to execute the proposed CNN program. Turtlebot3 burger is driven by the Raspberry Pi 3B+ control board, which

is equipped with a 64-bit CPU. Navigated by a wall-following algorithm, the robot was expected to automatically follow the wall and detect the cracks in real-time. A USB camera with a 4mm lens, mounted on the robot's right side, was employed to capture environmental videos. After computing the photographs using the 13-layer CNN, signs of "No Crack" and "Crack" were shown on the top left of the image window accordingly to present the inspection results.

The most commonly used You Look Only Once v3(YOLOv3) (Redmon and Farhadi, 2018) algorithm was tested first to compare the performance of the heavyweight and the lightweight CNN in the robotic system. The backbone architecture of YOLOv3 is the Darknet-53 model. The weight of Darknet-53 is more than 62 million, which is seven times greater than the proposed 13-layer CNN. Because of the large number of computation parameters, the Raspberry Pi 3B+ failed to execute YOLOv3 and the system got stuck as a response. Instead, the Raspberry Pi successfully executed the program of the lightweight 13-layer CNN and demonstrated both cracks and non-cracked surfaces. As seen from **Figure 5.8,** the cracked and non-cracked surfaces in the three experimental scenarios were correctly detected by the robots in real-time.

Although it is proven that the proposed lightweight CNN is feasible for CPU-driven devices, a slight delay was also observed when executing the deep learning programs in the ARM Cortex-A53 processor of the Raspberry Pi 3B+. As it is important to ensure a high frame rate, the configurations of the ARM Cortex-A53 processor can be considered as the minimum standard when choosing control boards for deep-learning based object inspection tasks. More efficient CPUs, such as the

ARM Cortex-A72 in the Raspberry Pi 4B+, are recommended. In addition, because of the limitations of the communication protocol, the delays become even more severe when using the “ssh-Y” command to connect the master computer and the host robot to remotely demonstrate the inspection outcomes in real-time. Therefore, to ensure the screen fluency, it is recommended to use the socket communication protocol, such as WebSocket, to connect the robot and the remote master computer instead of using the ssh command.



Senario A: Crack | Senario B: Crack | Senario C: No Crack

Robot Position A | Robot Position B | Robot Position C

Inspection A: Crack Surface | Inspection B: Crack Surface | Inspection C: No Crack Surface

Figure 5. 8 Crack inspection validation in Turtlebot3

5.5 Summary

To develop the robotic vision module, this chapter introduces a lightweight CNN architecture that can be successfully implemented for the majority of the robotic platforms. To do so, the network depth was first determined by comparing the performance of 24 experimental CNN groups. The experimental groups were established by combining the convolutional and pooling blocks of the VGG-16 model. 4×4 convolutional filters with 1 or 2 strides were inserted between the adjacent convolutional layers to reduce CNN's weight and memory. As a result, only 8.96 million parameters remained, which is lighter than most of the state-of-the-art CNNs for crack inspection. Meanwhile, the F1-score reached 96.8% and 92.4% in the training and testing, respectively, which outperformed the pretrained lightweight CNNs. On-site validation was conducted to illustrate the feasibility. It is validated that the proposed CNN can be successfully implemented in the mobile robot that is powered by the Raspberry Pi processor. Both the cracked and non-cracked surfaces were inspected automatically and correctly.

It is worth noticing that a slight delay exists when executing the proposed lightweight CNN in the Turtlebot 3 robot. The delay can be ignored when receiving the real-time outputs on an external screen that is directly connected to the robot's control board. However, because of the limitations of communication protocols, the delay becomes severe when using the "SSH-Y" commands to remotely receive the image frames on a master computer's screen. Therefore, a web-user interface that employs the webpage-based communication protocols was developed in Chapter 7 to realize the smooth and remote visualization. Before Chapter 7, a robotic autonomous navigation strategy is presented in Chapter 6. The developed

autonomous navigation algorithm assists in controlling the robot to automatically imitate the inspection motions by following the building components and conducting in-depth crack scanning.

Chapter6: Develop a Fuzzy Logic Controller for Robot Navigation

6.1 Introduction

The former section introduces the lightweight CNN developed for the robotic control system to realize vision-based automated crack inspection. This chapter developed an autonomous navigation strategy for the robotic control system to control the motions of the mobile robot.

In the robotics field, various autonomous navigation algorithms have been developed. Among them, this research focused on the wall-following algorithm (Yata and Yuta, 1998), a significant robotic motion control method, since it aids in controlling the robot to imitate the crack inspection behaviors of following the buildings and performing detailed inspection scanning. While the other algorithms, such as the occupancy grids approach (Moravec 1985) and the Kalman filtering approaches (Ayache and Faugeras 1989, Crowley 1989), concentrate more on the robots' navigation from the start locations to the target locations, which are not suitable for the motions engaging in crack inspection. Due to the simple logic, the conventional wall-following algorithms, particularly when employed to follow building components with arbitrary shapes such curving inside walls, are ineffective. When navigating using the conventional wall-following behavior, the wavy movements that affect the CNN inspection's accuracy are similarly challenging to prevent.

To enhance the wall-following navigation for the building crack inspection robots, a fuzzy logic controller was designed. To be specific, the ranges of the input data,

the membership functions, and the decision-making rules were detailed based on the configurations of the camera and the robot, building environments, and inspection criteria. Validated in both simulation and on-site building environments, the designed FLC system successfully controls the robot to conduct “finding wall”, “turning”, “wall-following”, and “obstacle avoidance” behaviors in various unknown building scenarios, including irregular regions such as concave and convex walls and narrow aisles. In addition, the FLC controls the robot to follow walls straight within a desired distance, and the path deviation problem can be effectively avoided. The outcomes can be easily coded into diverse mobile robotic platforms for their autonomous navigation, which contributes the industry with a fully automated robotic inspection system for daily building crack inspection work. Details of the designed FLC system for the wall-following behavior and the simulation and validation outcomes are presented in sections 6.2 to 6.5, respectively.

6.2 Navigation strategies of crack inspection robots

The basic theory, advantages, and limitations of the robot’ autonomous navigation, wall-following behavior, and designing FLCs for the wall-following behavior of building crack inspection robots are discussed to first explain the overall knowledge.

6.2.1 Navigation method for building crack inspection robot

Autonomous navigation is essential for building crack inspection robots to achieve a fully automated inspection process. Although various navigation strategies have been developed in the robotics field, few of them have been introduced to control the motions of building crack inspection robots. As a result, human interventions are required for the majority of inspection robots to control their movements and

speeds (Bui et al., 2020). For example, operators remotely control the movement and speed of the building inspection robots using joysticks (Özaslan, T et al., 2017, Kaiwart et al., 2022). Specifically, operators employ joysticks to control the robots to move forwards, backwards, and turn after observing the visions of the surrounding environments, which are provided by cameras or virtual reality models.

Even so, some of the popular algorithms have begun to be integrated into the building crack inspection robots to gradually achieve autonomous navigation. Among them, the simultaneous localization and mapping (SLAM) technique (Durrant and Bailey, 2006) has been frequently implemented for building inspection robots to conduct on-site quality inspections (Asadi et al., 2021, McLaughlin and Narasimhan 2020). In SLAM, an environmental map is needed in advance to obtain the positions of surrounding objects and the robots to plan their moving paths. For that reason, it is not convenient to employ SLAM for continuous building crack inspection work, although it contributes to autonomous and accurate navigation. For example, it is common to place or move objects in buildings, such as wardrobes, while the robot is operating. Because of the high computation cost of map construction, it is easy for the SLAM algorithm to fail to generate new path plans when objects change places, and a collision may occur. Motivated by the need to investigate appropriate navigation strategies for building crack inspection robots in unknown environments, the present research aims to investigate the local navigation strategies for the autonomous motion control of building crack inspection robots, which directly process real-time sensor-provided position information (Gul and Nazli 2019).

6.2.2 Wall following algorithm

The present research focused on the wall-following algorithm (Saman and Abdramane, 2013), one of the significant ideas of robotics (Che et al., 2022, Wu et al., 2021), among various local navigation strategies because it is an appropriate local navigation solution for autonomous crack inspection tasks and has been continuously implemented in the crack inspection robotic platforms (Wei et al., 2017). Other local navigation strategies, on the other hand, place a greater emphasis on generating a moving path from the start points to the goal points (Altman, 1992, Katoch et al., 2021), or obstacle avoidance (Poli and Blackwell, 2007). Controlled by the wall-following behavior, the robots can constantly follow building components within a desired distance to conduct detailed crack inspection scanning.

Initially, wall-following behavior was developed as an efficient way to achieve autonomous navigation in maze environments (Saman, 2013). By computing the position data, obtained by sensors, with decision-making loops, the wall-following behavior guides the robot to turn right, left, and straight to follow the maze walls until it reaches the goal location. The principle is depicted in **Figure 6.1**.

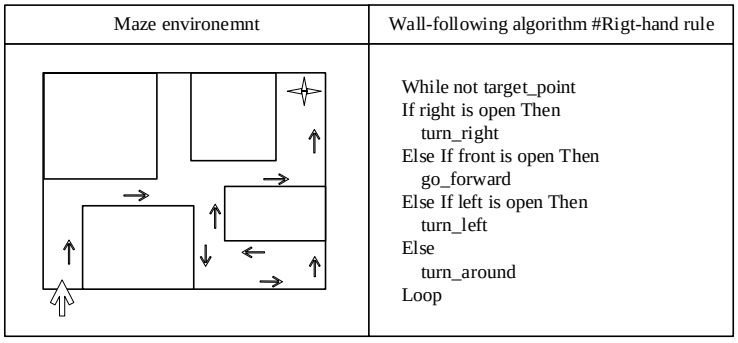


Figure 6. 1 Principle of wall-following algorithm

However, because of the simple logic, the initial wall-following behavior is ineffective for real-life building environments. Especially for the building

components with arbitrary shapes, such as curved columns, concave and convex corners. For example, when the robot follows the rounded columns, the second rule "IF front is open, Then go_forward" may be triggered to control the robot to go forward instead of following the rounded columns, which generates path deviations. In addition, the traditional wall-following algorithm fails to keep the robot following walls in a straight line and within a desired distance, which makes it hard to capture distinct pictures for computer-vision based crack inspection. Therefore, the present research intends to improve the robustness of the wall-following logic for building inspection robots. The purpose is to ensure the building crack inspection robots to smoothly follow building components in various unknown building environments within a desired distance.

6.2.3 Fuzzy logic controller

The significance of wall-following behavior has made it a worthwhile topic to discuss its optimization strategies (Xue et al., 2020). Several ways have been proposed to enhance wall-following behavior, including machine learning algorithms (Hammad et al., 2019, Teng et al., 2020) and the fuzzy logic controller (Omrane et al., 2016, Fatmi et al., 2006, Malhotra and Sarkar 2005, Faisal et al., 2013). Among them, the present research employs the fuzzy logic controller for the following reasons: 1) FLC has been proven as an effective tool and is widely used for improving wall-following behavior because of its outstanding ability to deal with complex uncertainty, such as various decision-making rules (Suwoyo et al., 2020). 2) FLC is more applicable because it can be simply coded and computed. It is an efficient navigation strategy for most robotic systems that are controlled with CPUs, such as the Raspberry PI. Powerful GPUs are needed to compute a hundred

million parameters in the machine learning process.

Similarly to the wall-following algorithm, the FLC generates navigation commands for robots based on input data and linguistic decision-making rules. Differently, the output of the FLC is more precise by computing precise input data with comprehensive decision-making rules and the membership functions (Novák et al., 2012). Instead of directly employing a developed FLC, a new one was designed for the wall-follow navigation of building inspection robots because: 1) it is hard to find a suitable one for building crack inspection robots. Majority FLCs were designed for the navigation problem of traveling from the start points to the goal points with obstacle avoidance instead of the wall-following behavior (Aouf et al., 2019, Singh and Thongam, 2018). 2) Because of the inappropriate design of the ranges of input data, the ranges and types of membership functions, and the decision-making rules, existing FLCs designed for wall-following behavior are not accurate enough for real-life building environments. For example, path deviation happens in concave regions and wavy motion occurs in the FLC designed by (Muthugala et al., 2020).

6.3 Methods

6.3.1 Robot kinematic model

In this research, the Turtlebot3 burger was employed as the testing robot. The Turtlebot3 burger is a three-floor octagon-shaped platform. The distance between the left and right wheels is 160mm (L), and its radius is 33mm (R). A 360-degree laser is installed on the top floor to obtain position information. The scanning distance ranges from 120mm to 3,500mm. A simplified depiction of the robotic platform is shown in **Figure 6.2**.

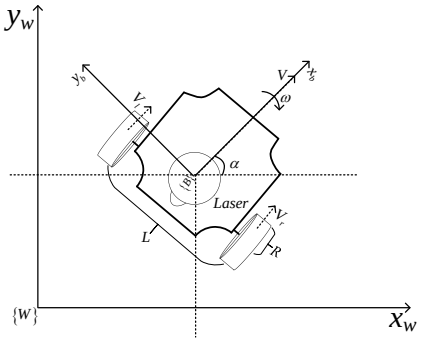


Figure 6. 2 Kinematic model of Turtlebot3 burger

Here, the established world frame and body frame are presented as $\{W\}$ and $\{B\}$, respectively. x_b, y_b refer to the heading directions of the robot, α refers to its orientations. V_l and V_r refer to the velocities of the left and right wheels, respectively. V and ω, referring to the linear and angular velocity, are directly used to control robot's movement. The kinematic dynamics can be explained using the mentioned variables as equation 1:

$$\begin{cases} \dot{x}_b = \frac{R}{2}(V_r + V_l)\cos\alpha \\ \dot{y}_b = \frac{R}{2}(V_r + V_l)\sin\alpha \\ \dot{\partial} = \frac{R}{L}(V_r - V_l) \end{cases} \quad \text{Equation (1)}$$

Here V_r and V_l can be expressed using V and ω as equation 2:

$$\begin{cases} \dot{V}_r = \frac{2V + \omega L}{2R} \\ \dot{V}_l = \frac{2V - \omega L}{2R} \end{cases} \quad \text{Equation (2)}$$

Based on the right-hand rule (Widnall and Peraire, 2008), both linear velocity V and angular velocity ω have three dimensions x, y, z. Because the robot is expected to conduct 2-dimension navigation, only the x dimension of V, and z

dimension of ω for both right and left wheels are considered, namely linear.velocity.x, angular.velocity,z.

6.3.2 Design fuzzy logic controller

The research framework of this chapter is shown in **Figure 6.3** based on the operation flow of FLC. Four groups of parameters should be designed rationally to meet the requirements of building crack inspection work: input data, membership functions, decision-making rules, and defuzzification functions. Based on the mentioned kinetic model, linear velocity and angular velocity are used as the FLC outputs to control the speed and direction, respectively.

The designs were detailed based on the following requirements. It is expected that, controlled by the FLC designs, the robot can successfully perform “finding wall,” “wall following,” “turning,” and “obstacle avoidance” behaviors in various building scenarios, as well as avoid wavy motions and path deviations. As mentioned, the building inspection robots employ computer-vision based crack inspection techniques to realize automated crack inspection. Therefore, in the “wall following” process, the robot is also expected to move straight while keeping a desired distance from the building components to provide distinct views.

Because a behavior-based strategy is the principle for designing FLC (Cai et al., 2008), the requirements were established in accordance with manual crack inspection behaviors. In a real-world inspection case, site inspectors basically walk along the edges of buildings within an appropriate distance and examine whether the crack exist. The inspectors will come to a stop until all of the building elements have been fully inspected. In most cases, inspectors will naturally avoid frontal

obstacles such as sand piles and hydrants.

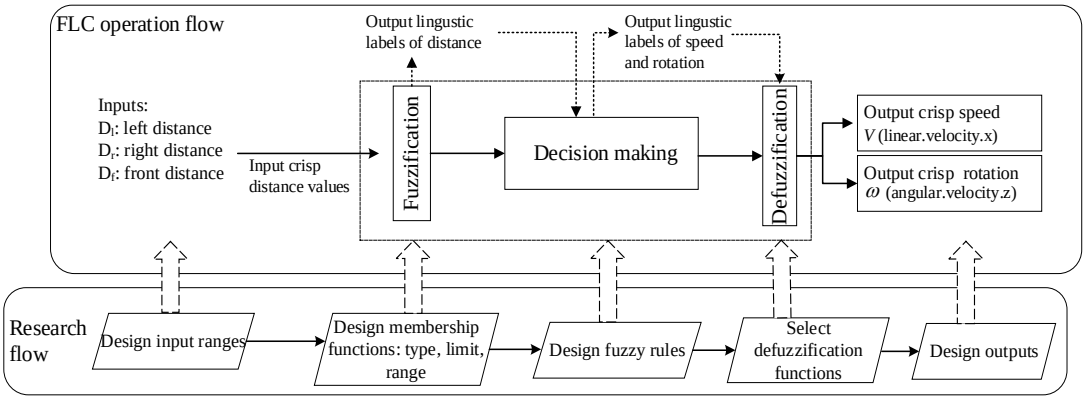


Figure 6. 3 Research framework of robotics autonomous navigation

(1) Design input datasets

Based on the principle of wall-following behavior, the left, front, and right distances between the robot and the nearby objects are employed as the input data. Several experiments were conducted to define the most appropriate ranges of the input distances. According to (Muthugala et al., 2020) and (Schiffer et al., 2012), the separate (left: -90°, front: 0°, right: 90°) and continuous ranges (left: -45°~-135°, front: -45°~45°, right: -225°~-315°) were first tested. The results showed that the robot fails to respond timely or responds too frequently when using the separate and continuous ranges, respectively. For example, when using the separate ranges, it is possible for the robot to collide with obstacles that are located obliquely ahead, such as those located at 15° instead of 0°. When using continuous ranges, it is possible to constantly alternate motion commands. For objects at the dividing lines of ranges, such as -315°, the robot is ambiguous between the “turning” command, to avoid forehead obstacles, or the “wall-following” command, to move straight, because distances of -315° belong to the “front” and “right” ranges at the same time.

Therefore, instead of separate and continuous ranges, the interval ranges were

designed for the left, front, and right distances: [150°-180°], referring to the left distance, [60°-120°], referring to the front distance, [0°-30°], referring to the right distance, which yielded optimal commands after testing. The defined interval ranges are in line with the findings in (Cherroun et al., 2019). A 360-degree laser mounted on the top floor of the testing robot was used to acquire distance data within a particular range.

(2) Design membership functions

To link the inputs to the decision-making rules, membership functions (MFs) were employed to transform the crisp inputs to their respective fuzzy degrees of each linguistic decision-making rule. Fuzzy sets (computing variables), types of MFs (computing functions), and linguistic labels (computing limits) of MFs were detailed to design the membership functions.

(i) Design of fuzzy sets

Based on the inputs and outputs, three fuzzy sets were determined as the computing variables for the designed MFs: 1) the distance set, contains the left, front, and right distances as well as their fuzzy degrees; the speed set, contains the linear velocity as well as its fuzzy degree; and the rotation set, contains the angular velocity as well as its fuzzy degree.

(ii) Design of MF types

The triangular MFs, trapezoidal MFs, and singleton MFs, which are the most extensively used MF types, were employed (Ali et al., 2015). The fuzzy degree (μ_A ,

A refers to the fuzzy set) in triangular MFs, trapezoidal MFs, and singleton MFs are calculated using equations 3 to 5, respectively.

$$\mu_A(x) = \max(\min(\frac{x-a}{i-a}, \frac{b-x}{b-i}), 0) \quad \text{Equation (3)}$$

Here, x is the input value, a and b are the lower and upper limit of the triangle, i is the average of a and b, $a < i < b$.

$$\mu_A(x) = \begin{cases} 0 & , (x < a) \text{ or } (x > d) \\ \frac{x-a}{b-a}, & a \le x \le b \\ 1 & , b \le x \le c \\ \frac{d-x}{d-c}, & c \le x \le d \end{cases} \quad \text{Equation (4)}$$

Here, x is the input value, a and d are the lower and upper limit of the trapezoid, b and c are the lower and upper support limit of the trapezoid, $a < b < c < d$.

$$\mu_A(x) = \begin{cases} 1, & \text{if } x = a \\ 0, & \text{otherwise} \end{cases} \quad \text{Equation (5)}$$

Here, x is the input value, a is the limit.

(iii) Design of linguistic labels and their limits and ranges

a. MFs for distance fuzzy set

The linguistic labels and their limits and ranges for the distance fuzzy set is shown in **Figure 6.4**.

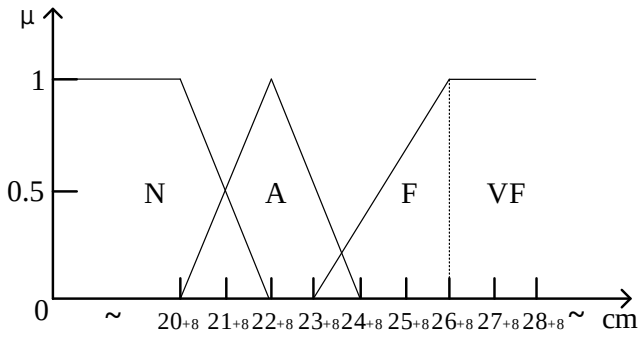


Figure 6. 4 Membership function for distance fuzzy set

The linguistic labels for the distance fuzzy set were designed to report the status of robot positions. Referring to Antonelli et al., 2007, linguistic labels were defined as N: “Near,” A: “Appropriate,” F: “Far,” VF: “Very Far” (as shown in **Figure 6.4**). A “VF” label was additionally designed to achieve the “finding wall” behavior, which enables the building crack inspection robots to locate inspection items at a fast speed and, therefore, to save time and energy.

As the building crack inspection robots employ cameras to receive defect vision, the lower limit was determined as 20 cm according to the minimum focus distance of cameras (Witt et al., 2022). If the shooting distance is closer than the minimum focus distance, blurred images may be shown because cameras are unable to focus on the subject properly. The minimum focus distance is varied for different focal lengths. In this research, we used a USB camera with a 4mm focal length as it is one of the most common camera types and has been frequently used in object recognition studies (Sagawa et al., 2004, Okazaki et al., 2008, Ren et al., 2021). The minimum focus distance for a lens with a 4mm focal length is typically around 20 cm.

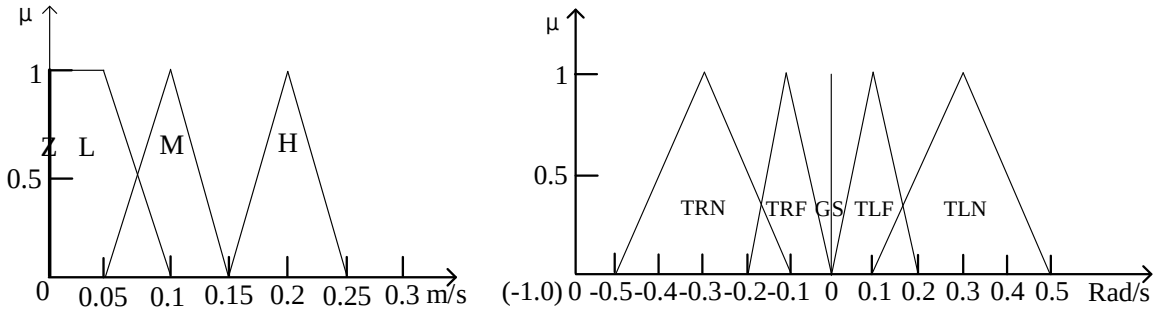
According to the inspection regulations, for example, the mandatory building

inspection scheme (MBIS), site inspectors are obliged to inspect cracks from a close distance (Chan et al., 2014). If inspectors work beyond a close distance, they may miss minor cracks. Therefore, the required close distance was employed as the upper limit. According to previous studies (Brown et al., 2001), the least distance of distinct vision of human eyes is 25 cm (Maclnnes and Smith 2010). The vision of objects' details gets blurred when the distance increases. Therefore, 25 cm was determined as the upper limit. This rule can be deemed feasible because images captured by cameras have a similar resolution to images perceived by human eyes when gazing at close objects (Skorka and Joseph 2011). Image resolution has a direct impact on how clear images are displayed (Patti et al., 1994).

For the Turtlebot 3, the inspection camera and laser were mounted at the right edge and in the centre of the robot's top floor, respectively. Because the distance MF is designed for the inspection camera (employed as a human eye) and the distance is measured starting from the laser, a supplemented distance of 8 cm, between the centre of the laser and the camera, was also considered. For different robots, the supplementary distance, the distance between the inspection cameras and the laser, can be varied. In summary, the desired distance between the robot and inspection elements is 28 to 33 cm, with a 1 cm margin of error. The ranges were fine-tuned in several experiments to realize the expected movements.

b. MFs for speed and rotation fuzzy sets

The designed linguistic labels and their limits and ranges for the speed and rotation fuzzy sets are shown in **Figure 6.5**.



MFs for the speed fuzzy set (a) MFs for the rotation fuzzy set (b)

Figure 6. 5 Membership function for speed fuzzy set

The linguistic labels for the speed and rotation fuzzy sets were designed based on the behavior-based strategy (Seraji and Howard, 2022), seen as a design principle for membership functions. To realize the expected behaviors detailed in section 3.2.3, the linguistic speed labels were designed as Z: "Zero," L: "Low," M: "Middle," H: "High" (as shown in **Figure 6-5 (a)**). The rotation labels were defined as TRF: "Turn right far," TRN: "Turn right near," GS: "Go straight," TLN: "Turn left near," TLF: "Turn left far" (as shown in **Figure 6-5 (b)**). The performances of the "L", "M", "H", "TRN", "TRF", "GS", "TLF", "TLN" labels have been proven in (Hagras, 2004) and (Lee et al., 2017), respectively. The "Z" label was additionally designed to realize the safely travel of the building crack inspection robots in special building environments, such as "turning in ground" in narrow building corners.

The limits of the speed and rotation fuzzy sets were determined based on the robot's configurations. The ranges were determined based on the relationship between the robot's turning radius, speed, and rotation (shown in Equation 6). To control the robot to slightly adjust movements (using "L" or "M" with "TRN" or "TLN") when it is near to the following objects, the turning radius is required to be within the designed distance limits (0.28m~0.34m). To achieve the "finding wall" behavior at

a fast speed (using 100% “H” with 100% “TRF” or 100% “TLF”) when the robot is far from the following objects, the turning radius is required to be around 2m, half of the average size of building rooms (Lead C, 2022), to make sure the robot can locate a building component instead of turning in the ground. It should be noted that the designed linguistic labels, limits, and ranges are feasible for most of the building inspection robots because the Turtlebot3 configurations are set at an average level. Whereas they can be adjusted for particular inspection tasks and robot specifications.

$$R = \frac{v}{\omega} \qquad \text{Equation (6)}$$

Here: R is the turning radius, v is the linear velocity, ω is the angular velocity.

(3) Design decision-making rules

Fuzzy rules are accountable for establishing decision-making logic. Modus ponens, the most essential expression of fuzzy rules (McGee, 1985), was employed to design fuzzy rules. The form of a modus ponens rule is: IF x is A THEN y is B. Specifically, x and y refer to the variables in the distance fuzzy set and the speed and rotation fuzzy sets, respectively. A and B refer to the linguistic labels of the three fuzzy sets. The t-norms (Gupta and Qi, 1991), used as an AND connector, were employed to connect the multiple conditions.

Based on the design principle: fuzzy rules are defined based on both the sensor input and the robot’s launching scenarios (Dias et al., 2018), the specific connections of x, y, A, B were defined based on the possible launching scenarios. **Figure 6.6** shows the representative sensing and launching scenarios for building crack inspection robots according to the architectural layout designs (Rahbar et al., 2022). A typical

building layout was used as a showcase. In **Figure 6.6**, the dotted lines on the robot split the obtained distance data into left, front, and right distance; the arrows refer to the expected moving directions (suppose the camera is mounted on the right side).

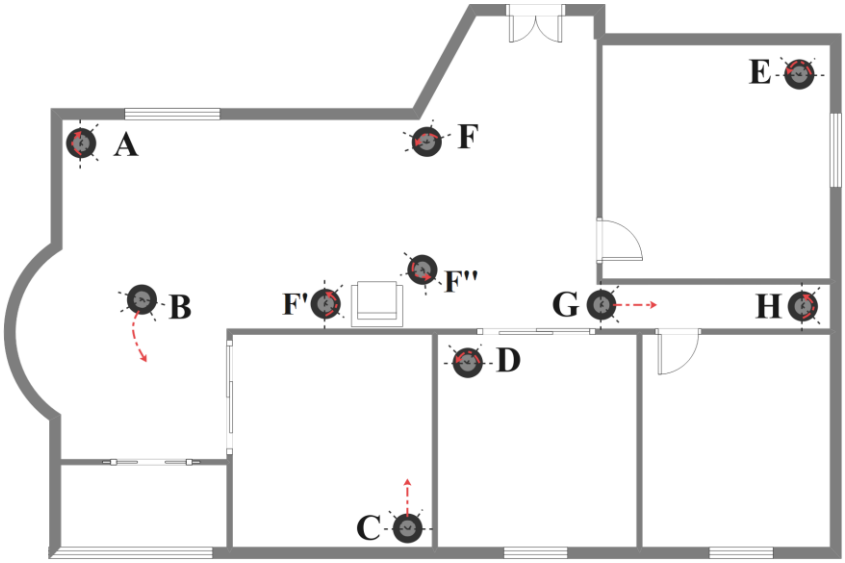


Figure 6. 6 Representative launching scenarios

The eight representative situations can be interpreted as **Table 6.1**. Avoiding forehead obstacles can be considered sub-scenarios of Scenario F, in which the robot can detect front elements. Examples are shown as F' and F''.

Table 6. 1 Interpretation of representative launching scenarios

Interpretations / Scenarios	Building element is located on			Next direction
	Left	Front	Right	
Scenario A	√	×	×	Turn right in place
Scenario B	×	×	×	Turn right
Scenario C	×	×	√	Go straight
Scenario D	√	√	×	Turn left in place
Scenario E	×	√	√	Turn left
Scenario F	×	√	×	Turn left in place

Scenario G	√	×	√	Go straight
Scenario H	√	√	√	Turn left in place

As shown in **Table 6.2**, to realize the expected behaviors, the fuzzy rules were determined by considering every possible combination of the eight representative scenarios with the designed membership functions in section 3.2.2. The modus ponens rule and t-norms were used to combine and connect the fuzzified variables. For example: IF Left is N AND Front is VF AND Right is VF, THEN Linear velocity is H AND Angular velocity is TRN. The minimum operator was used to decide rules' the entries (Hellman, 2001).

Table 6. 2 Fuzzy rules

Scenarios	Rules	Left	Front	Right	Linear velocity	Angular velocity
A	A1	N	VF	VF	Z	TRN
	A2	A			Z	TRN
	A3	F			Z	TRN
B	B	VF	VF	VF	H	TRF
C	C1	VF	VF	N	L	TLN
	C2			A	M	GS
	C3			F	L	TRN
D	D1	N	N	VF	Z	TLN
	D2		A		Z	TLN
	D3		F		Z	TLN
	D4	A	VF		Z	TLN
	D5		N		Z	TLN
	D6		A		Z	TLN
	D7		F		Z	TLN
	D8	F	VF		Z	TLN
	D9		N		Z	TLN
	D10		A		Z	TLN
	D11		F		Z	TLN
	D12		VF		Z	TLN

	E1			N	Z	TLN
	E2	VF	N	A	Z	TLN
	E3			F	Z	TLN
	E4			N	Z	TLN
E	E5	VF	A	A	Z	TLN
	E6			F	Z	TLN
	E7			N	Z	TLN
	E8	VF	F	A	Z	TLN
	E9			F	Z	TLN
	F1		N		L	TLN
F	F2	VF	A	VF	L	TLN
	F3		F		L	TLN
	G1			N	M	GS
	G2	N	VF	A	M	GS
	G3			F	L	TRN
	G4			N	M	GS
G	G5	A	VF	A	M	GS
	G6			F	L	TRN
	G7			N	M	GS
	G8	F	VF	A	M	GS
	G9			F	L	TRN
	H1	N	F	N	M	GS
	H2			A	M	GS
H	H3	N	N	N	Z	TLN
	H4	A	N	A	Z	TLN
	H5	A	N	F	Z	TLN

(4) Design defuzzification functions

After the fuzzification and decision-making process, linguistic labels of output linear and angular velocities are obtained. The defuzzification process contributes to converting the linguistic labels to crisp numbers of outputs. The most commonly used defuzzification method, the centroid method (Chakraverty et al., 2019), was employed. As shown in Equation 7 and Equation 8.

$$v^* = \frac{\sum_{i=1}^{4} \mu(v_i) \times \bar{v}_i}{\sum_{i=1}^{4} \mu(v_i)}$$ Equation (7)

Here: v^* is the crisp linear velocity, $\mu(v_i)$ is the fuzzy degree of the $i-th$ membership function in the speed fuzzy set, $\bar{v}_i$ is the centroid position of the $i-th$ membership function.

$$\omega^* = \frac{\sum_{i=1}^{3} \mu(\omega_i) \times \bar{\omega}_i}{\sum_{i=1}^{3} \mu(\omega_i)}$$ Equation (8)

Here: ω^* is the crisp linear velocity, $\mu(\omega_i)$ is the fuzzy degree of the $i-th$ membership function in the rotation fuzzy set, $\bar{\omega}_i$ is the centroid position of the $i-th$ membership function.

6.3.3 Establish optimization algorithms

(1) Heading adjust algorithm

After several tests in Robot Operating System (ROS) simulation (Mittler, 2017), it can be observed that when following the wall, the robot still moved in an S-curve rather than a straight path (scenario C), which causes difficulties for defect recognition. This may happen because the robot needed to adjust its heading timely in order to stay within the desired distance. Therefore, a heading adjust (HA) algorithm was designed to optimize the designed FLC by sending correct commands to control the robot to follow walls in a straight line.

The objective of the proposed HA algorithm is to keep the robot's heading parallel to the following elements. As shown in **Figure 6.7**, right-angled triangles with the

hypotenuse side a (distance from $30°$) and the other two sides b, c (distance from $0°$, and the following elements) are established in real-time. The HA algorithm requires the distance from $0°$ (side b) and $30°$ (side a) maintain a ration of $\sqrt{3}/2$ ($\cos 30°$), which keeps the robot's heading parallel to the following elements according to the Pythagorean theorem (Agarwall, 2020). To avoid noise, a range of -0.07 to +0.09 is adopted.

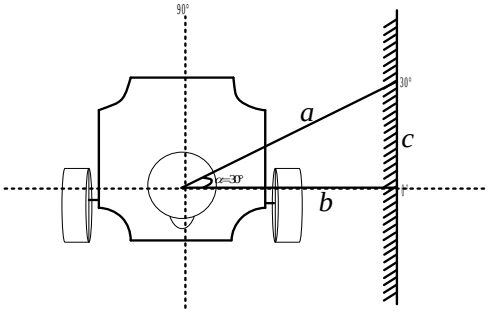


Figure 6. 7 Right-angled triangles in HA algorithm

In summary, if the ratio remains between [0.80, 0.97], the robot could follow a straight path by keeping its heading parallel to the following elements. The main concept of the proposed HA algorithm is presented below:

Algorithm 1 Heading adjust algorithm

Result: Heading (*H*)
Input: Distance from 0° *D0*, distance from 30° *D30*, right distance *Dr*
Initialization
Randomly initialize *D0, D30, Dr*
1: *dc*: 0.28, *df*: 0.33 ← lower and upper limits of right distance
2: **if** $dr \leqslant dc$ **then**
3: *H* is Turn left
4: **if** $dc \leqslant dr \leqslant df$ **then**
5: **if** $0.80 \leqslant D0/D30 \leqslant 0.95$ **then**
6: *H* is Go straight
7: else
8: *H* is Adjust heading slowly
9: ***end if***
10: **if** $dr \geqslant df$ t**hen**
11: *H* is Turn right
12: **end if**
13: **return** *H*

(2) Behavior distinguish algorithm

Another problem for the designed FLC is that the path deviation problem may happen during the “obstacle avoidance” behavior. Because of the designed fuzzy rule B, when the robot reached the end of an obstacle’s side, the “finding wall” behavior was triggered because the left, front, and right distances all belong to “VF” instead of “turning” and “following” the obstacle. Therefore, a behavior distinguish (BD) algorithm was developed to improve the FLC system by preventing path deviation.

The behind-right distance from [-135°-0°], and the behind-left distance from [180°-225°] shown in **Figure 6.8** were used to achieve this. When the left, front, and right distances are “VF”, the robot is required to first consider the behind-left/right distance. If the behind left/right distance is within 0.36m, the robot is expected to turn right/left slowly for a short distance to keep following the obstacle and return to the initial path. On the other hand, the robot is expected to speed up to find new walls. The main concept of the proposed BD algorithm is shown as follows:

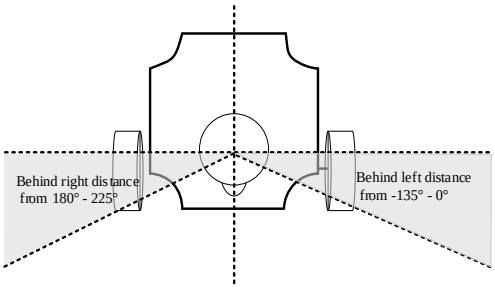


Figure 6. 8 Behind-left and behind-right distance

Algorithm 2 Behavior distinguish algorithm

Result: Moving state (*M*)
Input: Distance from 180° to 225° *br*, distance from -135° to 0° *bl*, left distance from 160° to 180° *l*, front distance from 60° to 120° *f*, right distance from 0° to 20°*r*.
Initialization
Randomly initialize *br*, *bl*, *l*, *f*, *r*
1: *VF*: very far, *A*: appropriate ← distance level

```
2: if l, f is VF and r is A then
3:     M is Following the obstacle
4: if l, f, r is VF then
5:     if br or bl <= 0.36 then
6:         M is Following the obstacle
7:     elif br or bl is VF then
8:         M is finding new elements
9:     end if
12: end if
13: return M
```

6.4 Results

6.4.1 Simulation in ROS

The feasibility of the designed FLC system was first validated in ROS simulation. The eight launching scenarios, typical and curved, square-shaped building layouts, are included in the simulation environments.

(1) Performances in various building scenarios

(i) Performance of “finding wall”, “turning”, “wall following” behaviors in eight individual building scenarios

The navigation paths, crisp values of distance, and linear and angular velocities of robot navigation in eight individual scenarios are shown in **Figure 6.9**.

It can be observed that in scenario A and D, the robot was located at the left corner with different towards. The robot properly recognized the position by computing the fuzzy degree of left, front, and right distance. In scenario A, the FLC system first output “VF”: 0.51m~ 3.5m for the front distance. “N”: 0.22m~0.27m, “A”: 0.29m~0.32m, and “F”: 0.32m~0.34m for the left distance, and “VF”:0.34m~3.5m for the right distance. In that case, the FLC system sent the angular velocity as “TRN”: -0.3rad/s and the linear velocity as “Z”: 0 m/s to control the robot to first

slowly turn right in place. When the robot properly turned its direction, the FLC system output “VF”: 0.38m~3.5m for both front and left distance. “F”: 0.31m~0.33m and “A”: 0.29m~0.31m for the right distance. In that case, the FLC system output the angular velocity as “GS”: 0rad/s, and linear velocity as “M”: 0.1m/s to control the robot to follow the wall at a normal speed.

In scenario D, the FLC system output “N”: 0.28m~0.32m, “A”: 0.28m~0.31m, “F”: 0.31m~0.34m first for the front distance. “A”: 0.28m~0.32m, “F”:0.32m~0.34m for the left distance. “VF”: 0.39m~0.66m for the right distance. In that case, the FLC system output the same velocity command as in scenario A to control the robot to first turn its heading in place. When the front and left distance changed to “VF”: 0.34~3.5m, right distance changed to “N”: 0.15m~28m and “A”: 0.30m~0.32m, the FLC system output “GS” and “M” to command the robot to follow the wall within 0.30~0.33m.

Scenario B refers to the “finding wall” behavior. In that case, the robot was located far away from the inspection elements. The FLC system sent both the left, front, and right distance as “VF”: 0.36m~3.5m. In that case, the robot was expected to turn right quickly over a long distance to find the wall as soon as possible. To achieve this, the FLC system sent the angular velocity as “TRF”: -0.1rad/s, linear velocity as “H”: 0.25m/s. When the robot found the wall, the FLC system sent the angular velocity as “TLN”: 0.3rad/s, and linear velocity as “L” and “M”: 0.05m/s~0.1m/s to control the robot to adjust its position and follow the wall.

Scenario C refers to the “wall following” behavior. In that case, the FLC system sent the fuzzy degree of the front and left distance as “VF”: 0.58m~3.5m. “N”:

0.25m~0.28m, “F”: 0.31m~0.33m for the right distance. To conduct inspection work within a desired distance, the FLC system sent the angular velocity as “TLN” and “TRN”: -0.3rad/s~0.3rad/s and linear velocity as “L”: 0.05m/s to control the robot to adjust position by turning to the left and right slowly. When the right distance was changed and kept to “A”: 0.31m~0.32m and the robot’s heading was parallel to the wall, the angular velocity was turned to “GS”: 0rad/s and linear velocity to “M”: 0.1 m/s to control the robot to follow the wall straight at normal speed.

In scenario E, the robot was located in the right corner. The FLC system sent “VF”: 0.59m~3.5m for the left distance. “N”: 0.21m~0.27m for the front distance. “A”: 0.28m~0.31m for the right distance. In that case, the FLC system sent the angular velocity as “TLN”: 0.3 rad/s and linear velocity as “Z”:0 m/s to control the robot to first turn left in place to avoid collision with forehead walls. When the front and left distance changed to “F”: 0.32m~0.33m and “VF”: 0.36m~3.5m. The right distance changed to “N”: 0.25m~0.28m and “A”: 0.28m~0.32m, the “wall following” behavior was activated. The FLC system then output the angular and linear velocity in scenario C.

When the robot launched in scenario F, the FLC system sent the same velocity command as in scenario E. The variation of the left and front distance fuzzy degree in scenario F was similar to that in scenario E. Because there were no blocks on the robot’s right side in scenario F, the FLC sent the right distance as “VF”: 0.35m~0.5m first, then changed to “N”, “A” as in scenario E.

Scenario G and H usually represent the narrow spaces in buildings. In scenario G,

the fuzzy degree of both the left and right distances was “N”:0.17m~0.27m and “A”:0.27m~0.28m, and the front distance was “VF”:0.37m~0.40m. In that case, the FLC system sent angular velocities as “GS”: 0rad/s, and “L”: 0.05m/s to control the robot to go straight slowly in narrow places. Similar navigation was conducted in scenario H. The only difference is that in scenario H, fuzzy degree of the front distance was “N”:0.23m~0.28m first, and then changed to “VF”: 0.36m~0.5m after the robot turned around. In that case, the robot was expected to first turn around slowly in place, with an angular velocity as “TLN”: 0.3 rad/s and linear velocity as “Z”: 0m/s. Then go straight slowly by changing the angular and linear velocity to “GS”: 0rad/s, and “L”: 0.05m/s. When the right or left distance was “F”: 0.31m~0.36m, the FLC system also output the angular velocity as -0.3rad/s~0.3 rad/s to control the robot to turn left or right slowly to keep following the wall within the desired distance.

The above-mentioned robot initial positions, expected behaviors, velocity commands, and changes of sensed distance in each scenario are briefly summarized in **Table 6.3**.

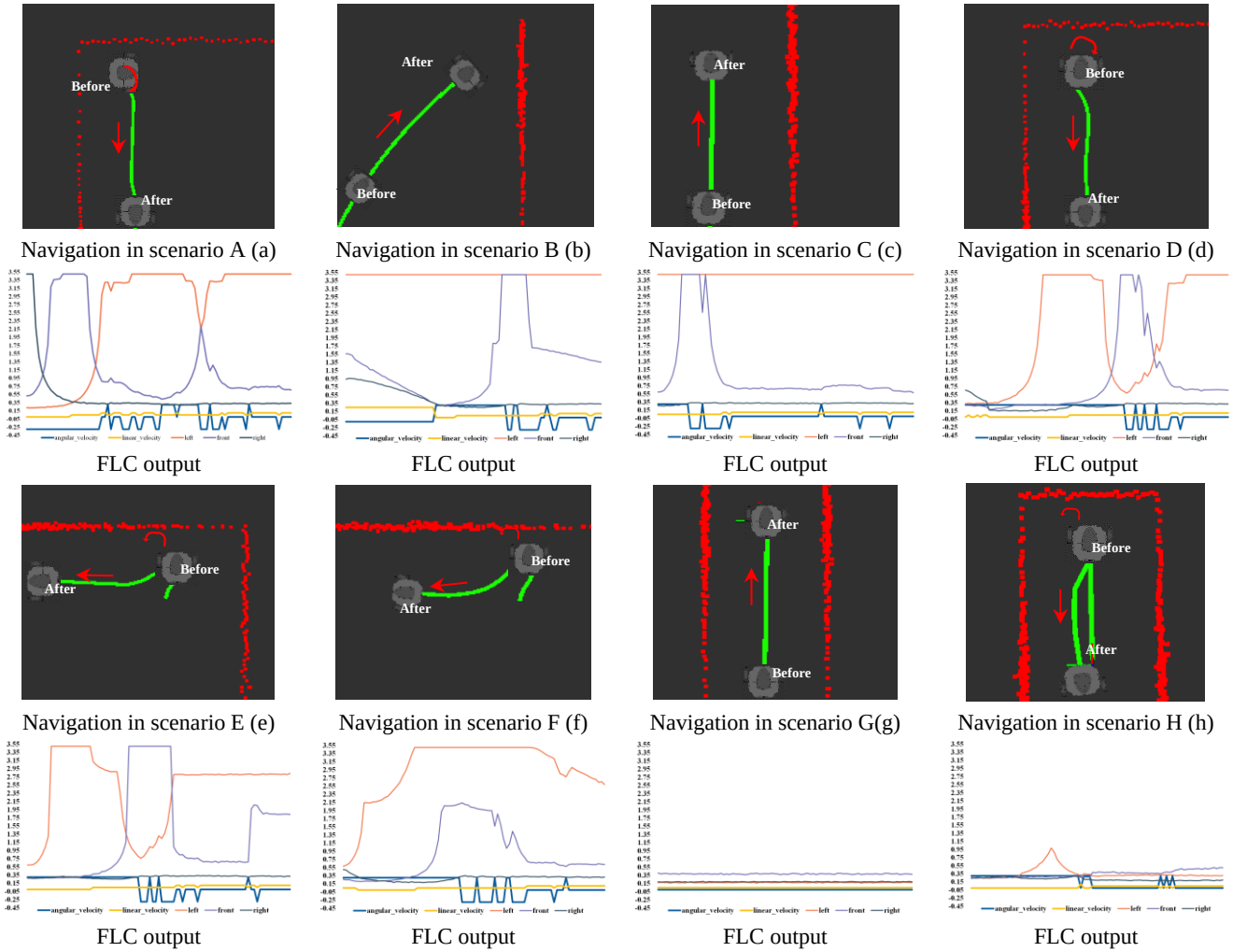


Figure 6. 9 Navigation path and variations of input distance and output velocities in the eight scenarios

Table 6. 3 Robot initial positions, expected behavior, velocity commands, changes of sensed distance in each scenario

Building scenario	Initial position	Expected behavior	Velocity command		Changes of sensed distance					
			Linear (m/s)	Angular (rad/s)	Left		Front		Right	
					Before(m)	After(m)	Before(m)	After(m)	Before(m)	After(m)
A	Left corner	Turn right in place	“Z”:0	“TRN”:-0.3	“N”:0.22~0.27 “A”:0.29~0.32 “F”:0.32~0.34	“VF”:0.38~3.5	“VF”:0.51~ 3.5	“VF” 0.38~3.5	“VF”: 0.34~3.5	“F”:0.31~0.33 “A”:0.29~0.31
B	Far away to the wall	Turn right quickly	“H”: 0.25	“TRF”:-0.1	“VF”:0.36~3.5	“VF” 0.38~3.5	“VF”:0.36~3.5	“VF” 0.38~3.5	“VF”:0.36~3.5	“A”: 0.29~0.31
C	Near to the wall	Slowly adjust and go straight	“L”: 0.05 Then “M”: 0.1	“TLN” and “TRN”: -0.3 ~0.3 Then “GS”:0	“VF”:0.58~3.5	“VF”:0.58~3.5	“VF”:0.58~3.5	“VF”:0.58~3.5	“F”:0.31~0.33 “N”:0.22~0.27	“A”: 0.31~0.32
D	Left corner	Turn right in place	“Z”:0	“TRN”:-0.3	“A”:0.28~0.32 “F”:0.32~0.34	“VF”: 0.34~3.5	“N”:0.28~0.3 “A”:0.28~0.31 “F”:0.31~0.34	“VF”:0.34~3.5	“VF”:0.39~0.66	“N”: 0.15~0.28 “A”:0.30~0.32
F	Right corner	Turn left in place	“Z”:0	“TLN”:0.3	“VF”:0.59~3.5	“F”:0.32~0.33 “VF”:0.36~3.5	“N”:0.21~0.27	“F”:0.32~0.33 “VF”:0.36~3.5	“A”:0.28~0.31	“N”:0.25~0.28 “A”: 0.28~0.32
F	Facing the wall	Turn left in place	“Z”:0	“TLN”:0.3	“VF”: 0.59~3.5	“F”:0.32~0.33 “VF”:0.36~3.5	“N”:0.21~0.27	“F”:0.32~0.33 “VF”:0.36~3.5	“VF”: 0.35~0.5	“N”:0.25~0.28 “A”: 0.28~0.32
G	In narrow places	Slowly go straight	“L”: 0.05	“GS”:0	“N”:0.17~0.27 “A”:0.27~0.28	“N”:0.17~0.27 “A”:0.27~0.28	“VF”:0.37~0.40	“VF”:0.36~0.5	“N”:0.17~0.27 “A”:0.27~0.28	“N”:0.17~0.27 “A”:0.27~0.28
H	In narrow places	Slowly turn around in place and adjust	“Z”:0	First 0.3, Then “GS”:0 and “TLN” and “TRN”: -0.3~0.3	“N”:0.17~0.27 “A”:0.27~0.28	“N”:0.17~0.27 “A”:0.27~0.28	“N”:0.23~0.28	“VF”:0.36~0.5	“N”:0.17~0.27 “A”:0.27~0.28	“N”:0.17~0.27 “A”:0.27~0.28

(2) Performance of "finding wall", "turning", "wall following" behaviors in integral building layers

As seen from **Figure 6.10**, a typical building layer with four rooms, eight external walls, and four internal walls, as well as special building layers with curve-shaped and square-shaped walls, were established to see if the robot could achieve autonomous navigation in the integral unknown building layers without collisions. The lines in **Figure 6.10 (a)** – **Figure 6.10 (c)** present the travelling path.

As seen from **Figure 6.10 (a)**, the results revealed that the FLC system successfully controlled the robot to complete navigation in typical building layers. The navigation path followed a sequence of rooms A, D2, B, C, D1, and exterior walls, covering all interior and external walls. The distance fuzzy degree, linear, and angular velocities in scenarios B, F, C, and E were combined to accomplish this. Outputs in scenario B were first used to control the robot to find the wall at a fast speed. When the robot detected the forehead walls, the outputs from scenario F were utilized to command the robot to turn left and adjust its position. Outputs from scenario C were then used control the robot to maintain following walls within a certain distance. When the robot reached the left corner, the outputs from scenario E were utilized to command the robot to turn left first to avoid collision and then continue the "wall following" behavior.

As shown in **Figure 6.10** (b) and (c), the FLC system can also control the robot to conduct crack inspection work in special-shaped building layers. In the same way, the outputs of the FLC system in scenarios B, F, C, and E were integrated. When

the robot approached the corner of a curve or a square, the FLC system would sometimes report all of the left, front, and right distances as “VF.” Different from “finding wall” cases, the robot was still located near the inspection elements, and there was no need to turn right fast to find new walls. In that case, the proposed BD algorithm assisted the robot in turning right slowly for a short distance and continuing the same inspection path.

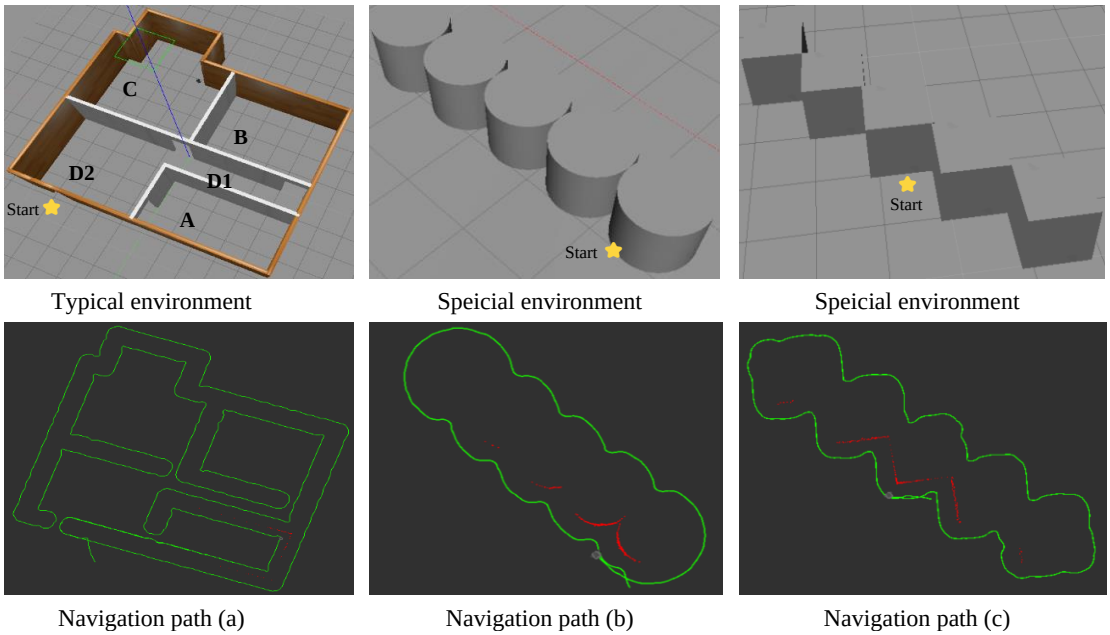


Figure 6. 10 Navigation in integral building layouts

(3) Performance of “obstacles avoidance” behavior

As shown in **Figure 6.11**, the robot successfully avoids both curved and square-shaped obstacles during the navigation process. The lines in **Figure 6.11** present the travelling path. The FLC system output distance fuzzy degrees and crisp velocities in scenarios E or F and C, respectively, to control the robot to turn first and continue “wall following.” Similarly, when the robot reached the end of one side of the obstacles, the FLC system reported all of the left, front, and right distances as “VF.” Instead of “finding wall,” the robot was expected to keep following the barriers and

return to the initial inspection path. In that case, the proposed BD algorithm also assisted in avoiding path deviations.

Specifically, when the robot moved to the end of one side of an obstacle, the distance from three directions was all rated as “VF,” as shown in **Figure 6.12 (c)**. The “finding wall” behavior was then triggered, causing the path deviation. Apart from reporting the left, front, and right distances as “VF,” the FLC additionally reported the “right-behind” distance after applying the BD algorithm, as shown in **Figure 6. 12 (d)**. According to the BD algorithm, the FLC system reported velocities as “TRN” and “L” instead of “TRF” and “H” when the “right-behind” distance was less than 0.36m. In that way, the FLC system controls the robot to keep the following behavior by constantly turning right slowly over a short distance rather than turning right fast over a large distance to find new walls. The green lines in **Figure 6.11** present the travelling path.

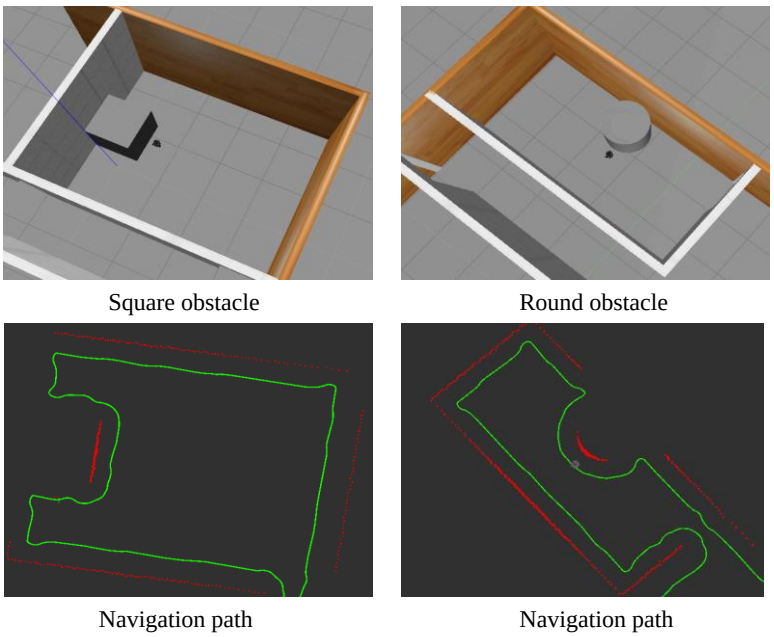
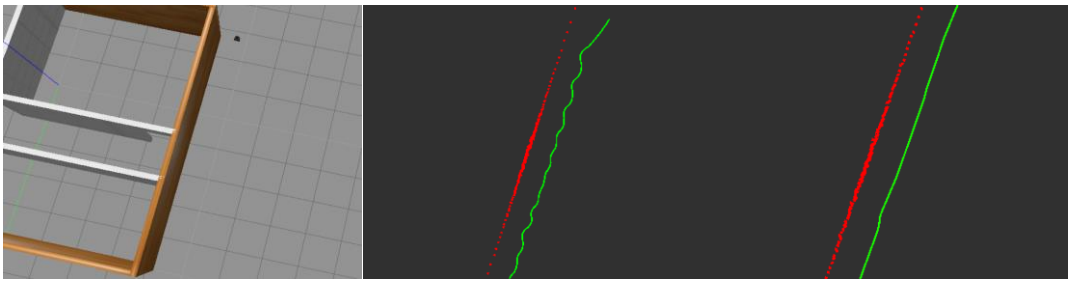


Figure 6. 11 Obstacle avoidance

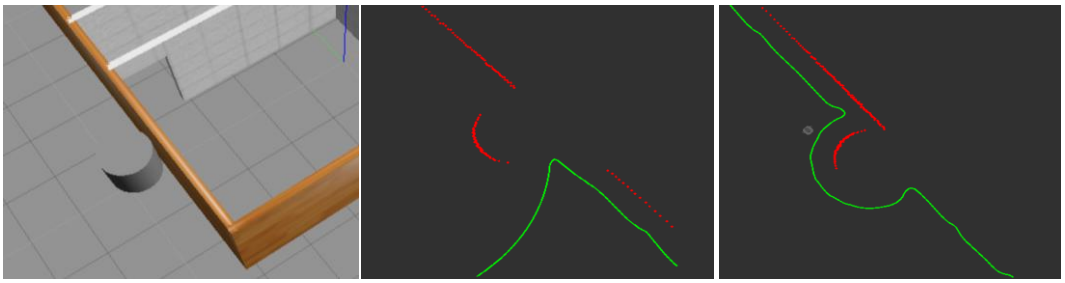
(3) Performance of the designed HA and BD algorithm

It should be noted that the robot may turn too often during the “wall following” stage to maintain a certain distance, as seen in **Figure 6.12 (a)**. The HA algorithm assists in providing the robot with a straight wall following path. Specifically, the FLC system sent the velocities “TRN” or “TLN” and “L” initially to command the robot’s left or right turn. According to the HA algorithm, instead of continually adjusting orientations, the FLC system gives “GS” and “M” commands to tell the robot to go straight without turning at a normal speed when its heading is parallel to the following elements.

It’s also worth noting that the FLC system is experimented to output velocity as “GS” and “Z,” to control the robot to turn straight while stationary. Although the robot’s heading may be adjusted more precisely in that way, it tends to stop and move constantly, which also causes camera shake. Therefore, adjusting direction at a slow speed is a better option. After using the proposed HA algorithm to optimize the FLC system, it is clear that the robot could move in a relatively straight path in the “wall following” stage, as shown in **Figure 6.12 (b)**. The lines in **Figure 6.12** present the travelling path.



Simulation environment | Path without HA algorithm (a) | Path with HA algorithm (b)



Simulation environment | Path without BD algorithm (c) | Path with BD algorithm (d)

Figure 6. 12 Navigation path with and without HA and BD algorithm

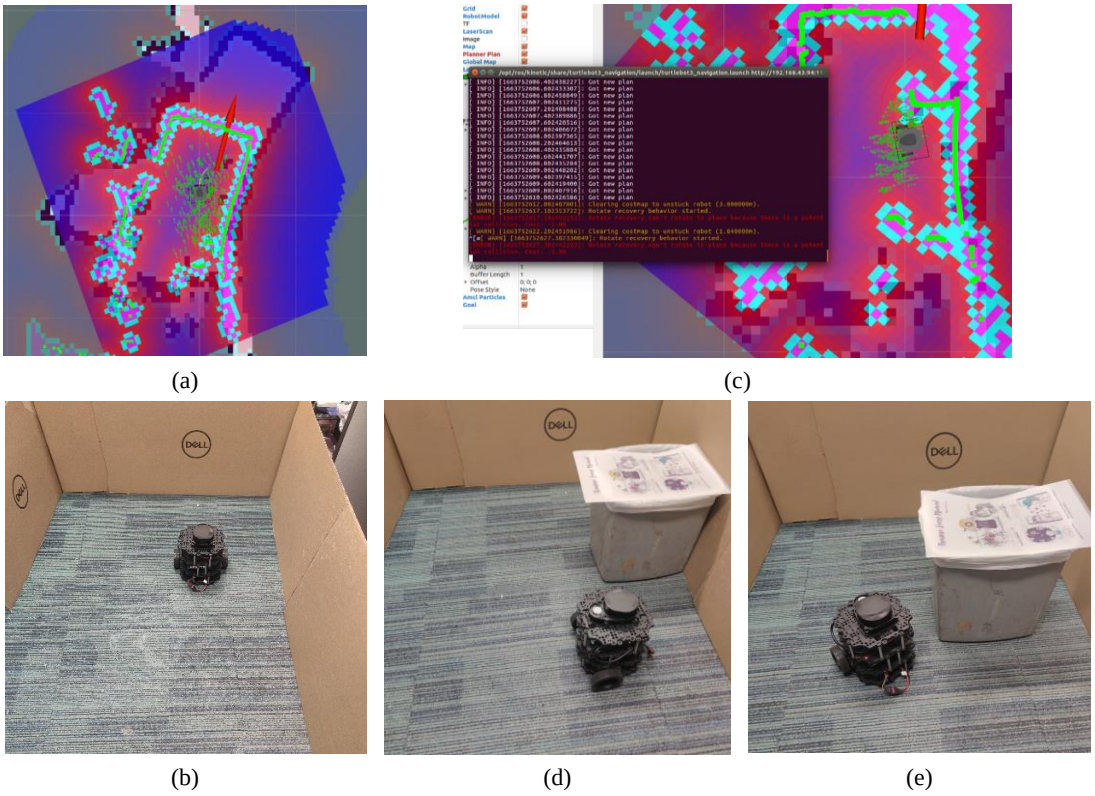
6.4.2 On-Site Validation

On-site validation was conducted inside the Hong Kong Polytechnic University to validate the designed FLC in real-world environments.

(1) Feasibility and efficiency in dynamic environment

To validate the advantages of the local navigation strategy in unknown environments, discussed in the literature review section, the performance of the designed FLC and the SLAM algorithm (a global navigation strategy) was compared. **Figure 6.13 (a)** shows the initial map prepared for SLAM navigation. As can be seen, by navigating using the initial map, the robot successfully reached the goal position (**Figure 6.13 (b)**). However, when placing a box obstacle later, the robot was blocked (**Figure 6.13 (d)**) because the environmental map was not updated in real-time and the SLAM algorithm failed to calculate new path (**Figure 6.13 (c)**). Differently, when navigated using the designed FLC, the robot can

successfully pass obstacles that are placed at any time and reach the goal position without collision (**Figure 6.13 (e)**).



(a) (c)

(b) (d) (e)

Figure 6. 13 Comparison of SLAM and the designed FLC

(2) Feasibility and efficiency in irregular building scenarios

Special locations, including concave and convex regions, curve-shaped columns, and narrow aliases, were selected to highlight the robustness of the designed FLC.

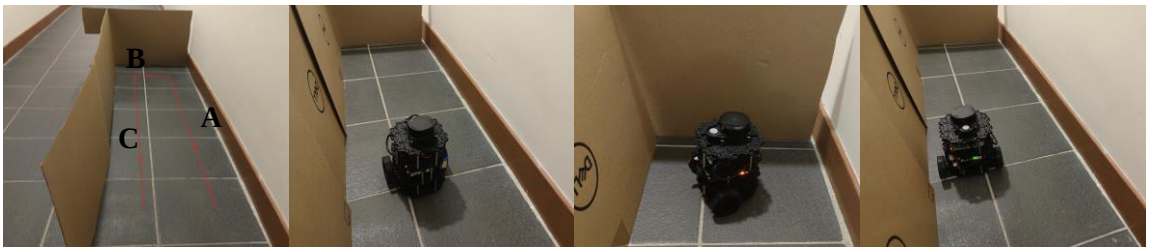
As seen from **Figure 6.14**, it was validated that the designed FLC successfully controlled the robot to navigate in concave and convex regions without collision. As expected, the robot firstly conducted “wall following” behavior using the velocities in scenario C. Velocities in scenario E was then triggered to control the robot to turn left first and continue following the wall. When the robot moved to the convex region, velocities in scenario B were sent and the designed BD algorithm

was activated to control the robot to adjust its heading without deviating from the path.

Validation environment A Region B Validation environment B Region B

Figure 6. 14 Navigation in concave and convex region

As seen from **Figure 6.15**, the FLC system was proven to be suitable for navigation in narrow regions and curve-shaped columns. When the robot was close to both the right and left walls, velocities in scenario G were delivered to control the robot to slowly move straight. When the robot reached the end of the narrow regions, velocities in scenario H were delivered to control the robot to turn around in place and then continue wall following. When the robot met curve-shaped columns, velocities in scenario C and B were sent alternately to control the robot's movement in a curving path. The BD algorithm helped to avoid path deviation.

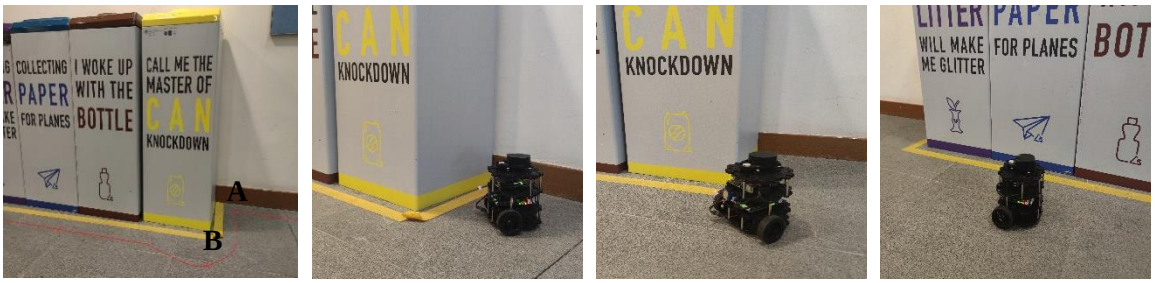


Validation environment B Region A Regin B Region C

Validation environment C Region A Regin B Region C

Figure 6. 15 Navigation in narrow regions and curved columns

As seen from **Figure 6.16**, it is validated that the FLC system successfully controlled the robot to pass through the forehead obstacles. The robot started with the "wall following" behavior first. The designed HA algorithm helped to control the robot's movement straight and keep a desired distance. Similarly, when the robot met forehead obstacles, the navigation strategies in scenario E, B, and C were activated respectively to make the robot avoid the obstacles and keep following the wall.



Validation environment D Region A Regin B Region C

Figure 6. 16 Passing forehead obstacles

In summary, the designed FLC system is validated as feasible for the wall-following navigation of building crack inspection robots in various unknown building environments. To achieve this, the basic "finding wall", "wall following," "turning,"

and “obstacle avoidance” behaviors can be realized, which are correlated to the findings in (BraunstingI et al., 1995, Nadour et al., 2019).

The findings outperform the existing algorithms from: 1) The designed FLC is suitable for various unknown launch situations. The robot can navigate properly in eight different launch scenarios, including narrow spaces and building corners. If the robot is launched far from the inspection elements, the designed “finding wall” behavior enables the robot to quickly locate the inspection region. 2) Optimized by the proposed HA algorithm, the designed FLC ensures a relatively straight wall-following trajectory and keeps the robot following within a desired distance. 3) Optimized by the proposed BD algorithm, the path deviation problem can be effectively avoided for complex environments, such as concave and convex regions, curved or square-shaped building elements.

6.5 Summary

In summary, this chapter proposes an autonomous navigation strategy for the robotic control system to control the motions of the building crack inspection robot. To do this, a wall-following navigation, empowered by a novel fuzzy logic controller, was designed. The designed FLC enables robots to conduct basic crack inspection behaviors: “finding wall”, “wall following”, “turning” and “obstacle avoidance” without referring to prepared environmental maps. The designed FLC is robust, it ensures the safe travel of the robot in various building scenarios. Adjusted by the proposed optimization algorithms, the FLC provides a straight following path within a desired distance. Meanwhile, the path deviation problem can be effectively addressed.

In the FLC system, the inputs are the left, front, and right distances within the designed interval ranges: [150°-180°], referring to the left distance, [60°-120°], referring to the front distance, and [0°-30°], referring to the right distance. The outputs are the angular and linear velocity. Three fuzzy sets (distance, speed, and rotation fuzzy set) and membership functions were established based on robot configuration, camera configuration, building scenarios, and building inspection criteria. 45 fuzzy rules are defined for the robot's decision-making based on every possible sensing and launching situation. In the fuzzification and defuzzification processes, the crisp distance data and crisp velocity data, as well as their linguistic fuzzy degrees, were interchanged considering the fuzzy sets and fuzzy rules. Two optimization algorithms were also proposed based on the Pythagorean theorem and the distances between the behind-right and left ranges.

The FLC system was validated in both simulation and real-world environments using the Turtlebot3 burger robot. It is validated that the designed FLC realizes the autonomous navigation of building crack inspection robots in unknown building environments. It is feasible to control robots to conduct crack inspection motions in eight different building scenarios, such as, building corners, narrow aisle. By integrating the output velocities of the eight individual scenarios, the robot successfully navigated in integral typical, curved, and square-shaped building layers and avoided collision with forehead obstacles. The proposed HA and BD algorithms effectively assisted in generating straight wall following paths within a desired distance and avoiding path deviation in complex regions, such as, concave and convex regions.

In the following chapter, the details of designing a web-user interface that realizes the functions of smooth, remote, continuous, and real-time visualization are demonstrated. Navigated by the designed FLC system, the robot is expected to automatically follow the building components and continuously display the crack inspection outcomes, which are processed using the designed lightweight CNN, to the web-user interface.

Chapter7: Develop a Web-User Interface for Robot Visualization

7.1 Introduction

Except from automated inspection and autonomous navigation, providing users with a friendly interface that visualizes the inspection data, including the video stream and inspection outcomes, is one of the major fields of robotics. This chapter aims to establish a user-friendly web interface for the visualization of the crack inspection process. There are five sections in the developed web user interface: 1) the date section, which aims to notify inspectors of the inspection date; 2) the title section, which aims to provide a clear understanding of the theme; 3) the inspection visualization section, which aims to demonstrate the inspection video streams and outcomes in real-time; 4) the video recording section, which aims to allow users to start recording and downloading the inspection videos for historical checking and tracing; 5) the contact information section, which presents the basic information of the researchers.

To develop the web user interface, the Hypertext Markup Language (HTML) and cascading style sheets were employed to present the front-end elements and format their styles. The JavaScript plugin was employed for the backend coding to realize the functions of CNN-based automatic crack inspection and video recording and downloading. The responsive web design (RWD) technique was used to allow for the adaptive adaptation of the webpage's contents to the user's screens. Details of the development of the frontend and backend of the web user interface are presented in sections 7.2 and 7.3, respectively.

7.2 Methods

7.2.1 Fundamental functions

A systematic literature review was conducted to identify the fundamental functions of crack inspection robotic interfaces. Fruitful knowledge in scientific literature has been widely referred to as the supporting basis for the development of advanced technologies because science provides fresh perspectives, tools and techniques, instruments, and benchmarks for technological possibilities and engineering designs (Brooks, 1994). Systematic literature review demonstrates interpretations, summaries, and discussions of critical information from published literature. Different from traditional literature reviews, systematic literature reviews present a balanced and unbiased analysis of existing literature because they examine all positive and negative research published in both low and high-impact journals (Nightingale, 2009).

During the review process, the literature was searched in the most popular academic literature database, Scopus. Generally, Scopus and Web of Science (WoS) are the most widespread and frequently used databases for searching literature. Both of them cover scientific literature from a variety of disciplines, including mechanical and engineering. The earliest publications indexed in Scopus and WoS trace back to the 1960s. Scopus instead of WoS was selected because Scopus covers a broader journal range, including journals, book series, conference proceedings, and open access journals, which contributes to more comprehensive literature search results (Chadegani, 2013).

The search string (TITLE-ABS-KEY (inspection AND robot) AND TITLE-ABS-

KEY (interface)) AND (LIMIT-TO (DOCTYPE, “cp”) OR LIMIT-TO (DOCTYPE, “ar”) OR LIMIT-TO (DOCTYPE, “ch”)) was initially searched to locate crack inspection robot interface-related literature in Scopus. In this round, 398 literatures from the years 1979 to 2022 were given. Multidisciplinary topics including engineering, computer science, mathematics, and more were covered. Only the three primary publishing types—conference papers, journal articles, and book chapters—were included in the documents. The conference review, review, short survey, and book were excluded because they provided less original and specific content.

The 398 pieces of literature were filtered by reading their titles and abstracts thoroughly. In the filtering process, 18 publications that are not relevant to interface designs and inspection robots were excluded. Interfaces that go beyond the engineering field were also excluded because they have less reference value for designing building inspection interfaces. For example, (Kyrkjebø, E. et al., 2009) proposed a “temperature view” function to demonstrate temperature inspection on oil platforms. After filtering, the entire content related to interface designs in the remaining literature was examined.

In addition, the specific keywords of “building”, “crack”, “pipeline”, “bridge”, “road”, “tunnel”, and “steel” that related to the building crack inspection were searched again, respectively, in Scopus to target relevant literature as completely as possible. Contents related to interface designs in the newly searched literature were intensively read. Finally, 22 publications that focused on developing interfaces for building crack inspection robots and ranked highly in terms of citation numbers

were recorded.

In total, twelve different interface functions can be observed after listing the interface functions developed for building crack inspection from the 22 qualified publications, including: showing map view, showing camera view, showing connecting state, showing task information, providing inspection record, enabling robot movement control, providing security login, enabling autonomous navigation, showing robot speed, showing battery state, showing environment state, showing motor state. Detailed explanations are shown in **Table 7.1**.

Table 7. 1 Explanations of interface functions

ID	Functions	Explanations
F1	Showing map view	The map view window is used to demonstrate the navigation maps, including robot positions, headings, and surrounding environments. The maps are shown in either 2D or 3D form.
F2	Showing camera view	The camera view window is used to demonstrate inspection videos in real-time.
F3	Showing connecting state	The connecting state box is used to demonstrate whether the interface has successfully connected with the robot control system.
F4	Showing task details	Task dates and inspection tasks, including building floors and room numbers, are listed in the task details box.
F5	Providing inspection record	The record button is used to show historical inspection videos.
F6	Enabling robot movement control	Users can guide robots to move in any direction, including forward, backward, right, left, and stop, using the movement control buttons.
F7	Providing security login	On the login page, users can fill in their usernames and passwords to protect their information.
F8	Showing robot speed	This function displays the speed of a robot's movement.
F9	Showing battery state	This function displays the battery's charge.

F10	Showing environment state	This function is used to monitor environmental factors, including temperature and humidity.
F11	Showing motor state	This function displays the motor states of the robot, such as impulse signals.

One of the essential features is the F2 “camera view” function. There were 17 occurrences of it throughout the 22 publications. The “motion control” function then made an appearance 15 times. Five interfaces are available for reporting F10 “environmental conditions,” F9 “battery state,” and F1 “map display.” Other functions were mentioned less than five times overall. The occurrence of reporting F3 “connecting state”, F8 “robot speed”, and F4 “task information” is no more than three times. The functions of F5 (“inspection record”), F7 (“security login”), and F11 (“motor state”) are the least frequent; they are only presented in one or two interfaces. As a result, F2, “camera view,” was selected as the target function when developing the present interface for visualizing building crack inspection process. It was expected that the interfaces would be able to 1) smoothly, constantly, and remotely display the video stream as well as the inspection outcomes on the webpage. 2) Providing customers with the ability to record and store inspection videos for the purpose of historical tracing.

7.2.2 Design and coding

(1) Front End development

The responsive web design (RWD) framework was employed in this research to develop the inspection interface, making it workable in a wide range of real-world inspection scenarios. The W3C defines RWD as the process of automatically scaling, hiding, shrinking, and enlarging a website’s contents in order to make web

pages look acceptable on all devices, including desktop computers, tablets, and phones (Gardner, 2011). When working at building sites, for instance, the inspectors can use their mobile phones to view the real-time inspection videos. When holding a negotiating meeting, the multiple stakeholders can share the inspection results on the desktop computers in the meeting room.

Responsive web design was implemented using Bootstrap framework, which was developed as the best toolkit for developing mobile-first webpages (Wehrens and Buydens, 2000). The Bootstrap framework, initially known as the Twitter Blueprint and developed by Mark Otto and Jacob Thornton at Twitter, is used to enhance the consistency of different web development tools including the Hypertext Markup Language (HTML), the cascading style sheets (CSS), and JavaScript plugins. By containing different syntax, Bootstrap offers a quicker and simpler solution for webpages to automatically recognize the user's screen and adapt the visualization properly. Additionally, the front-end framework provides comprehensive, pre-made templates and elements such as typography, forms, buttons, and videos that assist in the development of the intended functions of "camera view" and "video recording". Bootstrap version 5, which was published on May 5, 2021, was employed in place of Bootstrap versions 3 and 4 (Bootstrap 5 stable, 2022), as it offered an improved grid system, a lighter package, and enhanced backward compatibility with JavaScript plugins. The Internet Explorer (IE) 10 and 11 browsers are not supported by Bootstrap 5, however the majority of current browsers, including Firefox, Safari, and Chrome, as well as Android and iOS devices supported.

For the building crack inspection interface, five major sections were built using the RWD and Bootstrap frameworks: the date section, the title section, the inspection visualization section, the video recording section, the contact information section, and then the inspection visualization section. The primary goals of "visualizing camera videos" and "video recording," were achieved in the inspection visualization and video recording sections, respectively. The five designed sections are interpreted as follows:

1) The date section presents the time information to notice the inspection date for users. The date of a specific time zone can be presented on the webpage and updated dynamically by using the "new Date()" function and returning the date object as a string using the "toLocaleDateString()" function.

2) The interface's topic is presented in the title section, which provides the intention noticeable.

3) In the inspection visualization section, the camera view as well as the inspection results are demonstrated. The camera view is presented in the form of a video stream, and the inspection results are shown below the video in the form of text strings, such as "Results: 100% Crack" or "Results: No Crack." By computing the video stream with the developed lightweight CNN algorithm, coded in the backend of the interface using the JavaScript plugin, the inspection results and the prediction probability change automatically.

4) The video recording section contains two buttons that allow users to record and download the video stream. Users can start and stop the video recording whenever

they like by adding the “mediaRecorder” function to the buttons in the JavaScript plugin on the backend. Direct downloads of the video feeds are available from the webpage.

5) The contact information section includes the researchers' basic information, such as affiliations and names. By clicking the links given in the “names” string, users can locate the ResearchGate page of the researchers, which is attached to the "names" string using the <a>… element. The ResearchGate page records background information about researchers such as email addresses and research work.

To make it easier for users to notice the most important information, the five sections stated above were arranged using a grid layout in a “clean and simple” manner. The title, inspection visualization, and video recording portions were positioned in the centre of the interface to draw attention to the topic and contents, while the date and contact information sections were positioned at the top and bottom. Based on the web design principles of balance, symmetry, and continuity (McClurg and Joshua, 2005), the whole contents are arranged along the page axis and in horizontal symmetry. According to the psychologist Paul Fitts, who found that the average duration of human visual responses is related to not only the distance but also the size of the target, the sections of title, inspection visualization, and video recording are also designed in big font size, while the left two sections are presented in small font size. Black and white color matching was used to express to users the clean, pure, noble, stable, and high-tech emotions (Luo et al., 2021). The interface sketch is shown in **Figure 7.1**.

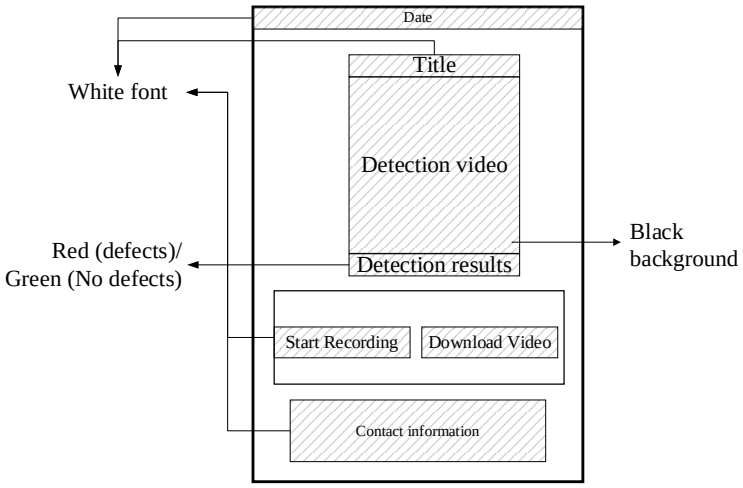


Figure 7. 1 Interface sketch

The standard web development markup language HTML was employed to structure the five sections mentioned. The five components described above were organized using the HTML markup language, which serves as the mainstream technology (Berners-Lee and Connolly, 1995). Using various HTML elements, the whole contents of the five sections is presented from the sketch to the webpage. Firstly, the … element was coded in the <body>…</body> element to draw the general containers for the five sections. A <script>…</script> element was coded in the "date" container. The Date().toLocaleDateString () function is included in the script element to present the local date. In the second element, a … element was used to present the interface title. The <video>…</video> element was included in the third to demonstrate the real-time video stream and CNN-based inspection outcomes. Two <button>…</button> elements were used to provide the buttons of start and save video to realize the functions of "start recording" and "download video", respectively. … and <a>… elements were used in the fifth to present the contact information. The <a>… element links the ResearchGate webpages. In the end, a

<script>…</script> element was coded to link the backend JavaScript to realize the real-time demonstration of CNN-based crack inspection outcomes.

In order to change the styles and formatting of the HTML elements, such as the background color, text font family, font size, color, button style, button hover, and the locations of each container and its included elements, such as align to the center, cascading style sheets (CSS), the basic language to format the webpages (Lie and Bos, 1997), were used. The CSS file was attached into the HTML file by means of the <link>.../link> element.

(2) Adapting display area

The visible area of a user's webpage is referred to as the viewport by the W3C. Varied devices, such as tablets and smartphones, have different screen sizes and resolutions. To make the designed interface applicable for both desktop and mobile devices, it is necessary to design the adaptive webpage and properly display the contents of the webpage in the visible area of the various devices. Due to the fact that desktop computer screens are usually bigger than those of mobile devices, the contents of the webpage could get blurry or blocked if the content size is fixed.

The responsive web design solves the problem by automatically rescaling and resizing the text, elements, images, and videos depending on the viewport dimensions. Both the media query and viewpoint units in RWD assist in the automated rescaling and resizing process. When using the media query, the breakpoints, which refer to the maximum width and height of the screen, are first determined. The size and scale of the webpage contents can be adjusted

automatically based on the self-defined width percentage within the breakpoints. For example, after setting the “column-1” element with 100% width at a breakpoint of 768px, the width of the “column-1” element will rescale to the device screen width, which is within 768px, regardless of whether its width on the computer screen is bigger, such as 1200px, or smaller, such as 576px. Defined by the Bootstrap, the breakpoints contain five levels: extra small (<576px), small (>=576px), medium (>=768px), large (>=992px), and extra large (>=1200px). Developers set the breakpoints according to the display requirements and device configurations. For example, the viewpoint of “Extra small” (<576px) is selected if the web page is designed to adapt to the iPhone 14, which has a 6.1-inch screen with a screen resolution of 390px. In that scenario, however, the contents may not adapt well if the screen resolution exceeds 576 pixels.

Differently, the viewpoint unit technique directly adjusts visualization scale and size without the limitations of breakpoints. By setting the four viewport-based units, vh, vw, vmin, and vmax, for each element in CSS, the width and height of the elements can be adjusted accordingly based on the layout viewport of the browser. Vh, vw, vmin, and vmax units represent viewport height, width, minimum, and maximum viewport, respectively. For example, by setting the vh as “50vh” and the vw as “50vw” to a particular element in the Chrome browser of a mobile device, the height and weight of the element automatically turn to 640px ×124px, 50% of the Chrome layout viewport, 1280px × 619px. The vmin and vmax units are based on the smaller and larger dimensions of the layout viewport, respectively. For example, by setting the height of an element as “50vmin,” the height equals 50% of the minimum size between height and width.

In the present research, the viewpoint unit technique was employed to adapt the webpage to as many user devices as possible. To better align the contents of the five sections to the center, the widths of the five sections are set to “100 vmax.” The height of the inspection visualization section was set at “70 vmin,” in particular to maintain the distribution scale of the sections designed in **Figure 7.1**.

7.2.3 Back End development

The backend coding of the web interface was established to realize the functions of CNN-based automated crack inspection and real-time video recording. The most widely used web programming language, JavaScript, was employed (Gardner and Smith, 2012). PHP and JavaScript are seen as the two most popular programming languages for web development. Although PHP has advantages compared to JavaScript, such as being easy and simple to use and having lightning speed (Cullen, 2022), JavaScript was selected because it supports a rich set of APIs, including TensorFlow.js, which was used to connect the CNN model to the webpage. No APIs for tensorflow.js are provided in PHP yet.

As mentioned, the designed inspection visualization section is responsible for capturing the real-time video stream and automatically recognizing whether the cracks are on the captured photographs. To do so, the web API, Media Capture and Streams API, was used to provide the video stream captured by a USB camera connected to the local computer. Particularly, the video stream is generated using the MediaDevices.getUserMedia () function. The audio was set as false because the cracks are inspected by processing the photographs with the lightweight CNN model instead of the sound information.

To make the developed lightweight CNN model applicable on the webpage, the open-source JavaScript library TensorFlow.js and its high-level layers API were employed. TensorFlow.js, which is powered by WebGL, includes a high-level layer API for defining and training machine learning models in the browser, as well as automatically importing pretrained models saved in the TensorFlow SavedModels and Keras hdf5 formats. **Figure 7.2** shows the steps of using TensorFlow.js to process CNN models in webpages. To be specific, the CNN model was first trained using the TensorFlow library and saved in the SavedModel format. The TensorFlow.js model, which contains the.bin files and the model.json file, was converted using the Tensorflowjs.converts.converter library to meet the requirements of the JavaScript function, tf.loadGraphModel (), that used to load the machine learning model to the webpage. The TensorFlow.js model was uploaded to a GitHub account to obtain its URL address. The model's URL address was then typed into the tf.loadGraphModel (model_url) function to upload the CNN model to the webpage to automatically inspect the cracks in the video stream.

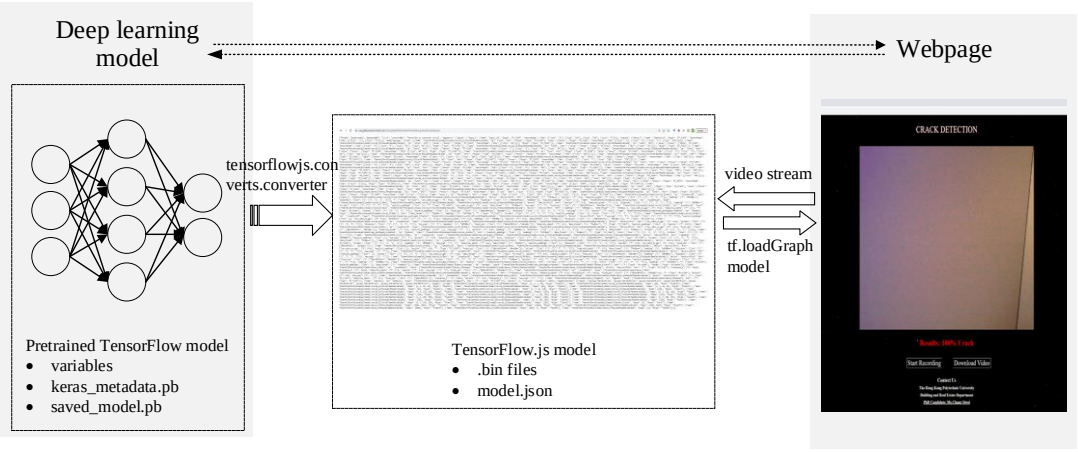


Figure 7. 2 Using TensorFlow.js to operate CNN models in webpages

The web API, MediaRecorder, was employed to enable the webpage to record the captured video streams. An on-click event that was included in the established media recorder function was given to the “Start Recording” and “Download Video” buttons to enable users to start recording and downloading videos by easily pressing the buttons on the webpage. The built-in video recording and downloading JavaScript code provides the functions of saving inspection videos directly on the webpage. The downloaded videos can be saved as mp4 files.

7.3 Results

Figure 7.3 displays the developed web-user interface. As can be seen, the five sections mentioned above can be basically accomplished. First, in the center of the webpage, users could see live video streams and crack inspection results such as “Results: No Crack” and “Results: a% Crack” (a is the possibility that the object is a crack). The “Start Recording” and “Download Video” buttons are located below the inspection results and can be used to record and download the video stream. Meanwhile, the date, title, and contact details are given at the top and bottom, respectively. The date information changes over time according to the local time. Contact information, such as email addresses, can be acquired by clicking the link listed at the bottom of the webpage. Development details of the CNN-based crack inspection section and the video recording section can be found below.

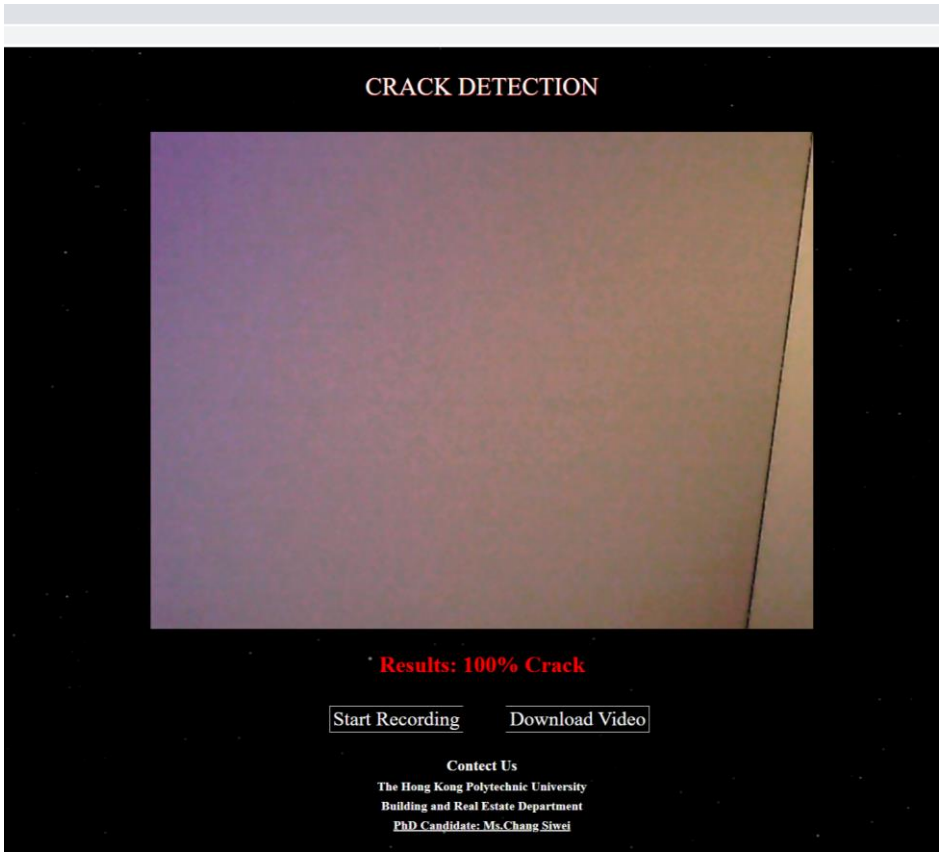
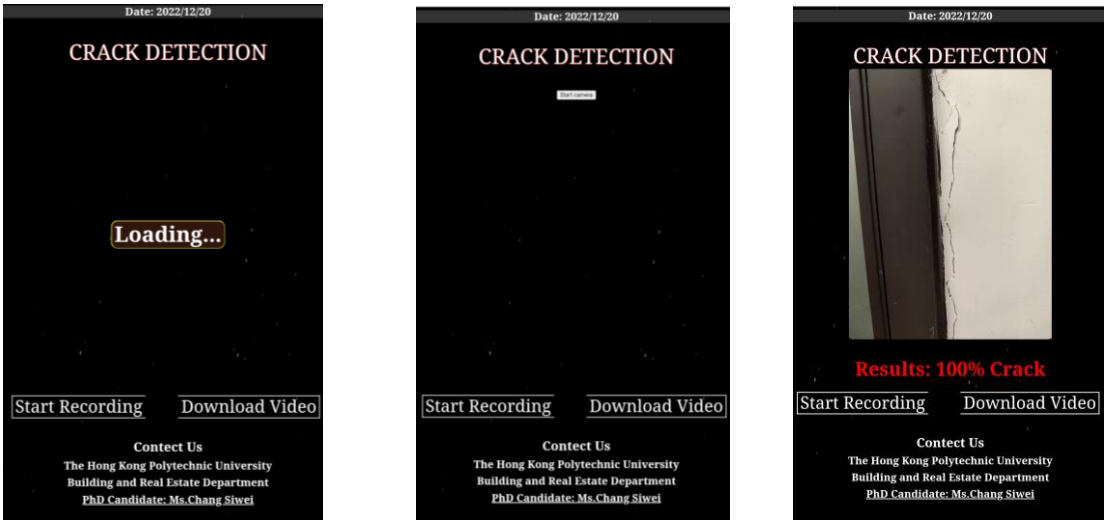


Figure 7. 3 Draft version of the developed web-user interface

7.3.1 CNN-based crack inspection

This section provides the development process and outcomes of the real-time video stream visualization and the CNN based automated crack recognition. After entering the interface link, the pre-trained lightweight CNN model is first loaded from a particular URL that saves the CNN model (.json file). As shown in **Figure 7.4**, a button of "Loading…" is shown in the video stream window initially to demonstrate the state of loading the CNN model. The button will then be updated to "Start camera" when the CNN model is successfully loaded. By clicking the "Start camera" button, the real-time video stream as well as the crack recognition outcomes will be demonstrated in the center of the webpage.



Before | After | After clicking "Start camera" button

Figure 7. 4 Before and after CNN model loaded

As shown in **Figure 7.5 (a)**, lines 19–22 of the code, the crack inspection section was distributed to a … element, which contains <button>…</button> and <video>…</video> elements to present the "Loading…", "Start camera", and "video stream" states, respectively. Three id selectors, "#liveView," "#webcamButton," and "#webcam" were given to each HTML element to format its cascade features. For example, to list the buttons, video stream, and inspection results in columns and align them to the center, the css properties of display, flex-direction, margin-left, margin-right, vertical-align, justify-content, and align-items in this div division were set as flex, column, auto, auto, middle, center, and center, respectively. Meanwhile, the viewport units of 100 vmax and 70 vmin were set for webpage width and height, respectively, to adapt the web page to different mobile devices.

```
19          <div id='liveView'>
20              <button id="webcamButton" class="invisible">Loading...</button>
21              <video id="webcam" class="background" playsinline crossorigin="
                anonymous"></video>
22          </div>
```

(a) Crack inspection section

```
25          <div class ='buttons'>
26              <div>
27                  <button  id='arrow-L'>Start Recording</button>
28                  <button  id='arrow-R'>Download Video</button>
29              </div>
30          </div>
```

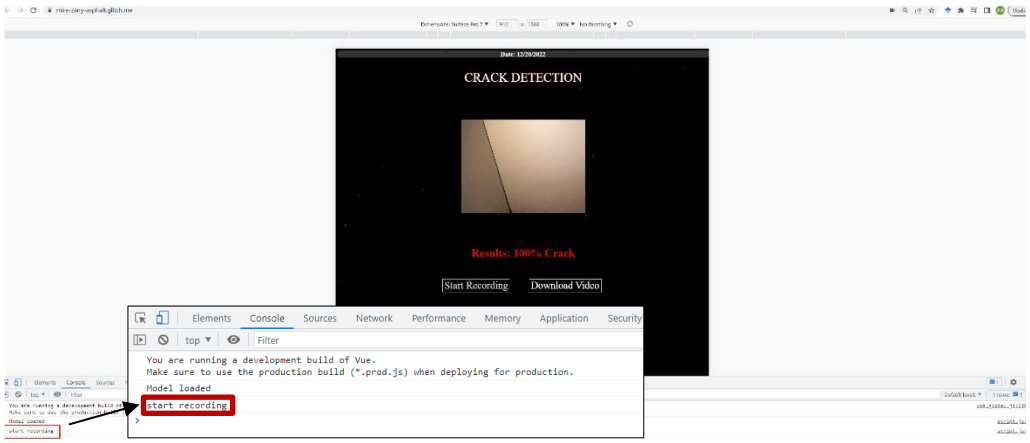
(b) Save Video section

Figure 7. 5 HTML syntax of crack inspection and robot control sections

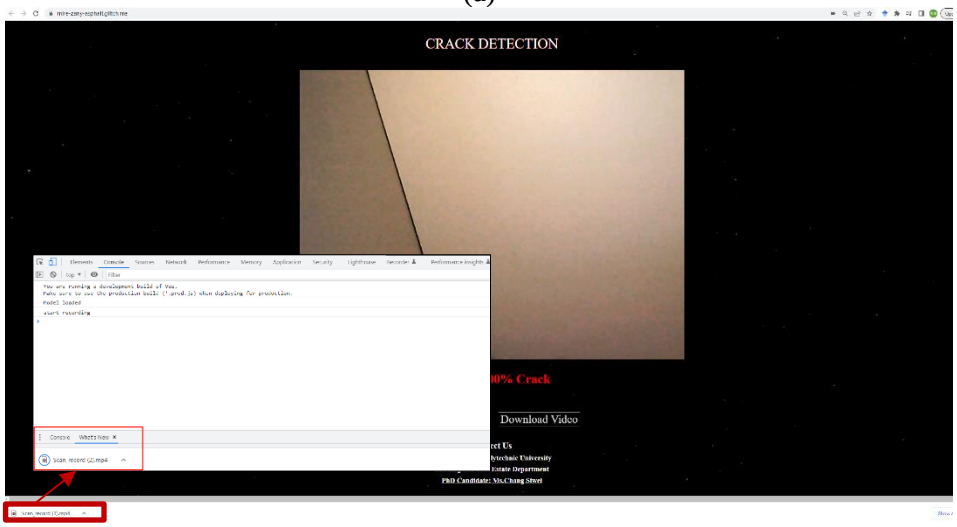
7.3.2 Recording video stream

In addition to checking for cracks instantaneously, recording the inspection videos is essential to allowing multiple stakeholders to check and track the inspection history. In order to implement the function of recording and saving inspection video streams, the developed user interface includes a “video recording” section. Users can directly record and download inspection videos on the webpage by pressing the “Start recording” and “Download Video” buttons. As shown in **Figure 7.5 (b)**, lines 25–30 of the code, the “video recording” section was distributed to a new … element, which was placed under the “crack inspection” section. Two <button>…</button> elements were established to realize the starting recording and downloading video actions, respectively. The recorded videos begin when the start recording button is pressed, and they end when the download video button is pressed. Two id selectors and one class selector “arrow-L”, “arrow-R”, and “buttons” was given to the <button> and elements, respectively to format their styles in the .css sheet. For example, similar to the crack inspection section, the css properties align-items and justify-content were set to center, respectively, to ensure the contents of the buttons div aligned along the center. **Figure 7.6** displays a sample of the “video recording” section. As seen in **Figure 7.6 (a)**, after clicking

the start recording button, the message “Start recording” is displayed in the web console to notice users that the video stream has started to be recorded. When users click the “Download Video” button on the webpage, the recorded video is immediately downloaded, as seen in **Figure 7.6 (b)**. The downloaded video can be readily moved or checked in the specified file paths by users.



(a)



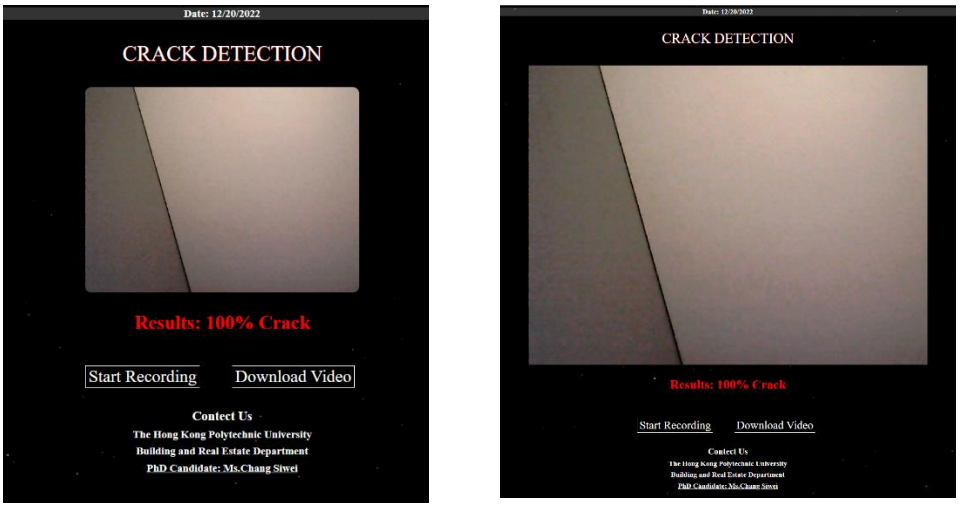
(b)

Figure 7. 6 Functions of start recording and download video

7.3.3 Autonomous rescaling and resizing webpage

To adapt the content size to different screens, such as desktop computer screens, mobile phone screens, or tablet screens, and keep the proper layout, the viewport units vh and vw were established for the sections of “date”, “title”, “video

stream," "video recording," and "contact information," respectively. For example, the width and height of the video stream window were set to "50vw" and "60vh" to keep the window at a ratio of 50% of the screen width and 60% of the screen height. Supported by the vw and vh units, the width and height of the window change appropriately based on the screen sizes. Therefore, the width and height of the video stream window automatically adapt to different screens when users browse the webpage on different equipment. Without configuring the viewport units, the size of the video stream window in the earlier version is given on a small scale, as seen in **Figure 7.7 (a)**. The size of the video stream window increases after the adding of the vw and vh units, as illustrated in **Figure 7.7(b)**, and the size of other sections decreases as well to highlight the crack inspection videos and inspection results.



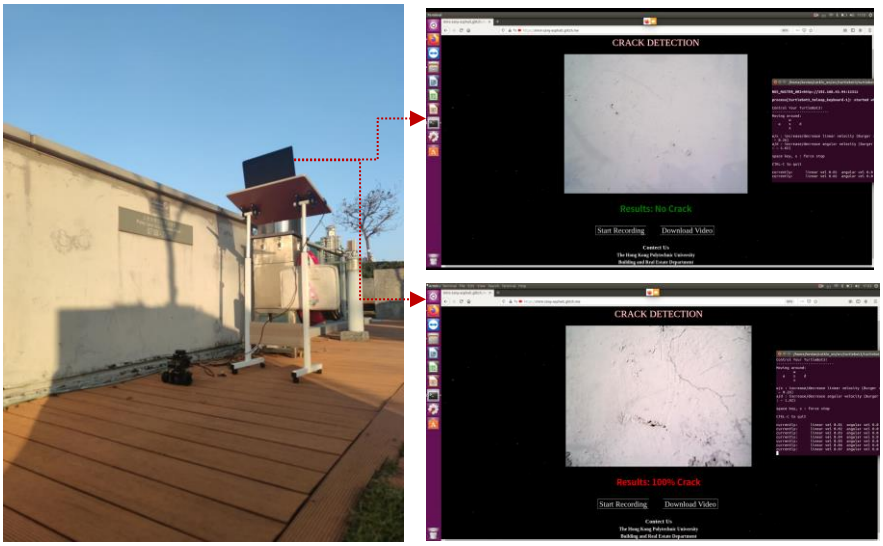
Without viewport units With viewport units

Figure 7. 7 With and without viewport units

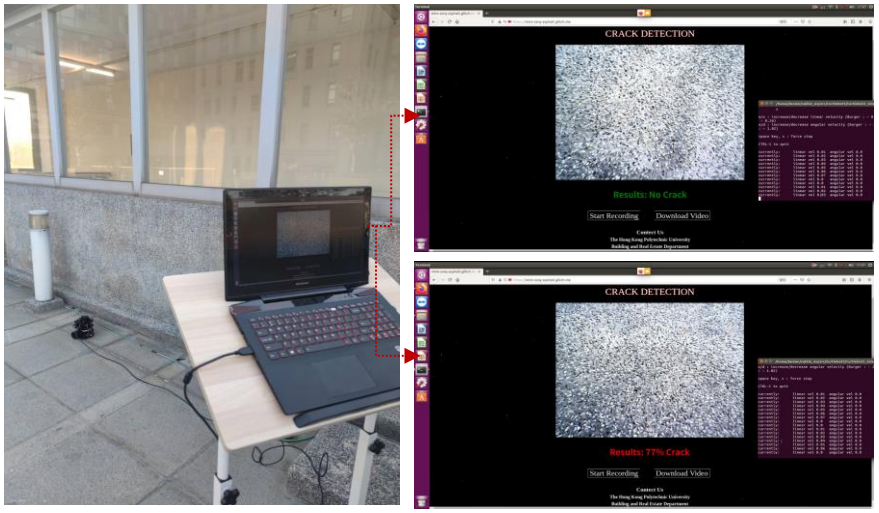
7.4 On-site validation

An on-site validation at the Hong Kong Polytechnic University was carried out to confirm the viability of the developed web user interface. It has been demonstrated that users can obtain the inspection robot's videos and outcome automatically,

remotely, smoothly, and in real time. The outcomes were automatically generated by processing the received video streams using the developed lightweight CNN model. The CNN model was coded in the backend JavaScript plugin. As shown in **Figure 7.8**, when the testing robot was following the walls to inspect cracks, the results “Crack” or “a% Crack” (where a refers to the probability of the crack) showed on a remote laptop computer screen accordingly under the video stream window. Cracked and non-cracked surfaces on both concrete and stone walls can be properly inspected.



Senario A



Senario B

Figure 7. 8 With and without viewport units

Although utilizing the “SSH-Y” protocol assists to provide remotely and real-time inspection results, the system frequently became stuck, making it difficult to smoothly demonstrate both the video streams and the inspection results. An example of the image frames demonstrated in the “SSH-Y” GUI and the web user interface is shown in **Figures 7.9**. It can be seen that new inspection frames are displayed in the “SSH-Y” GUI after the system refreshes the screen, which takes at least 10 seconds. On the other hand, the video stream and the results of the inspection were presented in the web user interface smoothly. The video streams in the web user interface can be updated every second. Processed by the lightweight CNN model coded in the back-end JavaScript, the inspection results of "Crack" or “No Crack” were also refreshed timely under the video stream. In conclusion, the web interface offers users a remote, real-time, and smooth demonstration as compared to using the “SSH-Y” protocol to display the inspection outcomes.

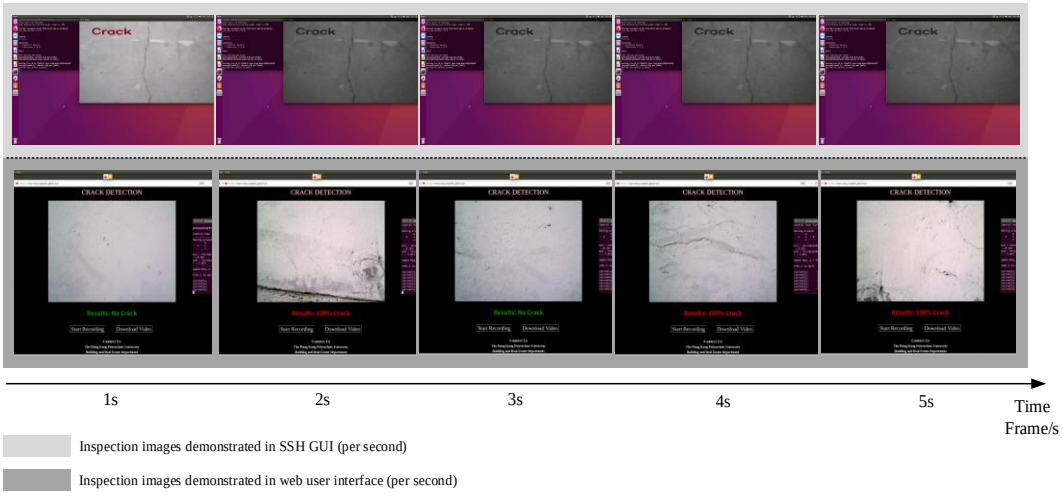


Figure 7. 9 Inspection images demonstrated in SSH GUI and web user interface

7.5 Summary

In this chapter, a web user interface was developed to realize the visualization of the crack inspection process. The web user interface provides the main functions of

viewing inspection video streams, automatically receiving inspection outcomes, and recording and downloading inspection videos. To enable the developed CNN network to be directly processed in the webpage, the tensorflow.js library and its high-level API was employed to import the machine learning model to the backend JavaScript coding. By establishing the viewport units, the web user interface supports automated scaling to adapt the contents to different equipment screens.

The feasibility of the developed web user interface was validated through an on-site validation. Using the web-based user interface, users can receive the crack inspection videos and outcomes automatically, remotely, and continuously. Meanwhile, users can record and download the inspection videos on the webpage at any time. It has been proven that using the web user interface apparently improves the visualization fluency of CNN models compared to the “SSH-Y” protocol.

It should be noted that the remote demonstration can be limited by the length of the USB cable because the MediaDevices.getUserMedia () function captures the video stream only from the local devices. Although USB cables provide faster and more stable data transfer, there’s a need to employ wireless data transfer techniques to realize remote crack inspection, such as connecting a USB port via a server or using the WebRTC technique to capture a remote video stream.

Chapter8: Conclusions and Recommendations

A robotic control system was proposed in this research to realize automatic building crack inspection. To achieve this, the development trend of construction inspection robotics was investigated to first target the supporting development technologies. Then, a lightweight convolutional network, a fuzzy logic-enabled wall-following algorithm, and a web user interface were designed for the development of robotic vision, navigation, and visualization, respectively. Validated in both simulations and on-site experiments, the robotic control system successfully controls the robot to implement the inspection behaviors of “following the buildings” and “scanning cracks”. The building cracks, recognized by the robot, can be demonstrated to users’ screens continuously, smoothly, and remotely. The developed robotic control system provides a fully automated way to accomplish the building crack inspection works, which assists in alleviating emerging challenges with construction inspections, such as manpower shortages, low accuracy, and the high cost of employing skilled labor.

8.1 Conclusions

This research presents both the theoretical and practical contributions to developing a fully automated robotic control system for performing crack inspection. First, the supporting technologies for the development of the robotic control system were targeted. The findings contribute to the academic research by 1) comprehensive identifying the characteristic of robotics technologies employed for construction inspections; 2) analyzing each type of robotic technology in terms of advantages, disadvantages, suitable application fields, evolution patterns, and potential trend.

The findings are reliable because they are obtained based on automated and quantitative analysis with minimal subjective effects. For practical purposes, the findings offer cutting-edge recommendations for developing and upgrading building crack inspection robots. Guided by the advice, the inspection robotic technology can be improved to serve practical construction needs at a rapid pace.

Second, a lightweight and accurate CNN architecture was provided for robotic vision. The proposed lightweight CNN is more suited for robotic platforms than the conventional CNNs for building crack inspection. Without being limited to those powered by GPUs, it is feasible for the majority of robotic control boards, such as the Raspberry Pi. The proposed lightweight CNN is also more accurate than the advanced lightweight CNNs. In the training and testing datasets, the F1-score was 96.8% and 92.4%, respectively. From a theoretical standpoint, the findings provide a method for designing lightweight CNN architectures that help to balance computation cost and accuracy. Practically speaking, the lightweight CNN design is more likely to be accepted by industrial clients since less special equipment will be required to operate the AI algorithms, which makes it easier to integrate high-tech solutions into traditional construction processes.

Thirdly, an FLC was developed for the wall-following behavior in order to implement robotic autonomous navigation. The designed FLC is considered as a viable path plan algorithm for carrying out the “following buildings” and “scanning cracks” behaviors of building crack inspection. The research findings help to share a reliable FLC design, which improves the wall-following algorithm’s suitability to be utilized in building inspection scenarios. The proposed FLC system is robust, it

allows the inspection robot to travel autonomously in eight distinct construction scenarios, including narrow alias and building corners. By integrating the output velocities of the eight distinct scenarios, the robot successfully navigated across integral building layers and avoided collision with forehead obstacles. The developed HA and BD algorithms, meanwhile, effectively assisted in maintaining straight-following pathways within a certain distance and preventing path deviation in complicated regions such concave and convex regions.

Finally, a web user interface was designed for robotic visualization. The findings contribute to the smooth, continuous, and remote demonstration of inspection video streams and the crack recognition outcomes for users. The “video recording” function is also provided to enable users to record and download the inspection videos directly from the webpage for checking and tracing the inspection records. Compared with visualizing the inspection outcomes using the “SSH-Y” protocol, the developed user interface effectively improves the system stuck problem. The video streams can be displayed smoothly, and the inspection results, which are processed by the CNN model, can be updated within one second on the webpage. Meanwhile, different from the existing GUIs designed for robotic visualization, the developed web user interface is more user-friendly and has high mobile compatibility. Users can view the webpage by directly opening the URL address on various mobile devices, including laptops, tablets, and mobile phones. With the advantages of RWD, the webpage can adapt to different screen sizes by automatically scaling its contents.

To summarize, unlike traditional building crack inspection, the developed robotic

control system achieves fully automated inspection through the following technological advances: 1) The CNN architecture designed for robotic vision is lightweight and accurate. The lightweight CNN can be implemented on the majority of robotic platforms without being limited to those powered by GPUs. 2) The proposed FLC system for wall-following behavior is robust and well-suited to building crack inspection motions. Autonomous navigation can be achieved in both common building regions and complicated regions, such as narrow places. 3) The designed web user interface for robotic visualization is user-friendly and adaptable. Users could easily monitor the crack inspection process on the webpage. The video streams and inspection results could be demonstrated smoothly, constantly, and remotely. on the webpage. Thanks to ROS, the proposed robotic control system could be easily implemented on various robotic platforms. These advantages make this approach suitable for automatically conducting the regular manual crack inspection work without human intervention.

8.2 Limitations

Although the proposed robotic control system allows the robot to automatically accomplish the building crack inspection work, the following limitations hinder its continuous implementation. Firstly, the designed lightweight CNN model is light-sensitive. Because the training photographs for the CNN model were taken during the day, it’s possible that the robot won’t be capable of inspecting the cracks in a darkened environment. Secondly, because of the shortcomings of the distance laser, it is possible for the FLC to generate wrong commands when the robot meets transparent and reflective building materials, such as glass walls and metal doors. This problem needs to be solved because: 1) the FLC is sensitive to the input

distance data. 2) Glass walls or metal doors are widespread in modern buildings. Thirdly, since the designed web user interface can only currently receive and present video streams from cameras connected to local computers, methods of receiving and processing remote video streams should be taken into consideration in the backend JavaScript in order to realize remote visualization from a distance.

8.3 Suggestions

Further studies concentrating on the following research topics are recommended to improve the aforementioned limitations. First, it is worth preparing crack datasets that include “non-visible” photographs, such as infrared crack images, to allow the robot to inspect building cracks in low-light conditions. Meanwhile, a novel lightweight CNN model that trained with the “non-visible” datasets is necessary. The development of multi-sensor-based path planning algorithms, such as those incorporating lasers, lidar, or cameras, can be suggested as a potential research topic for improving the robustness of autonomous navigation strategies and preventing invalid navigation commands when the robot confronts transparent or reflective building materials. Last but not least, it is worthwhile to employ interactive protocols, such as the WebRTC API, when developing the web user interface to enable the visualization and processing of remote video streams that were recorded from cameras attached to the client-side computers.

References

You, S., Kim, J. H., Lee, S., Kamat, V., & Robert Jr, L. P., 2018. “Enhancing perceived safety in human–robot collaborative construction using immersive virtual environments.” Automation in Construction. 96: 161-170. https://doi.org/10.1016/j.autcon.2018.09.008

Citf.cic.hk. 2020. [online] Available at: <https://www.citf.cic.hk/?route=background.

Brehm, E. 2019., “Robots for masonry construction–Status quo and thoughts for the German market.” Mauerwerk, 23(2): 87-94. https://doi.org/10.1002/dama.201900004

Heimig, T., Kerber, E., Stumm, S., Mann, S., Reisgen, U. and Brell-Cokcan, S., 2020. “Towards robotic steel construction through adaptive incremental point welding.” Construction Robotics, 4(1-2): 49-60. https://doi.org/10.1007/s41693-019-00026-4

Matsumura, H., Nakagomi, T., & Takada, S., 2014. “A study on the welding procedure of the first layer of the narrow groove welding for steel frames of building by welding robots.” Welding International, 28(4): 264-272. https://doi.org/10.1080/09507116.2012.715885

Wagner, H., Alvarez, M., Kyjanek, O., Bhiri, Z., Buck, M. and Menges, A., 2020. “Flexible and transportable robotic timber construction platform – TIM.” Automation in Construction, 120: 103400. https://doi.org/10.1016/j.autcon.2020.103400

Cho, Y., Kim, K., Ma, S. and Ueda, J., 2018. “A Robotic wearable exoskeleton for construction worker’s safety and health.” In Construction Research Congress, 19-28.

Yu, H., Choi, I., Han, K., Choi, J., Chung, G. and Suh, J., 2018. “Development of a upper-limb exoskeleton robot for refractory construction.” Control Engineering Practice, 72: 104-113. https://doi.org/10.1016/j.conengprac.2017.09.003

Liu, L., Leonhardt, S., Ngo, C. and Misgeld, B., 2020., “Impedance-Controlled Variable Stiffness Actuator for Lower Limb Robot Applications.” IEEE Transactions on Automation Science and Engineering, 17(2): 991-1004. https://doi.org/10.1109/TASE.2019.2954769

Chu, B., Jung, K., Lim, M. and Hong, D., 2013., “Robot-based construction automation: An application to steel beam assembly (Part I).” Automation in Construction, 32: 46-61. https://doi.org/10.1016/j.autcon.2012.12.016

T. S. Kim, I. S. Jang, C. J. Shin and M. K. Lee, 2014., “Underwater construction robot for rubble leveling on the seabed for port construction” In 14th International Conference on Control, Automation and Systems (ICCAS 2014). 1657-1661. Seoul.

Dörfler, K., Sandy, T., Giftthaler, M., Gramazio, F., Kohler, M. and Buchli, J., 2016. “Automation of a Discrete Robotic Fabrication Process Using an Autonomous Mobile Robot.” Robotic Fabrication in Architecture, 204-217. https://doi.org/10.1007/978-3-319-26378-6_15

Pritschow, G., Dalacker, M., Kurz, J. and Gaenssle, M., 1996. “Technological aspects in the development of a mobile bricklaying robot.” Automation in Construction, 5(1): 3-13. https://doi.org/10.1016/0926-5805(95)00015-1

Assessing cracks in houses - BRE Group, BRE Group - Building A Better World Together. 2022. https://www.bregroup.com/insights/assessing-cracks-in-houses (accessed 15

October 2022).

Chitte, C.J., 2018, “Study on causes and prevention of cracks in building.” International Journal for Research in Applied Science and Engineering Technology, 6(3): 453-461.https://doi.org/10.22214/ijraset.2018.3073

Darko, A., & Chan, A. P. 2016. “Critical analysis of green building research trend in construction journals.” Habitat International, 57: 53-63. https://doi.org/10.1016/j.habitatint.2016.07.001

Gharbia, M., Chang-Richards, A., Lu, Y., Zhong, R. and Li, H., 2020. “Robotic technologies for on-site building construction: A systematic review.” Journal of Building Engineering, 32: 101584. https://doi.org/10.1016/j.jobe.2020.101584

Pan, M., Linner, T., Pan, W., Cheng, H. and Bock, T., 2020. “Structuring the context for construction robot development through integrated scenario approach.” Automation in Construction, 114: 103174. https://doi.org/10.1016/j.autcon.2020.103174

Cai S , Ma Z , Xiang X., 2021. “Trends of research and development on construction robotics considering the supporting technologies and successful applications.” In Proceedings of the 18th International Conference on Computing in Civil and Building Engineering. 2021.

Lee, J. H., Yoon, S. S., Kim, I. H., & Jung, H. J., 2018, “Diagnosis of crack damage on structures based on image processing techniques and R-CNN using unmanned aerial vehicle (UAV).” In Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems. SPIE.

Dung, C.V. and L.D. Anh, Autonomous concrete crack detection using deep fully convolutional neural network. Automat Constr, 2019. 99: 52-58. https://doi.org/10.1016/j.autcon.2018.11.028

Yang, J., et al., 2019. Infrared thermal imaging-based crack detection using deep learning. IEEE Access. 7: 182060-182077.https://doi.org/10.1109/Access.2019.2958264

Shin, H. C., Roth, H. R., Gao, M., Lu, L., Xu, Z., Nogues, I., ... & Summers, R. M. 2016. “Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning.” IEEE transactions on medical imaging, 35(5), 1285-1298. https://doi.org/10.1109/TMI.2016.2528162

Ahmed, T. U., Hossain, M. S., Alam, M. J., & Andersson, K., 2019. “An integrated CNN-RNN framework to assess road crack.” In 2019 22nd International Conference on Computer and Information Technology (ICCIT). 1-6. Bangladesh: IEEE.

Dong, J.X., et al., 2021. “Intelligent segmentation and measurement model for asphalt road cracks based on modified mask R-CNN algorithm.” Cmes-Comp Model Eng, 2021. 128(2): 541-564.https://doi.org/10.32604/cmes.2021.015875

Li, S.Y. and X.F. Zhao, 2019., “Image-based concrete crack detection using convolutional neural network and exhaustive search technique.” Adv Civ Eng. https://doi.org/10.1155/2019/6520620

Pandey, A., Pandey, S., & Parhi, D. R., 2017. “Mobile robot navigation and obstacle avoidance techniques: A review.” Int Rob Auto J, 2(3): 00022. https://doi.org/ 10.15406/iratj.2017.02.00023

Tang, X., & Yamada, H., 2011. “Tele-operation construction robot control system with virtual reality technology.” Procedia Engineering., 15: 1071-1076. https://doi.org/10.1016/j.proeng.2011.08.198.

Durrant-Whyte, H., & Bailey, T., 2006. “Simultaneous localization and mapping: part I.” IEEE robotics & automation magazine., 13(2): 99-110. https://doi.org/10.1109/MRA.2006.1638022.

Chakraborty, J., Katunin, A., Klikowicz, P., & Salamak, M., 2019. “Early crack detection of reinforced concrete structure using embedded sensors.” Sensors., 19.18: 3879. https://doi.org/10.3390/s19183879

Hamia, Rimond, Christophe Cordier, and Christophe Dolabdjian., 2014. “Eddy-current non-destructive testing system for the determination of crack orientation.” Ndt & E International 61: 24-28. https://doi.org/10.1016/j.ndteint.2013.09.005

Yuan, F., Yu, Y., Li, L., & Tian, G., 2021. “Investigation of DC electromagnetic-based motion induced eddy current on NDT for crack detection.” IEEE Sensors Journal 21: 7449-7457. https://doi.org/10.1109/JSEN.2021.3049551

Omer, Hussein Mohammed Ali., 2020. “Determine The Depth of The Gaps and Cracks in The Concrete Using Ultrasonic Testing.” Diss. Sudan University of Science and Technology.

Yu, Chih-Peng, Yiching Lin, and Chia-Chia Chang., 2021. “An effective crack-identification approach for impact echo signals using MWT spectrograms and scaled FFT spectra.” Materials and Structures. 54.1: 1-21. https://doi.org/10.1617/s11527-020-01597-3

Hashimoto, Katsufumi, Tomoki Shiotani, and Masayasu Ohtsu., 2020. “Application of impact-echo method to 3D SIBIE rocedure for damage detection in concrete.” Applied Sciences. 10.8: 2729. https://doi.org/10.3390/app10082729

Li, Z. W., Liu, X. Z., Lu, H. Y., He, Y. L., & Zhou, Y. L., 2020. “Surface crack detection in precasted slab track in high-speed rail via infrared thermography.” Materials 13.21: 4837. https://doi.org/10.3390/ma13214837

Liu, J., Zhang, Z., Lin, Z., Chen, H., & Yin, W., 2021. “Characterization method of surface crack based on laser thermography.” IEEE Access 9: 76395-76402. https:// 10.1109/ACCESS.2021.3081435

Puthiyaveettil, Nithin, Prabhu Rajagopal, and Krishnan Balasubramaniam., 2021. “Influence of absorptivity of the material surface in crack detection using laser spot thermography.” NDT & E International 120: 102438. https://doi.org/10.1016/j.ndteint.2021.102438

Zoidis, N., Tatsis, E., Vlachopoulos, C., Gotzamanis, A., Clausen, J. S., Aggelis, D. G., & Matikas, T. E., 2013. “Inspection, evaluation and repair monitoring of cracked concrete floor using NDT methods.” Construction and Building Materials. 48: 1302-1308. https://doi.org/10.1016/j.conbuildmat.2013.06.082

Kilic, Gokhan., 2015. “Using advanced NDT for historic buildings: Towards an integrated multidisciplinary health assessment strategy.” Journal of Cultural Heritage 16.4: 526-535. https://doi.org/10.1016/j.culher.2014.09.010

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., ... & Farhan, L., 2021. “Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions.” Journal of big Data. 8.1: 1-74. https://doi.org/10.1186/s40537-021-00444-8

Cha, Young-Jin, and Wooram Choi., 2017. “Vision-based concrete crack detection using a convolutional neural network.” Dynamics of Civil Structures, 71-73. https://doi.org/ 10.1007/978-3-319-54777-0_9

Tong, Z., Gao, J., Han, Z., & Wang, Z., 2018. “Recognition of asphalt pavement crack length using deep convolutional neural networks.” Road Materials and Pavement Design 19.6: 1334-1349. https://doi.org/10.1080/14680629.2017.1308265

Huang, T. Vandoni, Carlo, E.,1996. “Computer Vision: Evolution and Promise.” 19th CERN School of Computing. Geneva: CERN. 21–25. https://10.5170/CERN-1996-008.21. ISBN 978-9290830955

Feng, X., Jiang, Y., Yang, X., Du, M., & Li, X., 2019. “Computer vision algorithms and hardware implementations: A survey. Integration.” 69: 309-320. https://doi.org/10.1016/j.vlsi.2019.07.005

Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K., 2018. “Convolutional neural networks: an overview and application in radiology. Insights into imaging.” 9(4): 611-629. https://doi.org/0.1007/s13244-018-0639-9

Chang, S., Siu, M. F. F., Li, H., & Luo, X., 2022. “Evolution pathways of robotic technologies and applications in construction.” Advanced Engineering Informatics. 51: 101529. https://doi.org/10.1016/j.aei.2022.101529

Klepper, S., 1997. “Industry life cycles.” Industrial and corporate change, 6(1): 145-182.

Yagishita, S. and Kanda, M., 1983. “Arc welding robot systems for large steel constructions.” IEEE Transactions on Industrial Electronics. (3):269-276. https://doi.org/10.1109/TIE.1983.356737

Kawashima, K., Sasaki, T., Miyata, T., Nagai, T., Chayama, K., Fujioka, A. and Mori, T., 2004. “Field test of remote control system for construction machines using robot arm.” In Proceedings of the 2004 IEEE International Conference on Control Applications, IEEE.

Feng, S., Xu, M., Zhao, D., Zhang, H., Tang, X. and Weng, G., 2006. “Research on RBF-PID control for the 6-DOF motion base in construction tele-robot system.” In The Proceedings of the Multiconference on Computational Engineering in Systems Applications. IEEE.

Sasaki, T., Miyata, T. and Kawashima, K., 2004. “Development of remote control system of construction machinery using pneumatic robot arm.” In 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE.

Yokoi, K., Nakashima, K., Kobayashi, M., Mihune, H., Hasunuma, H., Yanagihara, Y., Ueno, T., Gokyuu, T. and Endou, K., 2003. “A tele-operated humanoid robot drives a backhoe in the open air.” In Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003). Vol. 2:1117-1122. IEEE.

Yamada, H., Xinxing, T., Tao, N., Dingxuan, Z. and Yusof, A.A., 2009. “Tele-operation

construction robot control system with virtual reality." IFAC Proceedings Volumes, 42(16): 639-644. https://doi.org/10.3182/20090909-4-JP-2010.00108

Tang, X., Yamada, H., Huang, L. and Ahmad, A.Y., 2010, "Virtual reality-based teleoperation construction robot control system with 3dvisor device." In 2010 IEEE International Conference on Mechatronics and Automation. 384-388. IEEE.

Siles, I. and Walker, I.D., 2009. "Design, construction, and testing of a new class of mobile robots for cave exploration." In 2009 IEEE International Conference on Mechatronics. 1-6. IEEE.

Georgoulas, C., Linner, T. and Bock, T., 2014. "Towards a vision controlled robotic home environment." Automation in construction. 39: 106-116. https://doi.org/10.1016/j.autcon.2013.06.010

Ikeda, T., Yasui, S., Fujihara, M., Ohara, K., Ashizawa, S., Ichikawa, A., Okino, A., Oomichi, T. and Fukuda, T., 2017. "Wall contact by octo-rotor UAV with one DoF manipulator for bridge inspection." In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 5122-5127. IEEE.

Cui, D., Chen, D., Dong, H., Zhang, L., Qi, F., Lei, Y. and Gao, X., 2016. "Design and analysis of climbing robot based on construction surface inspection." In 2016 Chinese Control and Decision Conference (CCDC). 5331-5336. IEEE.

Takahashi, Y., Maehara, S., Ogawa, Y. and Satoh, T., 2018. "Concrete inspection systems using hammering robot imitating sounds of workers." In ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction. Vol. 35： 1-5. IAARC Publications.

Bolourian, N. and Hammad, A., 2020. "LiDAR-equipped UAV path planning considering potential locations of defects for bridge inspection." Automation in Construction. 117: 103250. https://doi.org/10.1016/j.autcon.2020.103250

Hoffmann, M., Skibicki, S., Pankratow, P., Zieliński, A., Pajor, M. and Techman, M., 2020. "Automation in the Construction of a 3D-Printed Concrete Wall with the Use of a Lintel" Gripper. Materials, 13(8): 1800. https://doi.org/10.3390/ma13081800

Brooks, H. 1994. "The relationship between science and technology." Research Policy. 23(5): 477-486. https://doi.org/10.1016/0048-7333(94)01001-3

Gharbia, M., Chang-Richards, A., Lu, Y., Zhong, R., & Li, H., 2020. "Robotic technologies for on-site building construction: A systematic review." Journal Of Building Engineering. 32: 101584. https://doi.org/10.1016/j.jobe.2020.101584

Gharbia, M., Chang-Richards, A. Y., & Zhong, R. Y., 2019. "Robotic technologies in concrete building construction: A systematic review." In ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction. 36: 10-19. Banff: IAARC Publications.

Cai, S., Ma, Z., Skibniewski, M. J., & Bao, S., 2019. "Construction automation and robotics for high-rise buildings over the past decades: A comprehensive review." Advanced Engineering Informatics. 42: 100989. https://doi.org/10.1016/j.aei.2019.100989

Li, J., Burnham, J. F., Lemley, T., & Britton, R. M., 2010. "Citation analysis: Comparison of web of science®, scopus™, SciFinder®, and google scholar." Journal of electronic

resources in medical libraries. 7(3): 196-217. https://doi.org/10.1080/15424065.2010.505518

Mongeon, P., & Paul-Hus, A., 2016. “The journal coverage of Web of Science and Scopus: a comparative analysis.” Scientometrics. 106(1): 213-228. https://doi.org/10.1007/s11192-015-1765-5

Falagas, M. E., Pitsouni, E. I., Malietzis, G. A., & Pappas, G., 2008. “Comparison of PubMed, Scopus, web of science, and Google scholar: strengths and weaknesses.” The FASEB journal. 22(2): 338-342. https://doi.org/10.1096/fj.07-9492LSF

Zhou, B., Gong, L., Chen, Q., Zhao, Y., Ling, X., & Liu, C., 2015. “Spatial-temporal database based asynchronous operation approach of fruit-harvesting robots.” In Intelligent Robotics and Applications. 392-400. Springer, Cham.

Xiang-sheng, C. H. E. N., Zhi-hao, X. U., Xiao-hua, B. A. O., Xue-tao, W. A. N. G., & Yan-bin, F. U., 2020. “Challenges and technological breakthroughs in tunnel construction in China.” China Journal of Highway and Transport. 33(12): 1.

Radhakrishnan, S., Erbis, S., Isaacs, J. A., & Kamarthi, S., 2017. “Novel keyword co-occurrence network-based methods to foster systematic reviews of scientific literature.” PloS one. 12(3): e0172778. https://doi.org/10.1371/journal.pone.0172778

Yu, Y., Li, Y., Zhang, Z., Gu, Z., Zhong, H., Zha, Q., ... & Chen, E., 2020. “A bibliometric analysis using VOSviewer of publications on COVID-19.” Annals of translational medicine. 8(13). https://doi.org/10.21037/atm-20-4235

Blei, D. M., Ng, A. Y., & Jordan, M. I., 2003. “Latent dirichlet allocation.” Journal of machine Learning research. 993-1022. https://dl.acm.org/doi/10.5555/944919.944937

Fabius, J., 1973. “Two characterizations of the Dirichlet distribution.” The Annals of Statistics. 583-587. https://www.jstor.org/stable/2958122

Alhamzawi, R., & Yu, K., 2012. “Variable selection in quantile regression via Gibbs sampling.” Journal of Applied Statistics. 39(4): 99-813. https://doi.org/10.1080/02664763.2011.620082

Mazzei, D., Chiarello, F. and Fantoni, G., 2021. “Analyzing social robotics research with natural anguage processing techniques.” Cognitive Computation. 13(2): 308-321. https://doi.org/10.1007/s12559-020-09799-1

Spreeuwers, L., 2014. “Derivation of LDA log likelihood ratio one-to-one classifier.” University Of Twente Students Journal Of Biometrics And Computer Vision. 1(1). https://doi.org/10.3990/3.utsjbcv.i1.1

Syed, S., & Spruit, M., 2017. “Full-text or abstract? Examining topic coherence scores using latent dirichlet allocation.” In 2017 IEEE International conference on data science and advanced analytics (DSAA). 165-174. Tokyo: IEEE.

Park, J. S., Hong, S. G., & Kim, J. W., 2017. “A study on science technology trend and prediction using topic modeling.” Journal of the Korea Industrial Information Systems Research. 22(4): 19-28. https://doi.org/10.9723/jksiis.2017.22.4.019

Yang, H. L., Chang, T. W., & Choi, Y., 2018. Exploring the research trend of smart factory with topic modeling. Sustainability. 10(8): 2779. https://doi.org/10.3390/su10082779

Kim, H. H., & Rhee, H. Y., 2016. “Trend analysis of data mining research using topic

network analysis.” Journal of the Korea Society of Computer and Information. 21(5): 141-148. https://doi.org/10.9708/jksci.2016.21.5.141

Park, J. S., Hong, S. G., & Kim, J. W., 2017. “A study on science technology trend and prediction using topic modeling.” Journal of the Korea Industrial Information Systems Research. 22(4): 19-28. https://doi.org/10.9723/jksiis.2017.22.4.019

Yang, H. L., Chang, T. W., & Choi, Y., 2018. Exploring the research trend of smart factory with topic modeling. Sustainability. 10(8): 2779. https://doi.org/10.3390/su10082779

Kim, H. H., & Rhee, H. Y., 2016. “Trend analysis of data mining research using topic network analysis.” Journal of the Korea Society of Computer and Information. 21(5): 141-148. https://doi.org/10.9708/jksci.2016.21.5.141

Watanabe, A., Even, J., Morales, L. Y., & Ishi, C., 2015. Robot-assisted acoustic inspection of infrastructures-cooperative hammer sounding inspection. In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 5942-5947). IEEE.

Journa Fukuda, T., Hosokai, H., & Uemura, M., 1989. Rubber gas actuator driven by hydrogen storage alloy for in-pipe inspection mobile robot with flexible structure. In 1989 IEEE International Conference on Robotics and Automation. 1847-1848. IEEE Computer Society.

Press releases -- https://www.info.gov.hk/gia/general/201501/28/P201501280416_print.htm

Herm, L. V., Janiesch, C., Helm, A., Imgrund, F., Hofmann, A., & Winkelmann, A. 2022. “A framework for implementing robotic process automation projects.” Information Systems and e-Business Management: 1-35. https://doi.org/10.1007/s10257-022-00553-8

Market Shapshot -- https://issuu.com/rjgpt6/docs/global_managed_print_services_marke_d4b7a3249a4329

Globenewswire -- https://www.globenewswire.com/en/news-release/2021/09/27/2303605/0/en/Global-Inspection-Robots-Market-to-Hit-13-94-Billion-by-2030-Allied-Market-Research.htmll.

Simonyan, K., & Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C., 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition. 4510-4520.

Shanmugam, J. V., Duraisamy, B., Simon, B. C., & Bhaskaran, P., 2022. “Alzheimer’s disease classification using pre-trained deep networks.” Biomedical Signal Processing and Control. 71: 103217. https://doi.org/10.1016/j.bspc.2021.103217

Ren, S., He, K., Girshick, R., & Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems. 28.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A., 2016. “You only look once: Unified,

real-time object detection." In Proceedings of the IEEE conference on computer vision and pattern recognition. 779-788.

Ledford, J. R., Lane, J. D., & Severini, K. E., 2018. "Systematic use of visual analysis for assessing outcomes in single case design studies." Brain Impairment. 19(1): 4-17. https://doi.org/10.1017/brimp.2017.16

Ali, R., et al., 2020. "Structural crack detection using deep convolutional neural networks." Automat Constr, 2022. 133: 103989.https://doi.org/10.1016/j.autcon.2021.103989

Wang, Y., et al. 2022. "Asphalt Pavement Crack Identification based on Two-Stage Training and Multi-Branch Model Integrating Multiple Attention." In 2022 IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI). IEEE.

Li, S.Y. and X.F. Zhao. 2019. "Image-Based Concrete Crack Detection Using Convolutional Neural Network and Exhaustive Search Technique." Adv Civ Eng. https://doi.org/10.1155/2019/6520620

Fang, W., et al., 2019. "DOG: A new background removal for object recognition from images." Neurocomputing. 361: 85-91. https://doi.org/10.1016/j.neucom.2019.05.095

Bouguettaya, A., A. Kechida, and A.M. Taberkit., 2019. "A survey on lightweight CNN-based object detection algorithms for platforms with limited computational resources." International Journal of Informatics and Applied Mathematics. 2(2): 28-44

Xue, H. and K. Ren. "Recent research trends on Model Compression and Knowledge Transfer in CNNs." In 2021 IEEE International Conference on Computer Science, Artificial Intelligence and Electronic Engineering (CSAIEE). IEEE.

Howard, A.G., et al., 2017. " Mobilenets: Efficient convolutional neural networks for mobile vision applications." arXiv preprint arXiv:.02767. https://doi.org/10.48550/arXiv.1704.04861

Nikouei, S.Y., et al. 2018. "Real-time human detection as an edge service enabled by a lightweight cnn." In 2018 IEEE International Conference on Edge Computing (EDGE). 2018. IEEE.

Anvarjon, T., Mustaqeem, and S. Kwon., 2020. "Deep-Net: A Lightweight CNN-Based Speech Emotion Recognition System Using Deep Frequency Features." Sensors-Basel. 20(18): 5212. https://doi.org/10.3390/s20185212

Iandola, F.N., et al., 2016. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size." arXiv preprint arXiv:.02767, https://doi.org/10.48550/arXiv.1602.07360

Sae-Lim, W., W. Wettayaprasit, and P. Aiyarak., 2019. "Convolutional neural networks using MobileNet for skin lesion classification." In 2019 16th international joint conference on computer science and software engineering (JCSSE). IEEE.

Bai, L., et al., 2022. "A Lightweight and Multiscale Network for Remote Sensing Image Scene Classification." IEEE Geoscience and Remote Sensing Letters. 19: 1-5. https://doi.org/10.1109/lgrs.2021.3078518

Kamel, D.K., D.S. S, and D.A. Bashar., 2020. "Tenancy Status Identification of Parking Slots Using Mobile Net Binary Classifier." Journal of Artificial Intelligence and Capsule

Networks. 2(3): 146-154. https://doi.org/10.36548/jaicn.2020.3.001

Li, Y.T., et al., 2018. “Research on a Surface Defect Detection Algorithm Based on MobileNet-SSD.” Appl Sci-Basel. 8(9): 1678. https://doi.org/10.3390/app8091678

Dais, D., et al., 2021. “Automatic crack classification and segmentation on masonry surfaces using convolutional neural networks and transfer learning.” Automat Constr, 2021. 125: 103606. https://doi.org/10.1016/j.autcon.2021.103606

Taresh, M.M., et al., 2021. “Transfer Learning to Detect COVID-19 Automatically from X-Ray Images Using Convolutional Neural Networks.” Int J Biomed Imaging. 2021. https://doi.org/10.1155/2021/8828404

Simonyan, K. and A. Zisserman., 2014. “Very deep convolutional networks for large-scale image recognition.” arXiv preprint arXiv:.02767. https://doi.org/10.48550/arXiv.1409.1556

Krizhevsky, A., I. Sutskever, and G.E. Hinton., 2017. “ImageNet Classification with Deep Convolutional Neural Networks.” Commun Acm. 60(6): 84-90. https://doi.org/10.1145/3065386

Lecun, Y., et al., 1998. “Gradient-based learning applied to document recognition.” P IEEE. 86(11): 2278-2324. https://doi.org/10.1109/5.726791

Gan, P.T., The Optimal Economic Uncertainty Index: A Grid Search Application. Comput Econ, 2014. 43(2): 159-182.https://doi.org/10.1007/s10614-013-9366-y

Cao, J., et al., 2018. “Softmax cross entropy loss with unbiased decision boundary for image classification.” In 2018 Chinese Automation Congress (CAC). IEEE.

Zhou, G.Q., et al., 2020. “Graphics Processing Unit-Accelerated Semiempirical Born Oppenheimer Molecular Dynamics Using PyTorch.” J Chem Theory Comput. 16(8): 4951-4962.https://doi.org/10.1021/acs.jctc.0c00243

Bojer, C.S. and J.P., 2021. “Meldgaard, Kaggle forecasting competitions: An overlooked learning opportunity.” Int J Forecasting. 37(2): 587-603.https://doi.org/10.1016/j.ijforecast.2020.07.007

Özgenel, Çağlar Fırat., 2018. “Concrete crack images for classification.” Mendeley Data, v1. https://doi.org/10.17632/5y9wdsg2zt 1

Dorafshan, S., R.J. Thomas, and M. Maguire., 2018. “SDNET2018: An annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks.” Data Brief. 21: 1664-1668. https://doi.org/10.1016/j.dib.2018.11.015

Tang, G., et al., 2019. “Adaptive CU split decision with pooling-variable CNN for VVC intra encoding.” In 2019 IEEE Visual Communications and Image Processing (VCIP). 2019. IEEE.

Agrawal, A. and N. Mittal., 2020. “Using CNN for facial expression recognition: a study of the effects of kernel size and number of filters on accuracy.” Visual Comput. 36(2): 405-412. https://doi.org/10.1007/s00371-019-01630-9

Ma, N., et al., 2018. “Shufflenet v2: Practical guidelines for efficient cnn architecture design.” In Proceedings of the European conference on computer vision (ECCV).

Townsend, J.T., 1971. “Theoretical Analysis of an Alphabetic Confusion Matrix.” Percept

Psychophys. 9(1a): 40-&. https://doi.org/10.3758/Bf03213026

Cha, Y.-J. and W. Choi., 2017. “Vision-based concrete crack detection using a convolutional neural network” In Dynamics of Civil Structures. Volume 2. Springer. 71-73.

Dung, C.V. and L.D. Anh., 2019. “Autonomous concrete crack detection using deep fully convolutional neural network.” Automat Constr. 99: 52-58. https://doi.org/10.1016/j.autcon.2018.11.028

Zhang, L., et al., 2016. “Road crack detection using deep convolutional neural network.” In 2016 IEEE international conference on image processing (ICIP). IEEE.

Yang, X.C., et al., 2018. “Automatic Pixel-Level Crack Detection and Measurement Using Fully Convolutional Network.” Comput-Aided Civ Inf. 33(12): 1090-1109. https://doi.org/10.1111/mice.12412

Zhang, A., et al., 2017. “Automated Pixel-Level Pavement Crack Detection on 3D Asphalt Surfaces Using a Deep-Learning Network.” Comput-Aided Civ Inf. 32(10): 805-819.https://doi.org/10.1111/mice.12297

Liu, Z.Q., et al., 2019. “Computer vision-based concrete crack detection using U-net fully convolutional networks.” Automat Constr. 104: 129-139.https://doi.org/10.1016/j.autcon.2019.04.005

Russakovsky, O., et al., 2015. “ImageNet Large Scale Visual Recognition Challenge.” Int J Comput Vision. 115(3): 211-252. https://doi.org/10.1007/s11263-015-0816-y

Ethisham R, Ahmad A, V Camp C, Chairman N, Mir J., 2022. “Evaluation of Pre-trained ResNet and MobileNetV2 CNN models for the Concrete Crack Detection and Crack Orientation Classification.” 1st International Conference on Advances in Civil and Environmental Engineering. Taxila Pakistan: MDPI. https://doi.org/10.13140/RG.2.2.31719.5264

Redmon, J. and A. Farhadi., 2018. “Yolov3: An incremental improvement.” arXiv preprint arXiv.02767. https://doi.org/10.48550/arXiv.1804.02767

Yata, T., Kleeman, L., & Yuta, S. I., 1998. “Wall following using angle information measured by a single ultrasonic transducer.” In Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146) (Vol. 2, pp. 1590-1596). IEEE.

Ayache, N., & Faugeras, O., 1989. “Maintaining representations of the environment of a mobile robot.” IEEE transactions on Robotics and Automation, 5(6), 804-819. https://doi.org/10.1109/70.8810

Moravec, H., & Elfes, A., 1985. “High resolution maps from wide angle sonar.” In Proceedings. IEEE international conference on robotics and automation., Vol. 2, 116-121. USA: IEEE.

Bui, H. D., Nguyen, S., Billah, U. H., Le, C., Tavakkoli, A., & La, H. M., 2020. “Control framework for a hybrid-steel bridge inspection robot.” In 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)., 2585-2591. USA: IEEE.

Özaslan, T., Loianno, G., Keller, J., Taylor, C. J., Kumar, V., Wozencraft, J. M., & Hood, T., 2017. “Autonomous navigation and mapping for inspection of penstocks and tunnels

with MAVs." IEEE Robotics and Automation Letters., 2(3), 1740-1747. https://doi.org/10.1109/lra.2017.2699790

Kaiwart, A., Dubey, N. D., Naseer, F., Verma, A., & Pradhan, S., 2022. "Design of Adaptive Wheel Driven Pipeline Inspection Robot." In Advances in Mechanical Engineering and Technology., 583-595. Singapore: Springer

Durrant-Whyte, H., & Bailey, T., 2006. "Simultaneous localization and mapping: part I." IEEE robotics & automation magazine., 13(2), 99-110. https://doi.org/10.1109/MRA.2006.1638022

Asadi, K., Haritsa, V. R., Han, K., & Ore, J. P., 2021. "Automated object manipulation using vision-based mobile robotic system for construction applications." Journal of Computing in Civil Engineering., 35(1), 04020058. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000946

McLaughlin, E., Charron, N., & Narasimhan, S., 2020. "Automated defect quantification in concrete bridges using robotics and deep learning." Journal of Computing in Civil Engineering., 34(5), 04020029. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000915.

Narasimhan, M., Wijmans, E., Chen, X., Darrell, T., Batra, D., Parikh, D., & Singh, A., 2020. "Seeing the un-scene: Learning amodal semantic maps for room navigation." In European Conference on Computer Vision. 513-529. Springer, Cham.

Gul, F., Rahiman, W., & Nazli Alhady, S. S.,, 2019. "A comprehensive study for robot navigation techniques." Cogent Engineering. 6(1), 1632046. https://doi.org/10.1080/23311916.2019.1632046

Saman, A. B. S., & Abdramane, I. 2013. "Solving a reconfigurable maze using hybrid wall follower algorithm." International Journal of Computer Applications. 82(3), 22-26. https://doi.org/ 10.5120/14097-2114

Che, X., Zhang, Z., Sun, Y., Li, Y., & Li, C., 2022. "A Wall-Following Navigation Method for Autonomous Driving Based on Lidar in Tunnel Scenes." In 2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD)., 594-598. China: IEEE.

Wu, P., Fang, M., & Ding, Z., 2021. "Wall-Following Navigation for Mobile Robot Based on Random Forest and Genetic Algorithm." In International Conference on Intelligent Computing., 122-131. Cham: Springer.

Wei, X., Dong, E., Liu, C., Han, G., & Yang, J., 2017. A wall-following algorithm based on dynamic virtual walls for mobile robots navigation. In 2017 IEEE International Conference on Real-time Computing and Robotics (RCAR)., 46-51. Japan: IEEE.

Altman, N. S. 1992.,, "An introduction to kernel and nearest-neighbor nonparametric regression." The American Statistician., 46(3), 175-185. https://doi.org/10.2307/268520

Katoch, S., Chauhan, S. S., & Kumar, V. 2021. "A review on genetic algorithm: past, present, and future." Multimedia Tools and Applications., 80(5), 8091-8126. https://doi.org/10.1007/s11042-020-10139-6

Poli, R., Kennedy, J., & Blackwell, T., 2007. "Particle swarm optimization." Swarm intelligence, 1(1), 33-57. https://doi.org/10.1007/s11721-007-0002-

Saman, A. B. S., & Abdramane, I., 2013. "Solving a reconfigurable maze using hybrid wall

follower algorithm." International Journal of Computer Applications, 82(3), 22-26. https://doi.org/ 10.5120/14097-2114

Xue S, Jiang R, Wong S, Feliciani C, Shi X, Jia B., 2020. "Wall-following behaviour during evacuation under limited visibility: experiment and modelling." Transportmetrica A: transport science., 16.3, 626-653. https://doi.org/ 10.1080/23249935.2020.1722281

Hammad, Issam, Kamal El-Sankary, and Jason Gu., 2019. "A comparative study on machine learning algorithms for the control of a wall following robot." 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO). China: IEEE.

Teng T, Veerajagadheswar P, Ramalingam B, Yin J, Elara Mohan R, Gómez B. 2020. "Vision based wall following framework: a case study with hsr robot for cleaning application." Sensors., 20(11), 3298. https://doi.org/10.3390/s2011329

Omrane, H., Masmoudi, M. S., & Masmoudi, M., 2016. "Fuzzy logic based control for autonomous mobile robot navigation." Computational intelligence and neuroscience., 1-10. https://doi.org/10.1155/2016/9548482

Fatmi, A., Al Yahmadi, A., Khriji, L., & Masmoudi, N., 2006. "A fuzzy logic based navigation of a mobile robot." In Transactions on engineering, computing and technology.

Malhotra, R., & Sarkar, A. 2005. "Development of a fuzzy logic based mobile robot for dynamic obstacle avoidance and goal acquisition in an unstructured environment." In Proceedings, 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics., 1198-1203. CA: IEEE.

Faisal, M., Hedjar, R., Al Sulaiman, M., & Al-Mutib, K., 2013. "Fuzzy logic navigation and obstacle avoidance by a mobile robot in an unknown dynamic environment." International Journal of Advanced Robotic Systems., 10(1), 37. https://doi.org/10.5772/54427

Suwoyo, H., Tian, Y., & Hajar, M. H. I., 2020. "Enhancing the Performance of the Wall-Following Robot Based on FLC-GA." Sinergi., 24(2), 141-152. https://doi.org/10.22441/sinergi.2020.2.008

Novák, V., Perfilieva, I., & Mockor, J., 2012. "Mathematical principles of fuzzy logic". Vol. 517. Springer Science & Business Media.

Aouf, Awatef, Lotfi Boussaid, and Anis Sakly., 2019. "Same fuzzy logic controller for two-wheeled mobile robot navigation in strange environments." Journal of Robotics., 2019, 1-11. https://doi.org/ 10.1155/2019/2465219

Singh, N. H., & Thongam, K., 2018. "Mobile robot navigation using fuzzy logic in static environments." Procedia Computer Science., 125, 11-17. https://doi.org/10.1016/j.procs.2017.12.004

Muthugala, M. A., Samarakoon, S. M., Mohan Rayguru, M., Ramalingam, B., & Elara, M. R., 2020. "Wall-following behavior for a disinfection robot using type 1 and type 2 fuzzy logic systems." Sensors, 20(16), 4445. https://doi.org/10.3390/s20164445

Cai, L., Liang, Z., Hou, Z. G., & Tan, M., 2008. "Fuzzy control of the inspection robot for obstacle-negotiation." In 2008 IEEE International Conference on Networking, Sensing

and Control., 117-122. China: IEEE.

Schiffer, S., Ferrein, A., & Lakemeyer, G., 2012. “Reasoning with qualitative positional information for domestic domains in the situation calculus.” Journal of Intelligent & Robotic Systems., 66(1), 273-300. https://doi.org/10.1007/s10846-011-9606-0

Cherroun, L., Nadour, M., & Kouzou, A., 2019. “Type-1 and Type-2 Fuzzy Logic Controllers for Autonomous Robotic Motion.” In 2019 International Conference on Applied Automation and Industrial Diagnostics (ICAAID)., 1 (1-5). Turkey: IEEE.

Ali, O. A. M., Ali, A. Y., & Sumait, B. S., 2015. “Comparison between the effects of different types of membership functions on fuzzy logic controller performance.” International Journal., 76, 76-83.

Antonelli, G., Chiaverini, S., & Fusco, G., 2007. “A fuzzy-logic-based approach for mobile robot path tracking.” IEEE transactions on fuzzy systems., 15(2), 211-221. https://doi.org/10.1109/TFUZZ.2006.87999

Witt, B. L., Wilbanks, J. J., Owens, B. C., & Rohe, D. P., 2022. “Stereophotogrammetry Camera Pose Optimization.” In Rotating Machinery, Optical Methods & Scanning LDV Methods., Vol.6, 13-38). Switzerland: Springer, Cham.

Sagawa, R., Kurita, N., Echigo, T., & Yagi, Y., 2004. “Compound catadioptric stereo sensor for omnidirectional object detection.” In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)., Vol. 3, 2612-2617. Japan: IEEE.

Okazaki, S. Y., Tanaka, T., Kaneko, S. I., & Takauji, H., 2008. “Object recognition based on relationship between camera vibration and measurement error on stereo measurement.” In Optomechatronic Technologies., Vol. 7266, 388-395. https://doi.org/10.1117/12.807265

Ren, J., Ren, M., Liu, R., Sun, L., & Zhang, K., 2021. “An Effective Imaging System for 3D Detection of Occluded Objects.” In The 4th International Conference on Image and Graphics Processing., 20-30. NY: ACM.

Chan, D. W., Hung, H. T., Chan, A. P., & Lo, T. K., 2014. “Overview of the development and implementation of the mandatory building inspection scheme (MBIS) in Hong Kong.” Built environment project and asset management., Vol.4, 71-89, https://doi.org/10.1108/BEPAM-07-2012-0040

Brown, M. M., Brown, G. C., Sharma, S., Busbee, B., & Brown, H., 2001. “Quality of life associated with unilateral and bilateral good vision.” Ophthalmology., 108(4), 643-647. https://doi.org/10.1016/S0161-6420(00)00635-7

MacInnes, I., & Smith, S., 2010. “A simple demonstration for estimating the persistence of vision.” The Physics Teacher., 48(6), 394-394. https://doi.org/10.1119/1.3479718

Skorka, O., & Joseph, D., 2011. “Toward a digital camera to rival the human eye.” Journal of Electronic Imaging, 20(3), 033009. https://doi.org/10.1117/1.3611015

Patti, A. J., Sezan, M. I., & Tekalp, A. M., 1994. “High-resolution image reconstruction from a low-resolution image sequence in the presence of time-varying motion blur.” In Proceedings of 1st International Conference on Image Processing., Vol. 1, 343-347. TX: IEEE.

Seraji, H., & Howard, A. 2002. “Behavior-based robot navigation on challenging terrain:

A fuzzy logic approach." IEEE Transactions on Robotics and Automation., 18(3), 308-321. https://doi.org/10.1109/TRA.2002.1019461

Hagras, H., 2004. "A type-2 fuzzy logic controller for autonomous mobile robots." In IEEE International Conference on Fuzzy Systems., Vol. 2, 965-970. Hungary: IEEE.

Lee, Y. T., Chiu, C. S., & Kuo, I. T., 2017. "Fuzzy wall-following control of a wheelchair." In 2017 Joint 17th World Congress of International Fuzzy Systems Association and 9th International Conference on Soft Computing and Intelligent Systems (IFSA-SCIS)., 1-6. Japan: IEEE.

Lead C. Standard Room Sizes & Their Location In Residential Building |Standard Room Size In a House - Civil Lead. Civil Lead. https://www.civillead.com/standard-room-sizes/. Published 2022. Accessed September 14, 2022.

McGee, V., 1985. "A counterexample to modus ponens." The Journal of Philosophy, 82(9), 462-471. https://doi.org/10.2307/2026276.

Gupta, M. M., & Qi, J.,, 1991. "Theory of T-norms and fuzzy inference methods." Fuzzy sets and systems., 40(3), 431-450. https://doi.org/10.1016/0165-0114(91)90171-L.

Dias, L. A., de Oliveira Silva, R. W., da Silva Emanuel, P. C., & Bento, R. T., 2018. "Application of the fuzzy logic for the development of automnomous robot with obstacles deviation." International Journal of Control, Automation and Systems., 16(2), 823-833. https://doi.org/10.1007/s12555-017-0055-9.

Rahbar, M., Mahdavinejad, M., Markazi, A. H., & Bemanian, M., 2022. "Architectural layout design through deep learning and agent-based modeling: A hybrid approach." Journal of Building Engineering., 47, 103822. https://doi.org/10.1016/j.jobe.2021.103822.

Hellmann, M., 2001. Fuzzy logic introduction. Rennes: Université de Rennes.

Chakraverty, S., Sahoo, D. M., & Mahato, N. R., 2019. "Defuzzification." In Concepts of Soft Computing., 117-127. Singapore: Springer.

Mittler, R., 2017. "ROS are good." Trends in plant science., 22(1), 11-19. https://doi.org/10.1016/j.tplants.2016.08.002

Agarwal, R. P., 2020. "Pythagorean theorem before and after Pythagoras." Adv. Stud. Contemp., Math, 30, 357-389. http://dx.doi.org/10.17777/ascm2020.30.3.357.

Braunstingl, R., Sanz, P., & Ezkerra, J. M., 1995. "Fuzzy logic wall following of a mobile robot based on the concept of general perception." In Procs. of the 7th Int. Conf. on Advanced Robotics (ICAR'95)., 367-376. Spain: IEEE.

Nadour, M., Boumehraz, M., Cherroun, L., & Puig Cayuela, V., 2019. "Hybrid type-2 fuzzy logic obstacle avoidance system based on horn-schunck method." Electrotehnica, Electronica, Automatica., 67(3), 45-51. http://hdl.handle.net/2117/178652

Brooks, H., 1994. "The relationship between science and technology." Research policy, 23(5), 477-486. https://doi.org/10.1016/0048-7333(94)01001-3

Nightingale, A., 2009. "A guide to systematic literature reviews." Surgery (Oxford), 27(9), 381-384. https://doi.org/10.1016/j.mpsur.2009.07.005

Chadegani, A. A., Salehi, H., Yunus, M. M., Farhadi, H., Fooladi, M., Farhadi, M., &

Ebrahim, N. A., 2013. “A comparison between two main academic literature collections: Web of Science and Scopus databases.” arXiv preprint arXiv:1305.0377. https://doi.org/10.48550/arXiv.1305.0377

Kyrkjebø, E., Liljeba¨ ck, P. L., & Transeth, A. A., 2009. “A robotic concept for remote inspection and maintenance on oil platforms.” In International Conference on Offshore Mechanics and Arctic Engineering. Vol. 43413: 667-674.

Gardner, B. S., 2011. “Responsive web design: Enriching the user experience.” Sigma Journal: Inside the Digital Ecosystem, 11(1), 13-19.

Wehrens, R., Putter, H., & Buydens, L. M., 2000. “The bootstrap: a tutorial.” Chemometrics and intelligent laboratory systems. 54(1), 35-52. https://doi.org/10.1016/S0169-7439(00)00102-7

Bootstrap 5 stable - summary, Download, Tutorial & Next releases (no date) MDB. Available at: https://mdbootstrap.com/docs/standard/bootstrap-5/ (Accessed: November 14, 2022).

McClurg-Genevese, Joshua David., 2005. “The principles of design.” Digital Web Magazine 13.

Kuo, L., Chang, T., & Lai, C. C., 2021. “Visual effect and color matching of dynamic image web page design.” Color Research & Application. 46(6): 1321-1331. https://doi.org/10.1002/col.22662

Berners-Lee, T., & Connolly, D., 1995. Hypertext markup language-2.0 (No. rfc1866).

Lie, H. W., & Bos, B., 1997. “Cascading style sheets: designing for the Web.” Addison-Wesley Longman Publishing Co., Inc.

Gardner, P. A., Maffeis, S., & Smith, G. D. 2012. “Towards a program logic for JavaScript.” In Proceedings of the 39th annual ACM SIGPLAN-SIGACT symposium on Principles of programming languages. 31-44.

Cullen, K. F. 2002. “PHP: An open source solution for Web programming and dynamic content.” Information technology and libraries. 21(3): 116.