國立臺北科技大學

Computer Vision and Image Measurement Final Project

Combining Machine Learning Image Segmentation Technique and Virtual Environment to Develop an Image-based Point-less Three-Dimensional Crack Information Algorithm

Researcher: Wei-Hsiang Chan

Advisor: Yuan-Sen Yang, Ph.D.

June, 2023

ABSTRACT

Title: Combining Machine Learning Image Segmentation Technique and Virtual Environment to Develop an Image-based Point-less Three-Dimensional Crack Information Algorithm

Pages: 142

School: National Taipei University of Technology

Department: Master Program of Civil and Disaster Prevention Engineering

Time: June, 2023

Course: Computer Vision and Image Measurement

Researcher: Wei-Hsiang Chan

Advisor: Yuan-Sen Yang, Ph.D.

Keywords: Image-based measurement, Synthetic scenario, Blender, Machine learning

Image-based measurement has the advantages of low cost, a certain level of accuracy, non-contact measurement, and suitability for global monitoring. Therefore, image-based measurement has higher potential and benefits for structure health monitoring. However, the development of image-based measurement typically requires a large amount of images for testing and validation. Collecting image-based measurement verification data in the field would incur significant costs. Traditional image tracking methods require features to facilitate algorithm tracking, but it may be difficult to accept the marking operation on the structure in practice. In recent years, computer vision, machine learning, and computer graphics techniques have rapidly advanced. By combining computer vision and machine learning, machines can automatically process complex and large amounts of image data

with good accuracy, significantly reducing labor costs and the risk of human error. At the same time, computer graphics software can generate highly realistic images.

In this study, the computer graphics software Blender was used to simulate realistic environmental conditions, and a dual-camera measurement system was used for measurement. The image of cracks on the wall is segmented using the image segmentation technology of machine learning, and the three-dimensional coordinates of the cracks on the wall are calculated using the algorithm developed in this research to explore the feasibility of this process for structural monitoring. The results of this study show that the 3D image measurement of cracks combined with machine learning methods has a certain level of accuracy.

Contents

Contents of Figure

Chapter 1 Introduction

1.1 Research Background

Traditional SHM (Structural Health Monitoring) is often measured by contact measurement systems, such as accelerometers, strain gauges and displacement gauges. The contact measurement system is mostly single-point measurement, which is less able to do global monitoring. Compared with contact measurement systems, non-contact measurement systems are more effective in monitoring structural health, and image measurement systems are one of them. The image measurement system has the advantages of low cost and a certain level of accuracy, and has considerable potential in SHM. In order to develop a good image measurement system, a large amount of image data is required for testing and verification during the development process, and the collection of image data is a big problem. If the collection is done on-site, it will cost a lot of money. Traditional image tracking methods often use template matching as the point of tracking. Therefore, in the past, it was necessary to set a target at the position to be tracked, and to do feature coating around it to reduce the tracking failure rate. However, setting the target and feature coating in It is not easy to be accepted in reality, and the traditional image tracking method is susceptible to environmental interference, such as changes in light and shadow, etc. Various reasons make it difficult for the method to be practically applied in the field.

Traditional SHM (Structural Health Monitoring) is often measured by contact measurement systems, such as accelerometers, strain gauges and displacement gauges. The contact measurement system is mostly single-point measurement, which is less able to do global monitoring. Compared with contact measurement systems, non-contact measurement systems are more effective in monitoring structural health, and image measurement systems are one of them. The image measurement system has the advantages of low cost and a certain level of

accuracy, and has considerable potential in SHM. In order to develop a good image measurement system, a large amount of image data is required for testing and verification during the development process, and the collection of image data is a big problem. If the collection is done on-site, it will cost a lot of money. Traditional image tracking methods often use template matching as the point of tracking. Therefore, in the past, it was necessary to set a target at the position to be tracked, and to do feature coating around it to reduce the tracking failure rate. However, setting the target and feature coating in It is not easy to be accepted in reality, and the traditional image tracking method is susceptible to environmental interference, such as changes in light and shadow, etc. Various reasons make it difficult for the method to be practically applied in the field.

1.2 Research Motivation and Purpose

The measurement process of the traditional contact measurement system needs to be in close contact with the test object and most of them can only do single-point measurement. Measurement systems such as image measurement, thermal imaging, and ultrasonic inspection are different from contact systems in that they are non-destructive detection methods that do not require contact with the test object during the measurement process, and are widely used today. Among them, the image measurement system has the advantages of low cost, easy installation, a certain level of accuracy, and suitable for global measurement. Its advantages will be more evident if the measurement area is located in a dangerous or difficult-to-reach place. With the development of image measurement technology so far, the image tracking error has the opportunity to be less than 0.02 pixels, and its advantages of global measurement enable image measurement technology to obtain small displacements and deformations in the test body. Through these measurement information, it is possible to calculate Find out the crack development trend, and fully obtain the mechanical behavior and characteristics of the test body.

Although the image measurement system has many of the above advantages, its sensitivity to environmental disturbances is much higher than other measurement methods, such as changes in light and shadow, slight shaking, etc., which may cause a decrease in measurement accuracy, and traditional image tracking methods It is necessary to carry out characteristic coating and set targets on the surface of the object to be measured, which makes this method less acceptable in monitoring practice. Therefore, in order to implement the image measurement method in practice, we developed a It is very important and urgent to set the target and directly perform image measurement with only one set of cameras.

If we want to develop a stable image measurement system, you must have a large amount of image data to test and verify repeatedly to optimize the algorithm. However, it is not easy to obtain image data. In the past, the image data used for the development of image measurement algorithms was mostly obtained from field experiments. Generally, the number of experiments on traditional concrete structures or steel structures is limited and the cost of experiments is high. Therefore, the image data of the image measurement algorithm is optimized. The sources are limited and precious, not to mention the development of image measurement systems for offshore wind turbines.

In recent years, with the advancement of hardware equipment, technologies in the fields of computer vision, machine learning, and computer graphics have also been improved. Today, computer graphics and animation technologies can achieve highly realistic image information. With this technology, a large number of image data of different experimental scenes and experimental conditions can be generated. In addition, there are many parameters that can be adjusted in the computer graphics software. Through these parameters, the light refraction, camera position, camera lens behavior, etc. can be precisely controlled, and the parameters of the image output by the computer graphics software can be set by the user, for example , camera position, lens angle, camera focal length and distortion parameters, etc., are conducive to the verification of image measurement algorithms. The combination of machine learning and computer vision is widely used today, such as unmanned driving, concrete deterioration detection and steel structure corrosion detection. Through its complex algorithms, machine learning models can use image features to obtain details that are difficult for humans to capture or distinguish. Structural degradation detection is the most widely used in civil engineering. Machine learning can automatically process a large amount of image data and quickly and accurately identify deteriorated areas, greatly saving labor costs. The image segmentation task in machine learning is widely used in medical imaging, which can quickly segment abnormal parts of organs, which is of great help to doctors in the diagnosis of diseases. Based on the above concept, this research uses computer graphics software to create a virtual experimental scene, generates a large amount of image data, and uses machine learning image segmentation technology to segment the crack shape, and finally uses feature point detection, least square method and back projection to calculate the three-dimensional space of the crack location, to explore the feasibility of this method for health monitoring of wall structures.

The main purpose of this research is to combine the image segmentation method of machine learning and the virtual environment of computer graphics to develop a method for measuring the coordinates of cracks in three-dimensional space without marking points. The research objectives of each stage of this research are briefly described as follows:

(1) Use the open source concrete crack training set and UNet3+ machine learning model architecture to train the concrete crack segmentation model.

(2) Use computer graphics software to build walls and attach concrete crack materials in the virtual experiment space, and set up dual cameras to illuminate the test body.

(3) Use the rendering technology of computer graphics software to output realistic virtual experiment images.

(4) Use the feature point matching technology to find the corresponding points of the left and right camera images.

(5) Use the corresponding point of the feature to perform triangulation, obtain the three-dimensional coordinates of the corresponding point of the feature, and obtain the approximate depth range of the crack in the wall.

(6) Use the trained machine learning model to predict the virtual experiment image to obtain the concrete crack segmentation image.

(7) Obtain pixels with cracks by segmenting images of concrete cracks.

(8) Calculate the three-dimensional space coordinates of the pixels with cracks by using the least square, the approximate depth range of the wall cracks, and the back-projection error.

(9) To sum up the above, this study will propose a set of image monitoring process without marker points for wall cracks, and explore the feasibility of this process.

1.3 Literature Review

The occurrence of extreme structural damage often catches people off guard and is accompanied by a certain degree of casualties. If there is a sound structural health monitoring mechanism, the precursors of damage can be detected early and corresponding reinforcement repairs can be made early. Therefore, the development of structural health monitoring is very urgent and important.

Image measurement technology is one of the non-contact monitoring systems. It has the advantages of low cost and a certain level of accuracy. It has great potential in structural health monitoring. In the past, limited by hardware equipment, including computer computing speed and photographic quality, it was difficult to realize these technologies. With the advancement of technology, hardware limitations have been overcome one by one, and image measurement has gradually emerged. Image measurement technology is an extension of the application of computer vision technology. OpenCV is a set of cross-platform and open source computer vision library. The library provides the most recent functions to make engineers more efficient in the development of computer vision related technologies. In recent years, image measurement technology has also been widely used in structural monitoring. Ji et al. (2020) used a high-speed camera to capture the deformation and cracks of structures under load and used the phase correlation method to calculate the displacement of adjacent frames to measure deformation and cracks. Yang et al. (2019) used computer vision technology to analyze cracks in concrete structures, and studied the use of image processing methods to extract crack information in images and convert them into feature vectors for damage assessment of concrete structures.

In recent years, machine learning has developed rapidly and has many applications in the civil engineering field. Cheng et al. (2021) used ultrasonic technology combined with machine learning for damage detection of steel I-beams. Kim et al. (2022) used binocular vision technology to detect concrete cracks. In this study, two lenses with different focal lengths were used to photograph the concrete surface, and the 3D model was reconstructed by image reconstruction and combined with machine learning and image processing technology to analyze the 3D model. , from which crack information is detected. Cha et al. (2017) used machine learning to automatically detect cracks in concrete surfaces.

The virtual environment is to create a realistic three-dimensional space in the computer through computer vision, video imaging and computer graphics technology. Virtual environments are often used for algorithm testing and optimization, and are often used to generate a large amount of training data required for machine learning model training to reduce development costs. Using machine learning model training requires a large amount of images and annotation data, but the collection of images and annotation data is a very time-consuming and expensive process. Rozantsev et al. (2015) and Tian et al. (2018) used synthetic images instead of actual images for training. It can easily generate image data of various scenes and situations, increase the diversity of training data, and greatly simplify the collection process of labeled data.

1.4 Paper Structure

There are four chapters in this paper, and the content of each chapter is briefly described as follows:

- Chapter 1 Introduction: Mainly introduces the research background, research motivation and purpose, and literature review.
- Chapter 2 Research Methods: Introduction to the software, programming language, and environment used in this study, and the development process and concept of the algorithm proposed in this study.
- Chapter 3 Conclusions and Recommendations: Summarize all the experimental results and provide some suggestions for future research.

Chapter 2 Research methods

2.1 Introduction to using the software

The software and tools used in this study are as follows:

(1) Computer vision library: OpenCV, a large library for image processing.

(2) Computer graphics software: Blender, which generates virtual images and provides the data needed for machine learning and image analysis.

The process of this research can be divided into two parts, the first part is machine learning, and the second part is virtual experiment, as shown in Figure 2.1. First of all, in the part of machine learning, this study obtained the open source concrete crack training set on Kaggle and used the UNet3+ machine learning model architecture for model training. This study uses the semantic segmentation in the image segmentation task, which can divide the object to be segmented Isolated from the background. The second part is a virtual experiment. In this study, a virtual wall model is built with computer graphics software, and the image of concrete cracks is attached to the wall in the form of material attachment, and then the image is rendered by the computer graphics model to generate a virtual wall. The image data used in the experiment, and finally use the virtual experiment image and segmented image to calculate the three-dimensional space coordinates of the wall cracks. The following will introduce the software tools used in this research respectively.

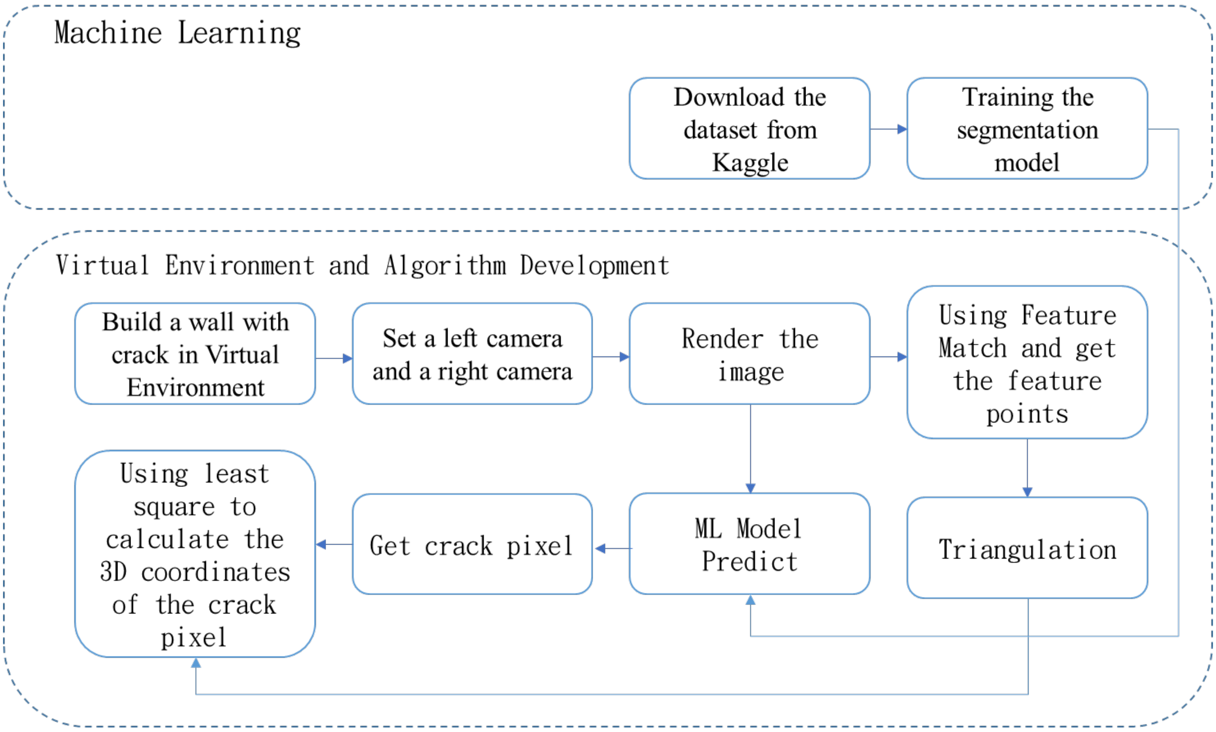


Figure 2.1 Research flow chart

2.1.1 Machine Learning Framework

This research adopts TensorFlow as the framework of machine learning and develops it in Python programming language. TensorFlow is an open source machine learning framework developed and maintained by Google. It provides a toolset for building and training machine learning models. In addition, TensorFlow supports multiple programming languages, including Python, C++ and Java, and supports cross-platform operation. TensorFlow is widely used in many fields, including image recognition, speech recognition, natural language processing, etc. One of the essential tools for machine learning researchers, engineers and academia. This study implements the UNet3+ (Huang et al., 2020) model based on the TensorFlow framework, and uses it as the model for image segmentation in this study.

2.1.2 Computer Vision Libraries

This study uses OpenCV (Open Source Computer Vision Library) as a library for image processing, and it is developed in Python programming language. OpenCV is an open source computer vision and image processing library. Composed of a set of cross-platform libraries and tools, it provides a large number of image processing and computer vision functions. OpenCV supports multiple programming languages, including C++, Python, and Java, and supports cross-platform operation. OpenCV is widely used in the field of computer vision and image processing. It can be used in object detection and recognition, surveillance systems, robotics, image processing, and autonomous driving. It is also one of the important tools for researchers, engineers and academia, providing many convenient and powerful libraries and tools, greatly simplifying the development process of computer vision and image processing. This research team will develop a single-camera measurement system based on OpenCV and C++ programming language in 2020 (Yan Hongxi, 2020). This research will use the version rewritten in Python language and OpenCV. In addition, this study also uses OpenCV and Python to develop a set of fan blade endpoint tracking algorithms based on machine learning segmentation images.

2.1.3 Computer graphics software

This research uses the open source and free 3D computer graphics software Blender to build a virtual experimental model and virtual environment, and uses the animation and rendering technology of the computer graphics software to generate the image data of the virtual experiment. Blender is used to create animation, visual effects, game development, modeling and rendering, and more. With powerful functions and a wide range of applications, it is often used in film production, animation production, game development, architectural visualization, industrial design and academic research. It provides a powerful toolset, including functions such as 3D modeling, animation production, material and texture setting, lighting and rendering. In addition, Blender also supports functions such as physical simulation, particle system, creation of game logic and Python scripting, providing a rich and flexible creative space. Blender is cross-platform and can run on different operating systems. Its user interface is very friendly, and it provides rich official documents and teaching, so that beginners can easily get started. Due to its open source nature, Blender is supported and contributed by a large number of users, and users can freely download, use, modify and share the software.

2.2 The development process and concept of the algorithm proposed in this study

2.2.1 Concept of the algorithm

This Study uses a virtual environment, machine learning, and computer vision technique to develop an Image-based Point-less Three-Dimensional Crack Information Algorithm.

In this study, I use Blender, Python, Tensorflow, and OpenCV as the tools to develop the Algorithm.

The study process is divided into two parts, one part is machine learning, and the other part is virtual environment and algorithm development.

In the part of machine learning, I download the segmentation training dataset of cracks from Kaggle in this study and use UNet3+ to train the model. UNet3+ is a model for segmentation improved by UNet.

In the part of the virtual environment and algorithm development, first, build a wall and post an image of cracked concrete on the wall in the virtual environment, set the left camera and right camera to aim at the wall, render images using the left and right cameras respectively, use the feature detector to detect the image and Match the feature points of the images of the left camera and the right camera, and triangulate the matched feature points. The camera's intrinsic parameters and extrinsic parameters used in the triangulation are taken from Blender. After the triangulation, the depth range of the crack can be obtained, using machine learning to predict the image of the left camera to generate a crack segmentation image, use the crack segmentation image to find all the pixel positions with cracks in the image, use these pixel positions and least square to calculate the three-dimensional coordinates of all cracks, and use the least square The part of will limit the depth direction range before iterative calculation, that is the previously calculated depth direction range.

As shown in Figure 2.2, this picture is a schematic diagram of building a wall in a virtual environment and posting an image of cracked concrete on the wall and setting the left and right cameras. As shown in Figure 2.3, These are the images rendered by the left and right cameras respectively.

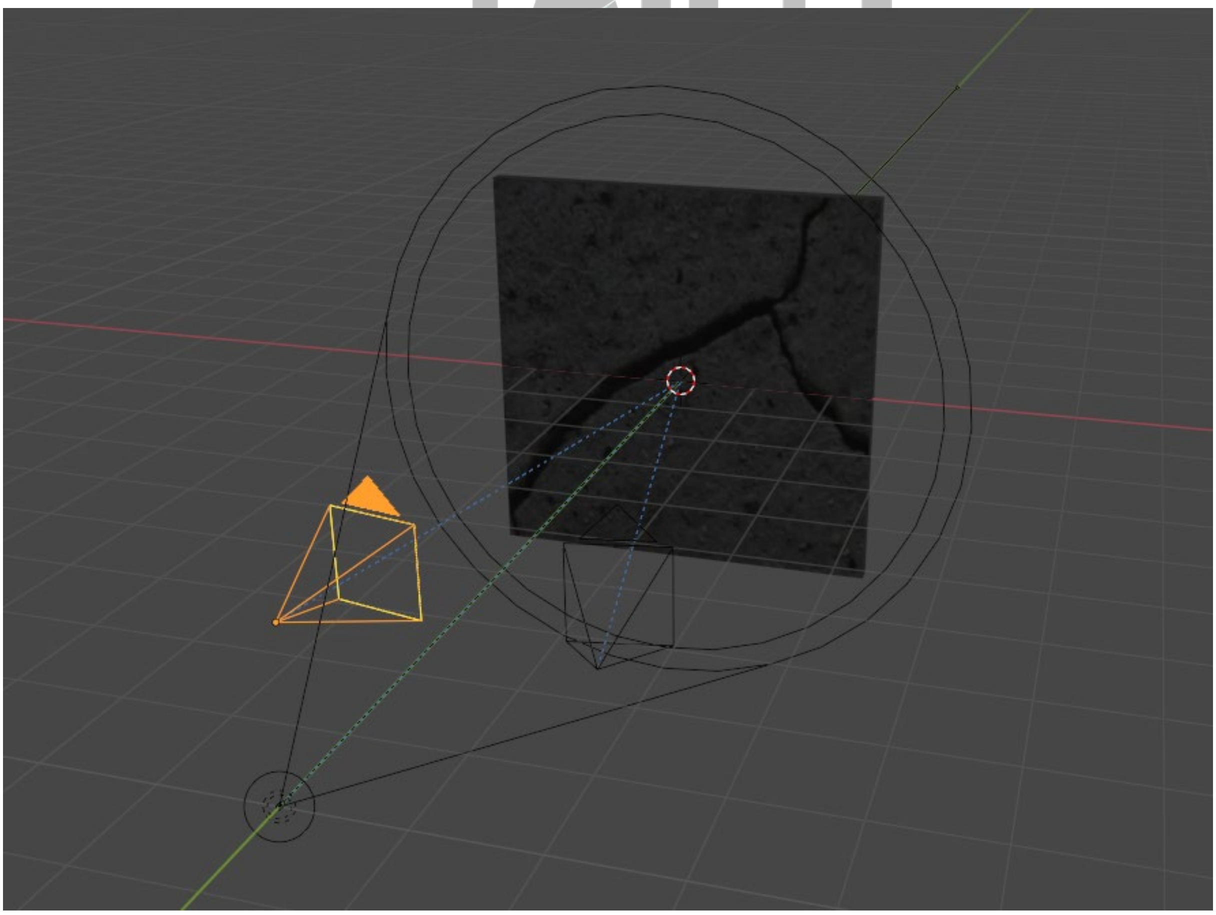


Figure 2.2 The schematic diagram of virtual environment.

(a) Rendered by left camera.

(b) Rendered by right camera.

Figure 2.3 The images rendered by Blender.

As shown in Figure 2.4, this picture is a schematic diagram of building a wall in a virtual environment and posting a segmented image of cracked concrete on the wall and setting the left and right cameras. because of lack of time, the machine learning model has not been trained yet, so the segmented image is replaced by this method. As shown in Figure 2.5, These are the images rendered by the left and right cameras respectively.

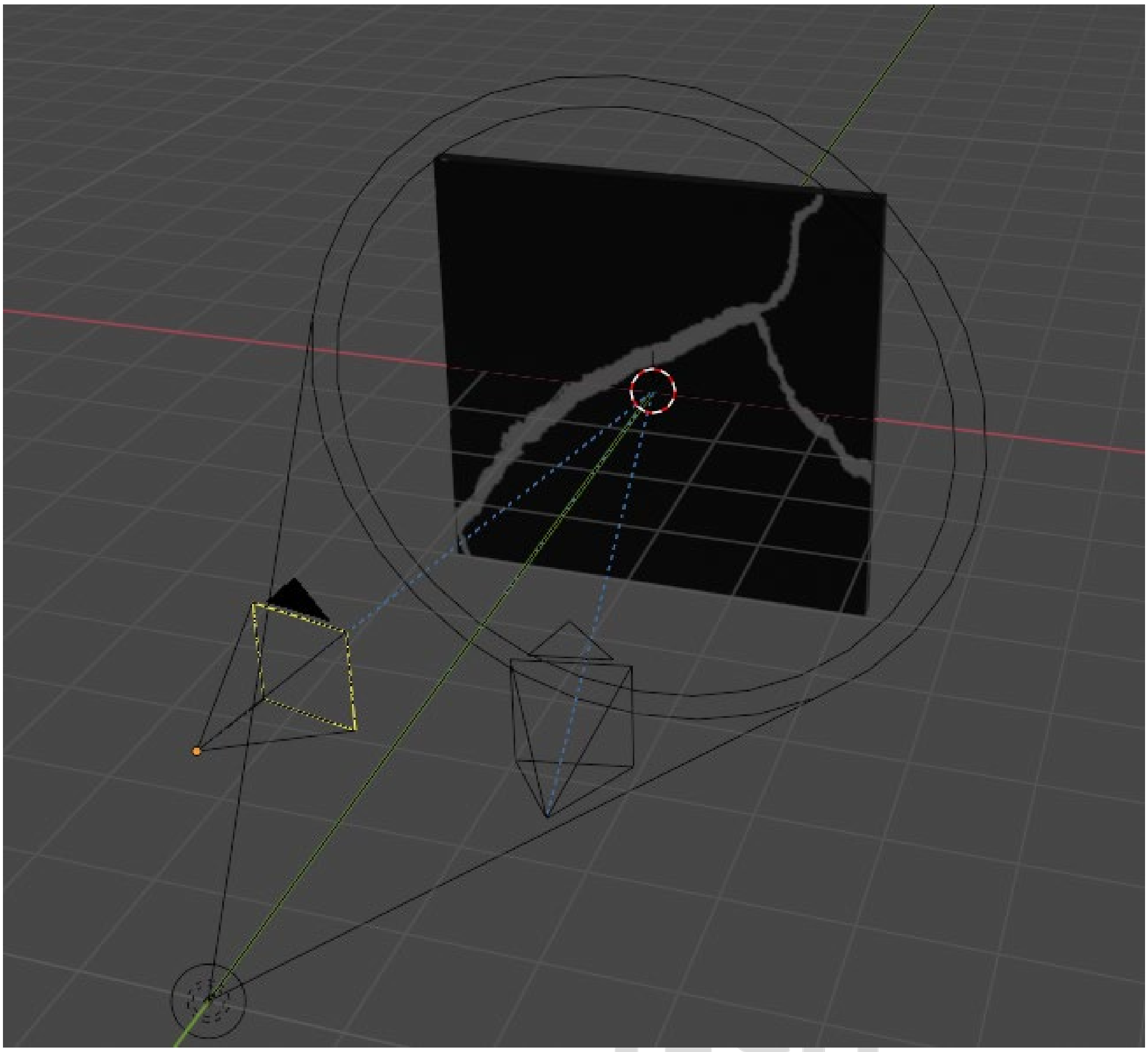
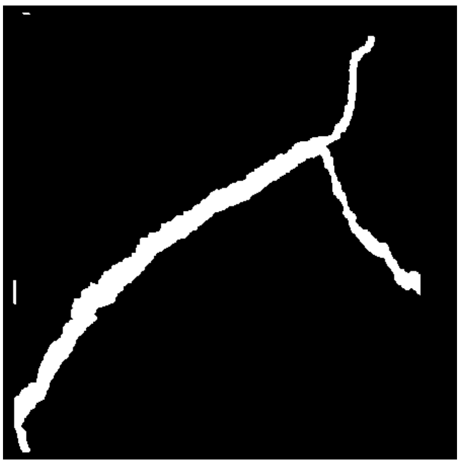
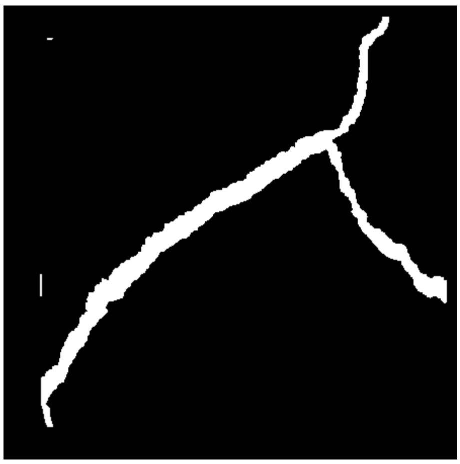


Figure 2.4 The schematic diagram of virtual environment.



(a) Rendered by left camera.



(b) Rendered by right camera.

Figure 2.5 The images rendered by Blender.

As shown in Figure 2.6, this picture is to match the feature points of the images taken by the left camera and the right camera, and the corresponding relationship of the feature points can be seen on the picture

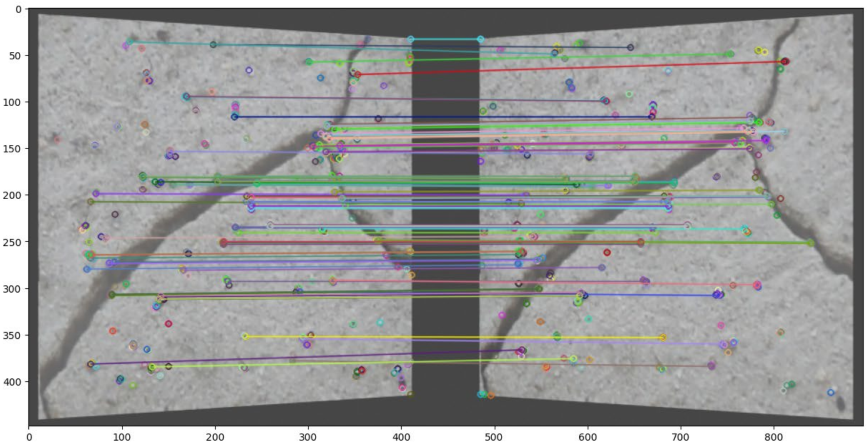


Figure 2.6 Feature points matching by original images.

As shown in Figure 2.7, this picture is the 3D coordinates of the crack calculated by the algorithm proposed in this study. The depth range here is calculated by matching the feature points of the original image. The points of crack position are about 6 to 7 cm. And the depth of wall surface in the model I built in the virtual environment is 5 cm. It can be observed that the depth calculated by this algorithm is quite close to the actual depth.

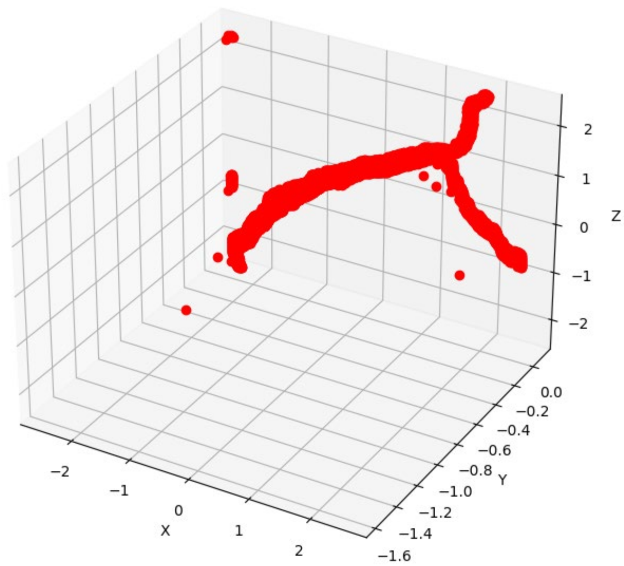


Figure 2.7 3D coordinates of the crack using the depth range here is calculated by matching the feature points of the original image.

As shown in Figure 2.8, here I try to use segmented images for feature point matching, this picture is the result of feature point matching. The correspondence shown in this image seems to be better than matching the original image.

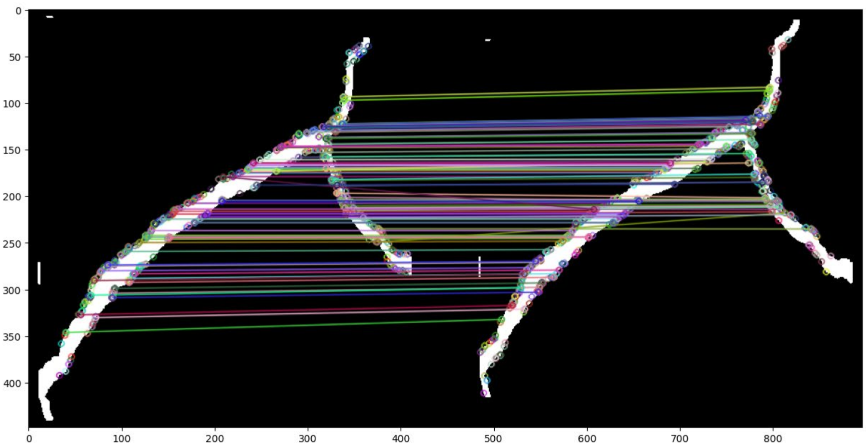


Figure 2.8 Feature points matching by segmented images.

As shown in Figure 2.8, this picture is the 3D coordinates of the crack calculated by the algorithm proposed in this study. The depth range here is calculated by matching the feature points of the segmented image. The points of the crack position are about 1.95 meters to 2 meters. Compared with using the original images, there's a large error.

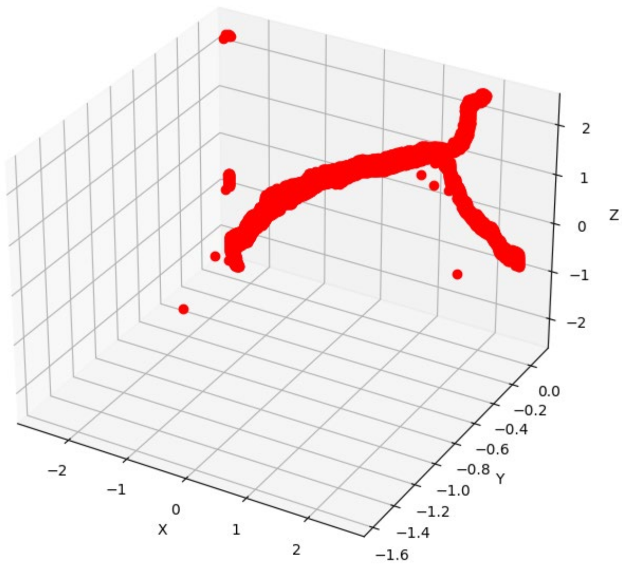


Figure 2.9 3D coordinates of the crack using the depth range here is calculated by matching the feature points of the segmented image.

Chapter 3 Conclusion and Suggestion

This study uses machine learning segmentation tasks, dual-camera image measurement systems, and virtual environments to develop a wall crack monitoring system based on machine learning segmentation tasks. Through experiments, it can be observed that this method can achieve good results. In the past, when performing image measurement, in order to carry out image tracking of the target to be measured, it is necessary to set target marking points and feature coating on the target to be measured in advance. Although this method can obtain high-precision measurement results, it is not practical in practical applications. It is so easy to implement, because the layout of target marks and feature coating will affect the appearance of the object under test, making the object under test not so good-looking, which makes it difficult to apply image measurement in practice. Therefore, this study changed the part of image tracking, and developed a set of three-dimensional space location algorithm for wall cracks based on machine learning segmentation images, replacing the target point setting and feature coating of traditional image tracking methods. The results show that using this method of 3D marking of fractures can achieve good results. This chapter will summarize the experience during the research period and divide it into two subsections for introduction, namely the conclusion and future prospects.

3.1 Conclusion

This section will introduce the key conclusions of this study, as follows:

1. This research uses the virtual image generated by the virtual environment to develop the image measurement algorithm, and has achieved very good results.
2. In this study, the UNet3+ architecture is used to train the machine learning segmentation task. It can obtain good segmentation performance on the details of the

image contour edge, which is helpful for the subsequent segmentation and identification of cracks.

3.2 Future Outlook

This section will introduce the future outlook for the follow-up of this research, as follows:

1. This study uses 448 x 448 pixel images for machine learning model training and virtual experiments to explore the feasibility of this research process. Using this resolution image for image measurement, the current experimental results show that the accuracy may be slightly insufficient. In the future, higher resolution images can be used for testing to see if the accuracy problem can be improved by using high resolution images.

Reference

Cha, Y. J., Choi, W., & Büyüköztürk, O. (2017). Deep learning−based crack damage detection using convolutional neural networks. *Computer−Aided Civil and Infrastructure Engineering,* 32(5), 361-378.

Cheng, L., Xin, H., Groves, R. M., & Veljkovic, M. (2021). Acoustic emission source location using Lamb wave propagation simulation and artificial neural network for I-shaped steel girder. *Construction and Building Materials,* 273, 121706.

Ji, X., Miao, Z., & Kromanis, R. (2020). Vision-based measurements of deformations and cracks for RC structure tests. *Engineering Structures,* 212, 110508.

Kim, H., Sim, S. H., & Spencer, B. F. (2022). Automated concrete crack evaluation using stereo vision with two different focal lengths. *Automation in Construction,* 135, 104136.

OpenCV Official Website. OpenCV. (2022, December 21). Retrieved December 28, 2022, from https://opencv.org/

Rozantsev, A., Lepetit, V., & Fua, P. (2015). On rendering synthetic images for training an object detector. *Computer Vision and Image Understanding,* 137, 24-37.

Tian, Y., Li, X., Wang, K., & Wang, F. Y. (2018). Training and testing object detectors with virtual images. *IEEE/CAA Journal of Automatica Sinica,* 5(2), 539-546.

Yang, Y. S., Chang, C. H., & Wu, C. L. (2019). Damage indexing method for shear critical tubular reinforced concrete structures based on crack image analysis. *Sensors,* 19(19), 4304.

Appendix

The appendix is the python program of the algorithm proposed in this study.

```
import numpy as np
import cv2
from matplotlib import pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from scipy.optimize import least_squares
from numba import njit

# 讀取左右相機影像
left_image = cv2.imread(r'D:\Lab\111-
2\computer_vision_and_image_measurement\test_crack_image_L.png')
right_image = cv2.imread(r'D:\Lab\111-
2\computer_vision_and_image_measurement\test_crack_image_R.png')

# 讀取左右相機 Mask 影像
left_mask = cv2.imread(r'D:\Lab\111-
2\computer_vision_and_image_measurement\test_crack_mask_L_binary.png')
right_mask = cv2.imread(r'D:\Lab\111-
2\computer_vision_and_image_measurement\test_crack_mask_R_binary.png')

# 創建特徵檢測器
orb = cv2.ORB_create()

# 在左右影像中檢測特徵點和描述符
left_keypoints, left_descriptors = orb.detectAndCompute(left_image, None)
right_keypoints, right_descriptors = orb.detectAndCompute(right_image, None)

# 創建特徵匹配器
bf = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)

# 進行特徵匹配
matches = bf.match(left_descriptors, right_descriptors)

# 根據特徵點匹配的距離進行排序
matches = sorted(matches, key=lambda x: x.distance)
```

```
# 選擇前 n 個最佳匹配
n = 100
best_matches = matches[:n]

# 提取最佳匹配的特徵點在左右影像中的位置
left_points = np.float32([left_keypoints[m.queryIdx].pt for m in
best_matches]).reshape(-1, 1, 2)
right_points = np.float32([right_keypoints[m.trainIdx].pt for m in
best_matches]).reshape(-1, 1, 2)

min_distance = 10000
max_distance = 0
for x in best_matches:
    if x.distance < min_distance:
        min_distance = x.distance
    if x.distance > max_distance:
        max_distance = x.distance

filtered_matches = []
for x in best_matches:
    if x.distance <= max(2 * min_distance, 30):
        filtered_matches.append(x)

# 繪製篩選後的匹配結果
outimage = cv2.drawMatches(left_image, left_keypoints, right_image,
right_keypoints, filtered_matches, outImg=None)
#plt.imshow(outimage[:,:,::-1])
#plt.show()

'''
# CAM L ##############################################################
K
<Matrix 3x3 (622.2222,   0.0000, 224.0000)
            (  0.0000, 622.2222, 224.0000)
            (  0.0000,   0.0000,   1.0000)>
RT
<Matrix 3x4 (0.9806, -0.1961,  0.0000, -0.0000)
```

```
            (0.0000,  0.0000, -1.0000,  0.0000)
            (0.1961,  0.9806,  0.0000,  7.6485)>

# CAM R ##############################################################
K
<Matrix 3x3 (622.2222,   0.0000, 224.0000)
            (  0.0000, 622.2222, 224.0000)
            (  0.0000,   0.0000,   1.0000)>
RT
<Matrix 3x4 ( 0.9806, 0.1961,  0.0000, 0.0000)
            ( 0.0000, 0.0000, -1.0000, 0.0000)
            (-0.1961, 0.9806,  0.0000, 7.6485)>
'''
# 定義左右相機的內部參數和外部參數
focal_length1 = 622.2222
principal_point1 = (224,224)
focal_length2 = 622.2222
principal_point2 = (224,224)

camera_matrix1 = np.array([[focal_length1, 0, principal_point1[0]],
                           [0, focal_length1, principal_point1[1]],
                           [0, 0, 1]])
camera_matrix2 = np.array([[focal_length2, 0, principal_point2[0]],
                           [0, focal_length2, principal_point2[1]],
                           [0, 0, 1]])

# 假設你已經有兩台相機各自的旋轉矩陣和平移向量
rotation_matrix1 = np.array([[0.9806, -0.1961, 0.0000],
                             [0.0000, 0.0000, -1.0000],
                             [0.1961, 0.9806, 0.0000]])
translation_vector1 = np.array([[-0.0000],
                                [0.0000],
                                [7.6485]])

rotation_matrix2 = np.array([[0.9806, 0.1961,  0.0000],
                             [0.0000, 0.0000, -1.0000],
                             [-0.1961, 0.9806, 0.0000]])
translation_vector2 = np.array([[0.0000],
```

```
                                [0.0000],
                                [7.6485]])

# 計算外部參數矩陣 [R | t]
RT1 = np.hstack((rotation_matrix1, translation_vector1))
RT2 = np.hstack((rotation_matrix2, translation_vector2))

# 計算投影矩陣 P
P1 = np.dot(camera_matrix1, RT1)
P2 = np.dot(camera_matrix2, RT2)

# 進行三角測量
points_4d_homogeneous = cv2.triangulatePoints(P1, P2, left_points,
right_points)

# 將齊次座標轉換為三維座標形式
points_3d = cv2.convertPointsFromHomogeneous(points_4d_homogeneous.T)

points_3d_list = []
# 印出三維座標
# points_3d_list = [np.array(2D_coord), np.array(3D_coord)]
for i, point in enumerate(points_3d):
    #print(f"Point {i+1}: {point.flatten()}")
    points_3d_list.append([left_points[i], point.flatten()])

# 取 Y 向的值
def cal_3D_points(image_points, rvec, tvec, camera_matrix, dist_coeffs,
nearest_leftimagePoint):
    # 定義投影誤差函式
    def reprojection_error(params, rvec, tvec, camera_matrix, dist_coeffs,
image_points):
        # 提取優化參數
        x, y, z = params

        # 將三維點位轉換維相機座標系下的座標
        camera_point = np.array([x, y, z])

        # 進行投影變換，將三維點位投影到圖向平面
```

```
        image_point_projected, _ = cv2.projectPoints(camera_point, rvec, tvec, camera_matrix, dist_coeffs)

        # 計算投影誤差
        error = np.linalg.norm(image_points - image_point_projected)

        return error

    x = image_points[0][0]
    y = image_points[0][1]
    # 初始化優化參數
    initial_params = np.array([x, 0, y], dtype=np.float32)
    # Y bounds
    points_3d_Y = points_3d[:, 0, 1]
    # 算最大值及最小值
    max_Y = np.max(points_3d_Y)
    min_Y = np.min(points_3d_Y)
    # 設定參數上下限(主要要限制 Y 向變化)
    lower_bounds = [-1000, 1.5 * min_Y , -1000]
    upper_bounds = [1000, 1.5 * max_Y, 1000]
    bounds = (lower_bounds, upper_bounds)

    # 用 least square 進行優化
    result = least_squares(reprojection_error, initial_params, args=(rvec1, tvec1, camera_matrix, dist_coeffs, image_points), bounds=bounds)

    # 取得優化後的三維點位
    optimized_params = result.x
    x_opt, y_opt, z_opt = optimized_params

    objpointCalFromImage = np.array([x_opt, y_opt, z_opt], dtype=float)
    return objpointCalFromImage

@njit
# Find crack pixel from L_camera segmentation image
def findcrackpixel(seg_img):
    crack_pixel = []
    for i in range(seg_img.shape[0]):
```

```
        for j in range(seg_img.shape[1]):
            if np.all(seg_img[i, j] == np.array([255, 255, 255])):
                crack_pixel.append([i, j])
    return crack_pixel

crack_pixel = findcrackpixel(left_mask)

def findNearestPointsFromFeature(image_points, points_3d_list):
    min_distance = 1e15
    min_distance_index = -1

    for i in range(len(points_3d_list)):
        distance = np.linalg.norm(image_points - points_3d_list[i][0])

        if distance < min_distance:
            min_distance = distance
            min_distance_index = i

    if min_distance_index != -1:
        min_distance_point_2d = points_3d_list[min_distance_index][0]
        min_distance_point_3d = points_3d_list[min_distance_index][1]

        minDisPointList = [min_distance, min_distance_point_2d,
min_distance_point_3d]

    return minDisPointList


# 已知的相機內外參及畸變參數
camera_matrix = camera_matrix1
rvec1 = rotation_matrix1
tvec1 = translation_vector1
dist_coeffs = np.zeros(5, dtype=float)

# 已知的二維點位
crack3Dinfo = []
for i, p in enumerate(crack_pixel):
    image_points = np.array([[p[1], p[0]]], dtype=np.float32)
```

```
    nearest_leftimagePoint = findNearestPointsFromFeature(image_points, 
points_3d_list)
    objectpoints = cal_3D_points(image_points, rvec1, tvec1, camera_matrix, 
dist_coeffs, nearest_leftimagePoint)
    crack3Dinfo.append(objectpoints)
    print("Point "+str(i+1)+"/"+str(len(crack_pixel))+" 3D Coordinates : ", 
objectpoints)

# 建立三維圖形
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

# 迴圈畫點(3D)
for point in crack3Dinfo:
    x, y, z = point
    ax.plot([x], [y], [z], marker='o', color='r')

# 設定坐標軸
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')

# 繪圖
    plt.show()
```