OKLAHOMA TRANSPORTATION CENTER

ECONOMIC ENHANCEMENT THROUGH INFRASTRUCTURE STEWARDSHIP

AUTOMATED AND ACCURATE BRIDGE DECK CRACK INSPECTION AND MAPPING

WEIHUA SHENG, PH.D.
JIANJUN GE, PH.D.
TYLER LEY, PH.D., P.E.
ROBERT EMERSON, PH.D.,P.E.

OTCREOS10.1-43-F

Oklahoma Transportation Center
2601 Liberty Parkway, Suite 110
Midwest City, Oklahoma 73110

Phone: 405.732.6580
Fax: 405.732.6586
www.oktc.org

Technical Report Data Page

1. REPORT NO. OTCREOS10.1-43-F	2. GOVERNMENT ACCESSION NO.	3. RECIPIENTS CATALOG NO.
4. TITLE AND SUBTITLE Automated and Accurate Bridge Deck Crack Inspection and Mapping		5. REPORT DATE October 31, 2012
		6. PERFORMING ORGANIZATION CODE
7. AUTHOR(S) Weihua Sheng, Jianjun Ge, Tyler Ley and Robert Emerson		8. PERFORMING ORGANIZATION REPORT
9. PERFORMING ORGANIZATION NAME AND ADDRESS School of Electrical and Computer Engineering Oklahoma State University 202 Engineering South Stillwater, OK 74078		10. WORK UNIT NO.
		11. CONTRACT OR GRANT NO. DTRT06-G-0016
12. SPONSORING AGENCY NAME AND ADDRESS Oklahoma Transportation Center (Fiscal) 201 ATRC Stillwater, OK 74078 (Technical) 2601 Liberty Parkway, Suite 110 Midwest City, OK 73110		13. TYPE OF REPORT AND PERIOD COVERED Final Aug 2010- Oct 2012
		14. SPONSORING AGENCY CODE
15. SUPPLEMENTARY NOTES University Transportation Center		

16. ABSTRACT

One of the important tasks for bridge maintenance is bridge deck crack inspection. Traditionally, a human inspector detects cracks using his/her eyes and finds the location of cracks manually. Thus the accuracy of the inspection result is low due to the subjective nature of human cognition. We propose a system that uses a mobile robot to conduct the crack inspection, where the robot collects bridge deck images with a video camera. In this method, the Laplacian of Gaussian algorithm is used to detect cracks and the global crack map is obtained through camera calibration and robot localization. To ensure that the robot collects all the images on the bridge deck, we develop a complete coverage path planning algorithm for the mobile robot. We compare it with other path planning strategies. Finally, we validate our proposed crack inspection system through both indoor and outdoor experiments. The proposed research can significantly improve the accuracy of bridge deck inspection in an efficient way. Timely awareness of bridge structural health can make our transportation system much safer while man power will be significantly saved.

7. KEY WORDS Crack inspection, bridge deck, mobile robot, mapping	18. DISTRIBUTION STATEMENT No restrictions. This publication is available at www.oktc.org and from the NTIS.		
19. SECURITY CLASSIF. (OF THIS REPORT) Unclassified	20. SECURITY CLASSIF. (OF THIS PAGE) Unclassified	21. NO. OF PAGES 68 + covers	22. PRICE

SI (METRIC) CONVERSION FACTORS

Approximate Conversions to SI Units

Symbol	When you know	Multiply by	To Find	Symbol
		LENGTH		
in	inches	25.40	millimeters	mm
ft	feet	0.3048	meters	m
yd	yards	0.9144	meters	m
mi	miles	1.609	kilometers	km
		AREA		
in^2	square inches	645.2	square millimeters	mm
ft^2	square feet	0.0929	square meters	m^2
yd^2	square yards	0.8361	square meters	m^2
ac	acres	0.4047	hectares	ha
mi^2	square miles	2.590	square kilometers	km^2
		VOLUME		
fl oz	fluid ounces	29.57	milliliters	mL
gal	gallons	3.785	liters	L
ft^3	cubic feet	0.0283	cubic meters	m^3
yd^3	cubic yards	0.7645	cubic meters	m^3
		MASS		
oz	ounces	28.35	grams	g
lb	pounds	0.4536	kilograms	kg
T	short tons (2000 lb)	0.907	megagrams	Mg
		TEMPERATURE (exact)		
°F	degrees Fahrenheit	(°F-32)/1.8	degrees Celsius	°C
		FORCE and PRESSURE or STRESS		
lbf	poundforce	4.448	Newtons	N
lbf/in^2	poundforce per square inch	6.895	kilopascals	kPa

Approximate Conversions from SI Units

Symbol	When you know	Multiply by	To Find	Symbol
		LENGTH		
mm	millimeters	0.0394	inches	in
m	meters	3.281	feet	ft
m	meters	1.094	yards	yd
km	kilometers	0.6214	miles	mi
		AREA		
mm^2	square millimeters	0.00155	square inches	in^2
m^2	square meters	10.764	square feet	ft^2
m^2	square meters	1.196	square yards	yd^2
ha	hectares	2.471	acres	ac
km^2	square kilometers	0.3861	square miles	mi^2
		VOLUME		
mL	milliliters	0.0338	fluid ounces	fl oz
L	liters	0.2642	gallons	gal
m^3	cubic meters	35.315	cubic feet	ft^3
m^3	cubic meters	1.308	cubic yards	yd^3
		MASS		
g	grams	0.0353	ounces	oz
kg	kilograms	2.205	pounds	lb
Mg	megagrams	1.1023	short tons (2000 lb)	T
		TEMPERATURE (exact)		
°C	degrees Celsius	9/5+32	degrees Fahrenheit	°F
		FORCE and PRESSURE or STRESS		
N	Newtons	0.2248	poundforce	lbf
kPa	kilopascals	0.1450	poundforce per square inch	lbf/in^2

ACKNOWLEDGMENTS

The authors gratefully thank and acknowledge the financial support provided by the Oklahoma Transportation Center (OkTC) and the RITA University Transportation Center Program. The authors would also like to thank the Oklahoma State University Police Department for providing the road cones that enable the field tests of the project.

Automated and Accurate Bridge Deck Crack Inspection and Mapping

Final Report

Weihua Sheng, Ph.D
Principal Investigator

JianjunGe, Ph.D
Tyler Ley, Ph.D, P.E.
Robert Emerson, Ph.D, P.E.
Co-Principal Investigators

School of Electrical and Computer Engineering
School of Civil and Environmental Engineering
Department of Geography

Oklahoma State University

Stillwater, OK 74078

October 2012

Table of Contents

List of Figures

List of Tables

EXECUTIVE SUMMARY

The purpose of this report is to present the results of our research and development of the Robotic Crack Inspection and Mapping (ROCIM) system, which is a mobile robot equipped with advanced sensors such as cameras, laser range finders, and innovative software algorithms to conduct accurate crack inspection and mapping for bridge decks.

This project is conducted by a team of interdisciplinary researchers with expertise in the area of mobile robotics, remote sensing and civil engineering. The project consists of three major tasks: 1) the development of the ROCIMplatform; 2) the development of robot autonomous navigation (localization and path planning) for inspection; 3) the development of crack detection and crack map generation.

This project improves the accuracy and efficiency of bridge deck inspection during maintenance. The results show that we can detect cracks with size of 0.016inch given proper high resolution cameras are used and the inspection time for a typical bridge deck is about half an hour. Timely awareness of bridge structural health will make our transportation system much safer while man power will be significantly saved. Such a system improves the state of the art of bridge deck structural inspection. The developed system can also be applied in a much wider spectrum of structure health monitoring (SHM) applications.

The proposed work addresses the safety of transportation systems with a focus on bridges. Due to the ubiquitousness of bridges, such a project has great impact at all levels of the transportation system, including global, national and local communities. The goal of this project clearly reflects part of the OTC's mission–"...to provide world-class solutions to challenges in building, designing and maintaining transportation systems". Most importantly, this project has its great impact to the economic growth of the State of Oklahoma and the nation as well. Bridges are an important part of the interstate highway system and essential to the continuous functioning of the latter. Improved bridge safety and reduced maintenance time imply higher productivity and efficiency of our interstate highway system, which are vital to the economic vitality of the state and the nation. This project paves the way to build advanced, automated bridge inspection system which will overcome the limitations inherited in existing inspection technologies.

CHAPTER 1

INTRODUCTION

1.1 Motivation

For many engineered transportation structures, including civil, mechanical and aerospace structures, timely awareness of their structural health can prevent functional failures which may lead to catastrophic consequences. On August 1, 2007, the collapse of the I-35W Mississippi River Bridge (See Figure 1.1) that carried Interstate 35W across the Mississippi River in Minneapolis left 13 dead and more than 100 injured [3], not to mention its big impact on the traffic and business in the surrounding areas. This accident has clearly demonstrated the catastrophic results of structural failures and the importance of timely awareness of structural health. Bridge decks are typically the elements of first maintenance on a bridge. Since the surface of a bridge deck is exposed to the environment, direct loading from vehicles and exposure to deicing chemicals, constant maintenance is a must. Therefore bridge deck inspection is helpful and can provide owners warning to the future deterioration of the bridge deck. Currently bridge decks are inspected with very rudimentary methods in the form of visual inspection by a trained engineer as shown in Figure 1.2. While these inspections are useful, they are only useful if the deterioration of the bridge deck is significant. The inspectors usually walk through the bridges and measure the crack locations and crack sizes. This kind of manual approach has the following disadvantages:

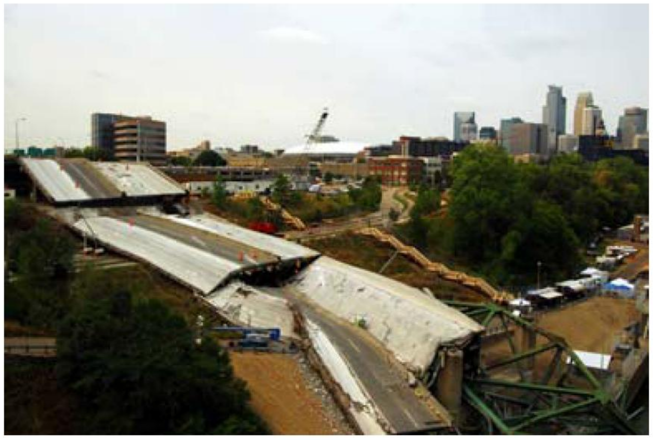


Figure 1.1. I-35W Bridge over the Mississippi River in Minneapolis collapsed.

1. It is prone to human errors.
2. It has very limited accuracy due to the limited visual capability of human inspectors.
3. It cannot guarantee the full coverage of the whole bridge deck.

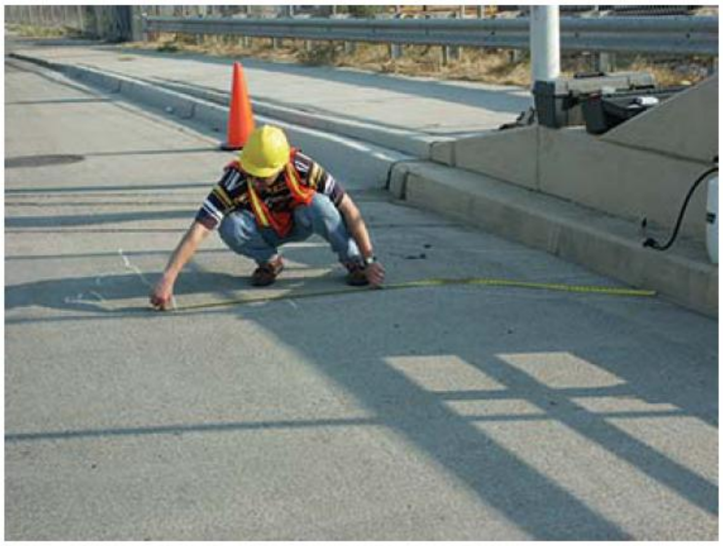


Figure 1.2 A human inspector measures cracks for bridge deck maintenance.

Additionally, conducting visual crack inspection of a bridge deck is a dangerous job with passing traffic. Therefore, we need to develop a robotic crack inspection and mapping (ROCIM) system which can conduct accurate assessment of cracking on bridge decks. The ROCIM system will have the following features which outperform human inspectors:

1. A high resolution camera can be mounted on the mobile robot to achieve high accuracy of crack detection.
2. The crack locations can be accurately determined since the robot can localize itself precisely.
3. Using mobile robots will reduce the inspection time and the chance of the loss of human life.

1.2 Objective

In this project, the PIs aim to develop a robotic crack inspection and mapping (ROCIM) system which can conduct accurate assessment of cracking on bridge decks. Such a ROCIM system can be deployed on the bridge deck and it can collect raw data in an automated way. The final output of the ROCIM system is a crack map indicating the location and dimension of detected cracks. The targeted minimum detectable crack width is 0.016 inch (0.41mm) and the inspection time for a medium-size bridge deck is about 10-30 minutes. The obtained crack map can be used for maintenance purpose. It can also be used in the bridge deck safety analysis so that the potential risks and their causes can be identified.

1.3 Challenges

In the ROCIM system, a mobile robot is utilized to create a two dimensional (2D) map of the bridge deck using a laser sensor and collect images of the bridge deck's surface using a camera. Then, the collected images will be processed using image processing techniques to detect cracks. We store the crack location in the global 2D map, which becomes a crack map. The crack map will be useful for the bridge maintenance such as measuring, classifying and monitoring cracks periodically. There are several challenging problems that we need to address in order to implement the ROCIM system.

1. The mobile robot localization: to create a 2D map of the bridge deck, the mobile robot needs to know its current position. Robot localization is a challenging problem in robotics research.
2. The coordinate transformation: to create a crack map, the crack location has to be plotted in a global coordinate system which is the same coordinate system as the 2D map of the bridge deck. Since the crack is detected in the image coordinate system, we have to map the crack location from the image coordinate system to the global coordinate system.
3. The path planning of mobile robot: the path planning should ensure the mobile robot can inspect the whole area of bridge deck's surface in the efficient way. Unlike a typical complete coverage path planning of a vacuum cleaner robot, which uses the robot body as the coverage, the mobile robot in ROCIM system is interested in the camera field of view as the coverage, which poses some difficulties in the path planning.

1.4 Related Work

Recent years have witnessed growing research interests in structural health monitoring (SHM) [4],[5],[6],[7],[8] for bridges, buildings and other civil infrastructures. Research in structure inspection using robotic devices has resulted in several prototypes. Yu et al. [9] presented an auto inspection system using a mobile robot for detecting concrete cracks in a tunnel. Their system consists of a mobile robot, a data storage system and a crack detector. The crack is detected using image processing and the crack information is extracted using Sobel and Laplacian operators. In addition to their system, an illuminator is used to distinguish the crack and non-crack areas. Their mobile robot system has a complex imaging system to reduce noise resulted from the mobility of mobile robot. Sinha et al. [10] developed a statistical filter for crack detection in pipes. A two-step algorithm is proposed. First, crack features are extracted from the segmented image. Two filter detectors are applied to extract the crack information. Then, both responses are merged to obtain a unique response. Second, apply cleaning and linking operation to form the global cracks. Tung et al. [11] proposed the development of a mobile manipulator imaging system for bridge crack inspection. The manipulator system is equipped with a binocular charge coupled devices (CCD) camera. Their system aims to replace human inspectors. Lee et al. [12] and Oh et al. [13] proposed a bridge inspection system which consists of a specially designed car, a robotic mechanism and a control system for automatic crack detection. Their system measures the length and width of the cracks. The measurement result is stored in the database which is utilized in the Bridge Management System. Sohn et al. [14] proposed an automatic procedure for monitoring crack growth on concrete structures. Their system focuses on quantifying the change of cracks from multi temporal images during the monitoring period. A modified iterated Hough transform (MIHT) is developed to solve a 2D projection between 2D concrete surface and 2D digital image [15]. Ito et al. [16] demonstrated an automatic measurement system for concrete block inspection by means of fine crack extraction. Most of these studies classify, measure, and detect cracks. However, none of these crack inspection studies finds the global location of cracks. In this project, we develop an overall system for robotic crack inspection. We detect cracks using a high resolution camera. Then we find the location of cracks in the global map. To ensure the completeness of the crack map, the inspection has to cover the whole area of the bridge deck surface. In other words, the complete coverage path planning must be applied for the ROCIM system.

There are several related works for complete coverage path planning using mobile robots. Choset et al. [17] [18] developed a complete coverage path planning using boustrophedron motion. The environment is decomposed into cells and the back and forth motion is applied for each cell. Two types of decomposition are discussed: boustrophedron decomposition and trapezoidal decomposition. The boustrophedron decomposition gives more efficient coverage path planning than trapezoidal decomposition because it has fewer cells to be covered. Muzaffer et al. [19] investigated a genetic algorithm to coverage path planning for mobile robots. The coverage is modeled as a disk which represents the range of sensing devices. The genetic algorithm is used to find an optimum or near optimum path in order to cover the disk at least once. Paulo et al. [20] utilized a genetic algorithm to find the most efficient coverage path in a static environment. The proposed algorithm decomposes the region into sub regions to separate the obstacle and non obstacle area. Their algorithm considers two common templates, zigzag and windowing, to cover in the sub region area. The fitness function of the genetic algorithm is defined based on distance and time. Simon et al. [21] developed a neural network approach to complete coverage path planning. The dynamic neural activity based on Hodgkin and Huxley's [22] shunting equation is used to develop the complete coverage path planning. The idea is unclean area attracts the mobile robot and the obstacle pushes the mobile robot away. The result shows that the number of robot turns and traveling distance are minimized. Most of these works consider the robot body as the coverage and the camera field of view (FoV) is not studied. In the ROCIM system we are interested in covering the camera FoV. There are challenges in the planning problem due to the configuration of the camera and the mobility of the robot. In addition, we present a solution to this problem based on genetic algorithms. The solution of complete coverage path planning (CCPP) is developed under the condition that the mobile robot knows their position. To solve this localization problem, we use Monte Carlo Localization algorithm [23]. We utilize an Advanced Range Navigation Laser (ARNL) from Mobilerobotcompany [1] to implement the MCL algorithm.

1.5 Report Organization

The rest of the report is organized as follows.Chapter 2 presents the ROCIM system overview. The camera calibration is discussed in Chapter 3. The camera calibration finds the intrinsic and extrinsic parameters that describe the geometric mapping between the 3D world and the 2D image. Chapter 4 presents the algorithm for crack detection. Chapter 5 discusses the complete coverage path planning for the ROCIM system. The path planning is developed using a genetic algorithm to ensure the complete coverage of camera field of view (FoV) and minimize the traveling distance and the number of robot turns. The experiment results for the overall ROCIM system and the complete coverage path planning are discussed in Chapter 6. Finally, conclusion and recommendations are given in Chapter 7.

CHAPTER 2

ROCIM SYSTEM DESIGN

In this chapter, we first introduce the hardware setup of our ROCIM system. The main hardware for the ROCIM system is developed based on a mobile robot. The robot basically travels in the inspection area and collects images of the bridge deck surface. Then, the main principle of the ROCIM system is presented which consists of the following steps: navigation, data collection and crack map generation.

2.1 Hardware setup of the ROCIM system

Inside the mobile robot, there is an on-board PC103 single board computer which has limited memory and processor. Therefore, we utilize a laptop computer to implement complicated algorithms such as cracks detection, localization, and path planning. Figure 2.1 shows the overall hardware setup of the ROCIM system which consists of:

• one Pioneer3-DX mobile robot

• one LMS-200 laser sensor

• one Pan-Tilt-Zoom (PTZ) Canon VC-C50i camera

• one laptop

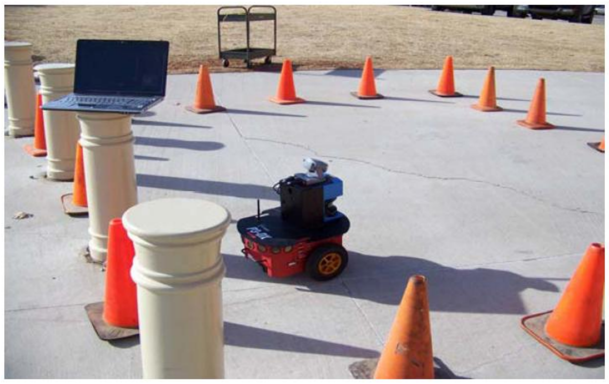


Figure 2.1 The ROCIM system in action.

The Pioneer3-DX is a two-wheel drive mobile robot and has an on-board computer. The processor of the on-board computer is Pentium III 533 Mhz with 128 MB RAM and also microprocessor is 44.2368 MHz Renesas SH2 32-bit RISC microprocessor with 32K RAM. The communication architecture of this mobile robot is based on a client and server communication through RS-232 protocol. In addition, the microprocessor handles the low-level devices of the mobile robot.

The laptop computer is HP G62-140US which has a 2.13GHz Intel Core i3-330 Processor and 4GB DDR3 System Memory. The laptop and the mobile robot are communicated via wireless technology through Transmission Control Protocol (TCP). The TCP data packet is converted into serial data to control the low level device in the on-board computer in order to communicate with the microprocessor. The illustration of the wireless communication is shown in Figure 2.2. In addition, the Ethernet access point can be removed by using peer to peer wireless communication between the laptop and the on-board computer.

The LMS-200 laser sensor is used to create the 2D map of the bridge deck and localize the robot during the inspection. This sensor has 1800 bearing angle and is able to estimate a range up to 15 meters [1]. In addition, the laser beam is highly focused and is not distorted by the reflecting medium. This device connects to the on-board computer through serial communication.

Another important device is the PTZ Canon VC-C50i camera, which is a color camera with a resolution of 860×640 and 26× zoom. The orientation of this camera can be changed and the ranges of the pan angle and tilt angle are -1000 degreeto 1000 degree,300 degreeto 900 degree, respectively. The camera produces high resolution image which can be used for crack detection in concrete environments.

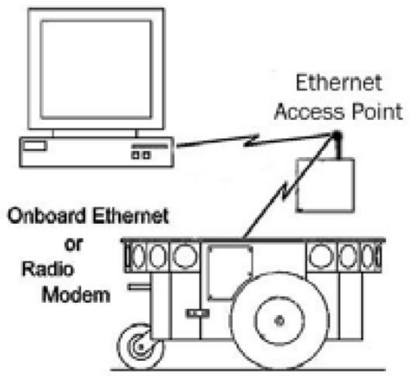


Figure 2.2 Connection setup between the laptop and the mobile robot [1].

2.2 Working principle of the ROCIM system

To conduct bridge deck crack inspection and mapping, as illustrated in Figure 2.3, we will block half of the bridge. The ROCIM system will then be deployed to inspect the blocked half of the bridge. Once completed, the traffic will be switched to the completed half and the other half will be inspected. During the ROCIM operation, the images of the mobile robot will get blurred due to the vibration caused by the passing traffic. To solve this problem, an anti-shock device can be used underneath the camera. Moreover, when the mobile robot collects the images, we assume the mobile robot completely stops in order to capture a clear image.

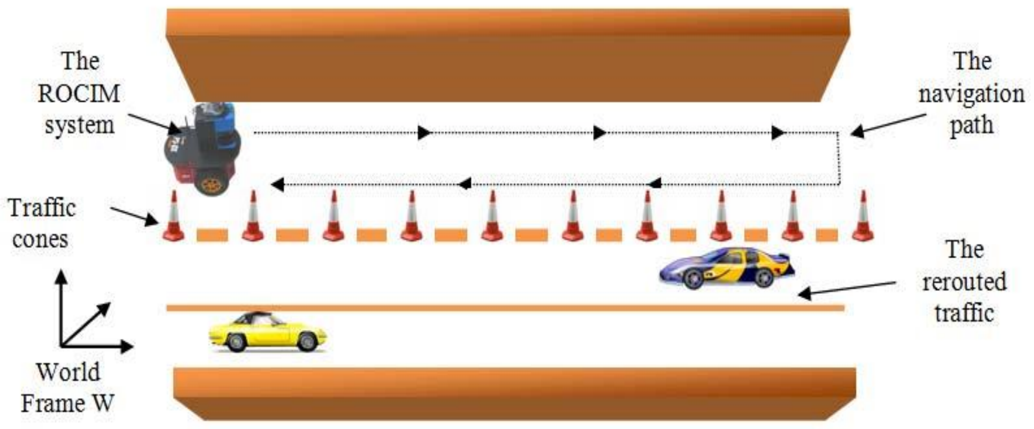


Figure 2.3 Scenario of crack inspection and mapping using the ROCIM system.

There are three steps to conduct crack inspection and mapping.

1. Navigation map building: A 2D bridge deck map will be created first, which will be used to localize the robot during the data collection step.

2. Data collection: The robot will navigate on the bridge deck to collect the image data at different locations. The raw image data can be stored in the on-board computer or transferred to a nearby laptop computer using wireless connection.

3. Crack detection and crack map generation: Crack will be detected through image processing. The crack map will be created by piecing together multiple local crack maps. This step can be performed off-line on the laptop computer.

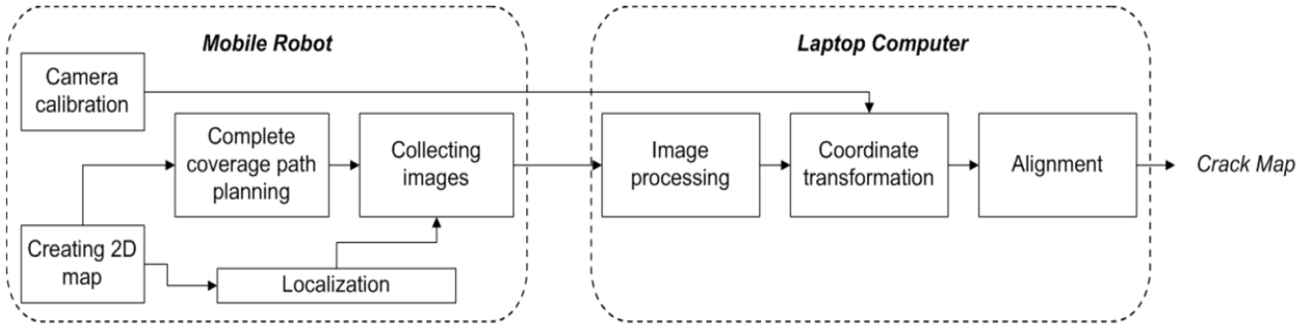


Figure 2.4 Principle of the ROCIM system.

The proposed crack inspection system works according to Figure 2.4. To solve the robot localization problem, we use simultaneous localization and mapping (SLAM) algorithm [24]. Here, the robot location is estimated using an odometry data and landmark information. The landmark information is extracted using a laser sensor. We utilize the Mapper3 software to create the 2D map [1]. After that, the mobile robot obtains a prior knowledge of the environment and it can localize itself based on that map using Monte Carlo Localization (MCL) algorithm [23].

Camera calibration is used to find the relationship between the image coordinate system and the robot coordinate system. We mount the camera on the top of the mobile robot. The position andorientation of the camera are assumed to be static relative to the mobile robot. Therefore, therelationship between the image and robot coordinate system is assumed to be fixed. The crack location in the image coordinate system can be mapped to the robot coordinate system through this relationship. During the inspection, the robot will travel on the bridge deck and localize itself according to the 2D map. Therefore, the relationship between the robot and the global coordinate can be found using the MCL algorithm. In order to create the crack map, we compute the image-global coordinate transformation which is the multiplication of the image-robot and robot-global transformation.

After all images are collected, they will be processed using the Laplacian of Gaussian (LoG) algorithm [25] [26] to detect cracks. The LoG algorithm finds a zero crossing as the edge in the second differential of image intensity. Here, the crack location is in the image coordinate system and we apply an image-global coordinate transformation to map this crack location to the global coordinate system.

To ensure the mobile robot collects all the images on the bridge deck in the efficient way, we solve the Complete Coverage Path Planning (CCPP) problem for the ROCIM system. We propose a an algorithm for Robotic Inspection Path planning based on the Genetic Algorithm (RIP-GA) approach to solve this complete coverage path planning problem.The RIP-GA algorithm will be run off-line after the 2D map of the bridge deck is generated. The output of RIP-GA is basically a path which consists of a sequence of locations in the 2D map with a particular camera pose. The mobile robot has to travel according to this path and collect an image at each location. In the last part of the ROCIM system, there is an alignment process to link the continuous crack in different images. Since the calibration and localization have an error due to distortion and noise, this alignment process can be used to improve the accuracy of the crack map.

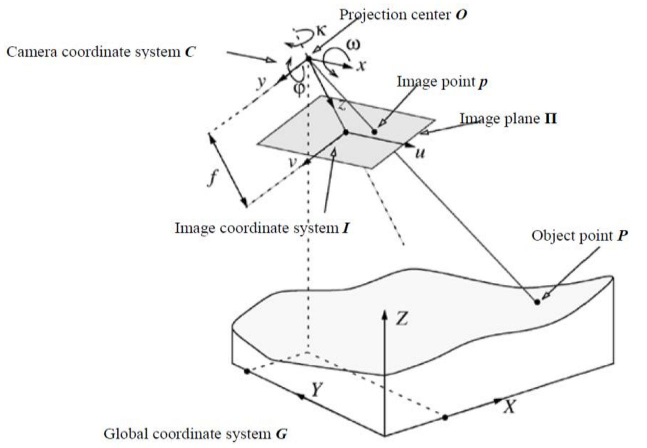
CHAPTER 3

CAMERA CALIBRATION

In the ROCIM system, a mobile robot detects the crack in the image coordinate system and the crack map should be created in the global coordinate system. Hence, camera calibration is needed to map the location from the image coordinate system to the global coordinate system. We need to find a transformation matrix to map the crack location. In this section, first, we discuss the pin-hole camera model for camera calibration. Second, a Matlab camera calibration toolbox is introduced to solve the camera calibration problem. Last, we discuss the overall coordinate transformation to create the crack map.

3.1 Camera model

Camera calibration is a process of geometric mapping between the 3D world and 2D images using a set of points. The geometric mapping consists of extrinsic and intrinsic parameters. Figure 3.1 illustrates the pure perspective projection of a pin hole camera model. Here, we have a point P in the global coordinate system and the center of the projection is at the origin O of the camera coordinate system C. The image planeis parallel $\pi$to the xyplane of the camera coordinate system with displacement of focal length (f) along the z axis. The principal point (o) is an intersection of planeπ and the z axis. The coordinates of the principal point in the image coordinate system I are [u0, v0]^T.


Projection center O
Camera coordinate system C
κ
ω
x
y
φ
z
Image point p
Image plane Π
u
v
f
Image coordinate system I
Object point P
Z
Y
X
Global coordinate system G
Figure 3.1 Illustration of pinhole camera model [2].

Let P be an arbitrary 3D point located in the global coordinate system $[X, Y, Z]^T$. The coordinate of P in the camera coordinate system is $[x, y, z]^T$. The coordinate of p in the image coordinate system is $[u, v]^T$. The following homogeneous equation can be used to express the transformation of these coordinate systems.

$$\begin{matrix} u \\ v \\ 1 \end{matrix} \; \alpha \; \begin{matrix} \lambda u \\ \lambda v \\ \lambda \end{matrix} = \mathbf{F} \begin{matrix} X \\ Y \\ Z \\ 1 \end{matrix} = \mathbf{LM} \begin{matrix} X \\ Y \\ Z \\ 1 \end{matrix} \tag{3.1}$$

where**F** is the perspective transformation, λ is a scale factor, **L** describes the intrinsic matrix, and **M** denotes the mapping from the global coordinate system to the camera coordinate system which consists of rotation **R** and translation **t**. The decomposition of matrix **L** and **M** are shown in Equation 3.2 and Equation 3.3, respectively.

$$\mathbf{L} = \begin{matrix} sf & 0 & \mu_0 & 0 \\ 0 & f & \nu_0 & 0 \\ 0 & 0 & 1 & 0 \end{matrix} \tag{3.2}$$

$$\mathbf{M} = \begin{matrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{matrix} \tag{3.3}$$

where**t** = $[tx, ty, tz]^T$ and **R** can be defined by the three Euler angles θ, φ, and ψ. These angles can be computed using the rotation matrix **R.**

$$\theta = sin^{-1} r_{31} \tag{3.4}$$

$$\phi = atan2\left(-\frac{r_{32}}{cos\theta}, \frac{r_{33}}{cos\theta}\right) \tag{3.5}$$

$$\psi = atan2\left(-\frac{r_{21}}{cos\theta}, \frac{r_{11}}{cos\theta}\right) \tag{3.6}$$

The variables t_x,t_y,t_z,θ,φ, and ψ are called extrinsic parameters and s, f,u_0, and v_0 are denoted as intrinsic parameters.

Figure 3.1 shows the ideal projection of a point in the global coordinate system to the image coordinate system. In practice, there will be an error (distortion) caused by the lens system. Many research works have been proposed to deal with this problem. The main approach is to use a decomposition of the distortion into radial and decentering components [2]. Here, we utilize the Matlab camera calibration toolbox to conduct the camera calibration. The interface of this toolbox is shown in Figure 3.2.

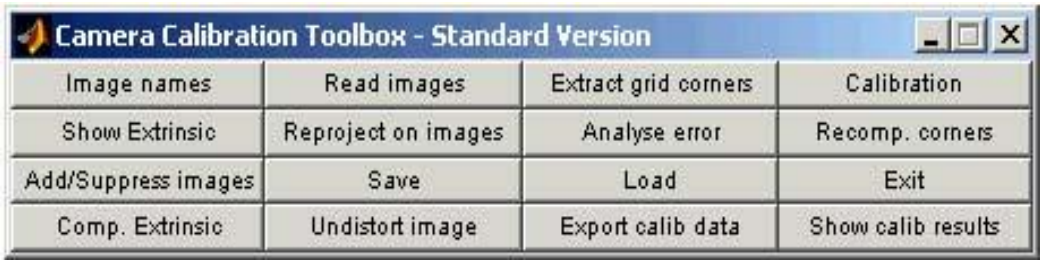


Figure 3.2: Matlab camera calibration toolbox.

3.2 Matlab camera calibration toolbox

In this section, we briefly present the overview of Matlab camera calibration toolbox to solve the camera calibration problem for the ROCIM system. The input of this toolbox is a set of n points in the image and the world coordinate system where $n > 5$. The output is the intrinsic and extrinsic parameters of the camera calibration. The procedure of the calibration in our experiment is described as follows:

1. Setup the experiment as shown in Figure 3.3. A chessboard is used as the object for the calibration. Since the grid has the same length but different colors for neighboring grid cells, the location of the n points can be estimated accurately.
2. Capture an image of the chessboard using the mobile robot camera as shown in Figure 3.4(a). This image will be used as the input for the toolbox.
3. Extract the grid corners to find the set of n points in the image coordinate system and the locations of these points in the global coordinate system are calculated. Figure 3.4(b) shows the

result of the corner extraction.

4. Run the calibration. The results of the calibration are the intrinsic parameters and the extrinsic parameters which are shown in Figure 3.5(a) and Figure 3.5(b).

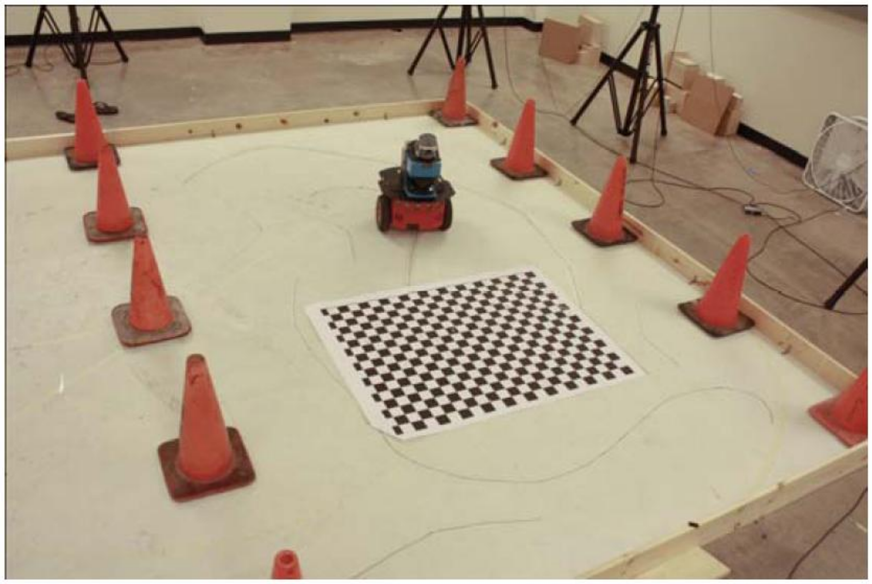
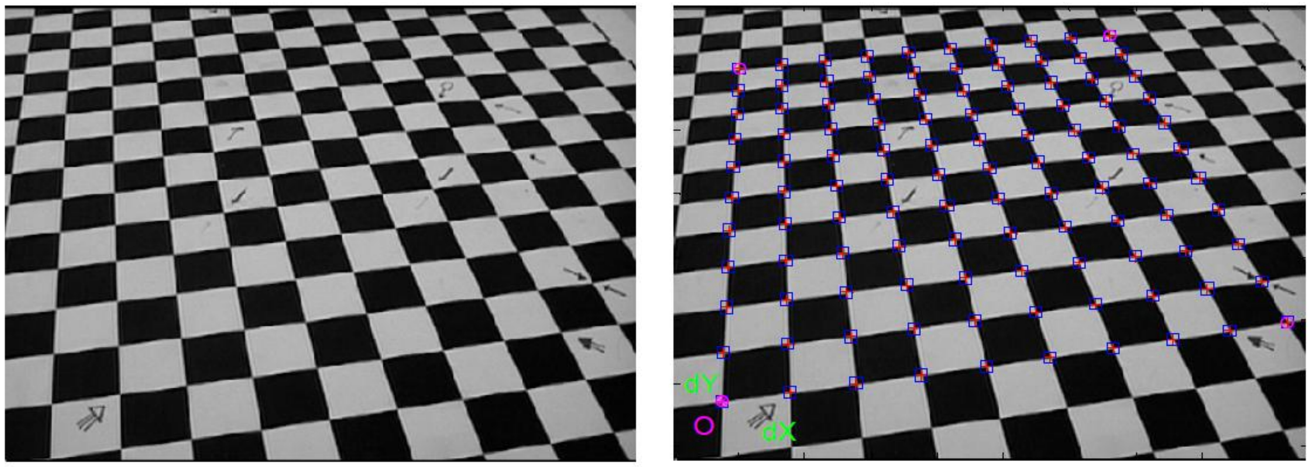


Figure 3.3 Camera calibration for the ROCIM system.



(a) (b)
Figure 3.4 A chessboard is used for camera calibration. (a): image for camera calibration. (b): Corner extraction result.

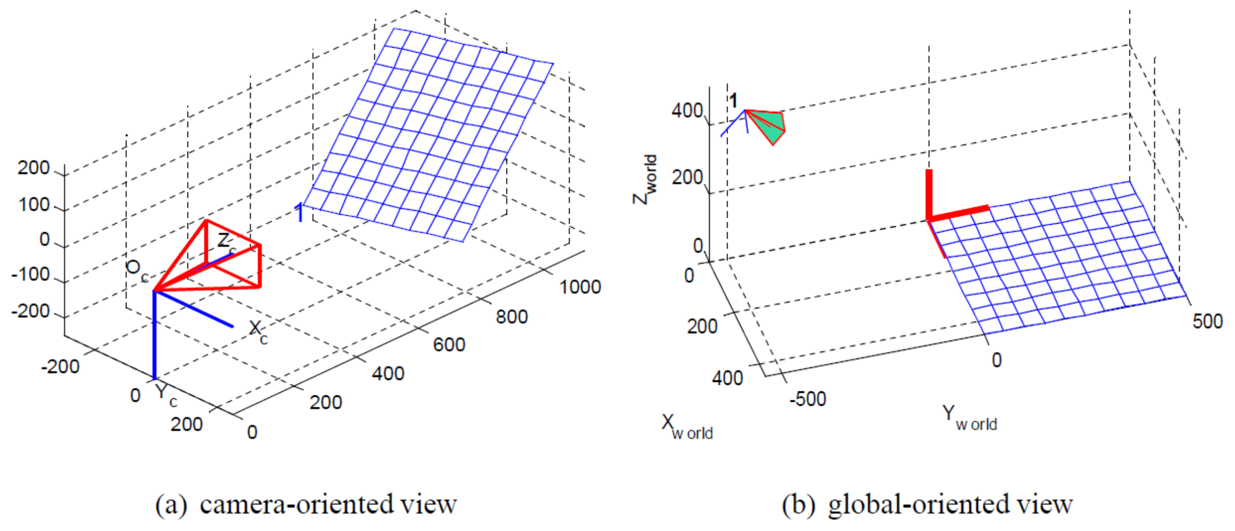


Figure 3.5 Illustration of the relationship between the camera coordinate system and theglobal coordinate system.

3.3 Coordinate transformation

In this section we discuss the coordinate transformation to map the crack location from the image coordinate system to the global coordinate system. As we mentioned before that the mobile robot travels in the inspection area and collects the image of the bridge deck surface. The location and orientation of the mobile robot can be estimated using the Monte Carlo Localization (MCL) algorithm [27] [23]. The MCL algorithm is a sampling-based method to approximate a probability density distribution of robot location. Each sample consists of a possible robot locations and a probability that the robot currently is at that location. We use Advanced Range Navigation Laser (ARNL) to implement the MCL algorithm [1]. Figure 3.6 shows a graphic user interface of mobile eyes software 3.6. The red circle represents the mobile robot location and orientation. The green and blue dot denote previous and current samples, respectively. These samples are randomly chosen using a laser sensor. The information about the current position, orientation and a value of probability for the current state can be seen in the bottom label.

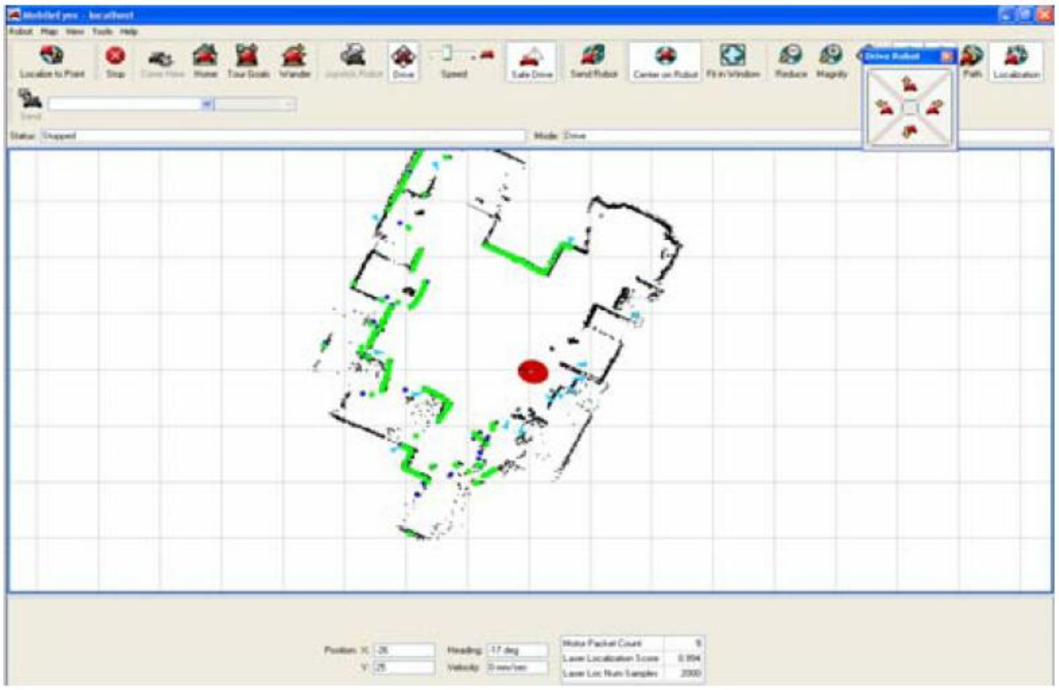


Figure 3.6 Graphic user interface for mobile eyes.

Camera calibration finds the extrinsic and the intrinsic parameters of the camera withrespect to the mobile robot. Subsequently, we utilize those parameters to map all cracklocations to the 2D global map by assuming that the transformation between the cameracoordinate system and the robot coordinate system is fixed. The ROCIM system has five coordinate systems as shown in Figure 3.7. They are: image coordinate system (F_I), camera coordinate system (F_C), local coordinate system (F_L), robot coordinate system (F_R) and global coordinate system (F_G). After we do the camera calibration, we have the intrinsic and the extrinsic parameters. The intrinsic parameters *are focal length (f)*, skew value (s) and the origin of image coordinate system (μ_0,v_0) as described in Equation (3.2). The extrinsic parameters are rotation (**R**) and translation (**t**) matrices as in Equation (3.3) [2]. These parameters define the transformation matrix ${}^{I}T_C$ and ${}^{C}T_L$, respectively. In the ROCIM system as the robot collects the images, it saves its current location in the global coordinate system. After we detect the crack in each image, we should plot all the crack locations in the global coordinate system. To do this, we have the crack locations in the image coordinate system (u,v). We map the crack locations to the robot coordinate system (X_R,Y_R) using the camera-robot transformation ${}^{C}T_R$, which is derived from the multiplication of transformation matrix as below.

$$C_{T_R} = C_{T_L} L_{T_R} \tag{3.7}$$

The transformation ${}^{L}T_{R}$ is estimated using a set of points in optical tracking system [28]. We compute the location of each point for local and robot coordinate system. Then, we calculate the transformation matrix using pseudo-inverse based on these locations.

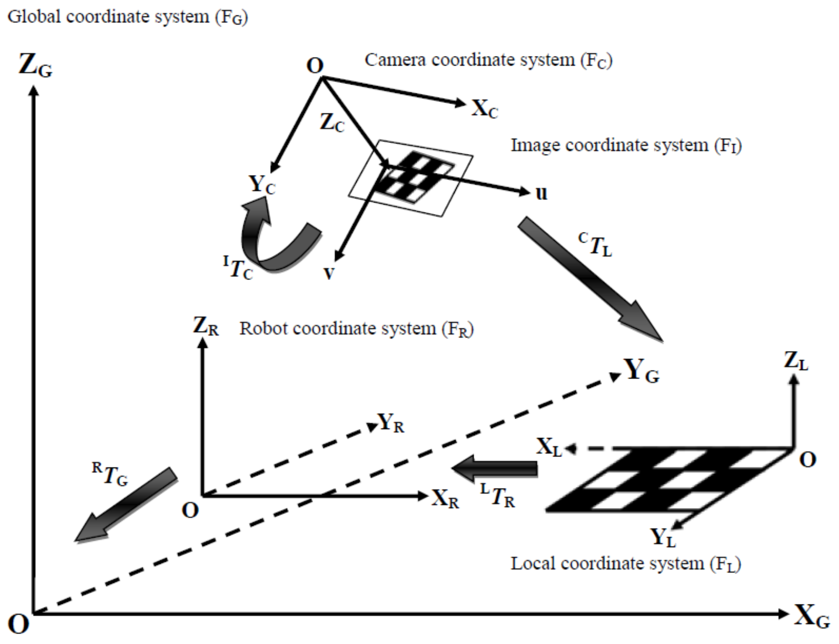

Global coordinate system (F_G)
Z_G
O
Camera coordinate system (F_C)
X_C
Z_C
Image coordinate system (F_I)
Y_C
u
^C T_L
^I T_C
v
Z_R
Robot coordinate system (F_R)
Y_G
Z_L
Y_R
X_L
^R T_G
O
X_R
^L T_R
O
Y_L
Local coordinate system (F_L)
O
X_G

Figure 3.7 Coordinate systems in the ROCIM system.

We apply another 2D transformation from the robot coordinate system to the global coordinate system using the transformation ${}^{G}T_{R}$, which can be calculated from the robot location (x,y,θ). Therefore from the image coordinate system to the global coordinate system, we have the following transformation:

$${}^{I}T_{G} = {}^{R}T_{G}\ {}^{C}T_{R}\ {}^{I}T_{C} \tag{3.8}$$

CHAPTER 4

CRACK DETECTION

In this chapter, we describe how to detect cracks in each image captured by the mobile robot. The Laplacian of Gaussian (LoG) algorithm is used to detect edges which are the cracks. Then, we discuss our graphic user interface to detect the cracks.

4.1 Crack detection algorithm

The main idea of detecting cracks is to find out the edge points in an image. These edge points can be detected by finding the zero crossings of the second derivative of the image intensity. However, calculating the second derivative is very sensitive to noise. Hence, this noise should be filtered out. To achieve this, the Laplacian of Gaussian (LoG) algorithm is used [29]. This method combines Gaussian filtering with the Laplacian for edge detection. In the LoG edge detection, there are mainly three phases: filtering, enhancement, and detection. Especially, Gaussian filter is used for smoothing and its second derivative is used for enhancement. The detection criteria is the presence of a zero-crossing in the second derivative with the corresponding large peak in the first derivative as shown in Figure 4.1.

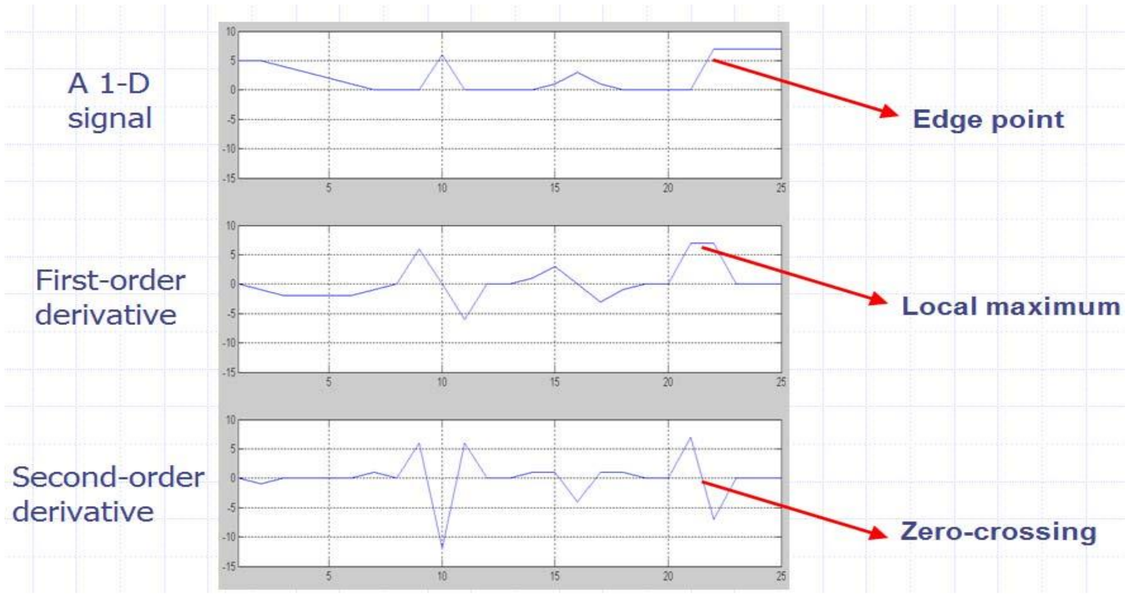


Figure 4.1 First and second order derivative of one dimension signal.

In this approach, the noise is firstly reduced by convoluting the image with a Gaussian filter. Then isolated noise points and small structures are filtered out. With smoothing, however, edges are spread out. Those pixels that have locally maximum gradient are considered as edges by the edge detector in which zero crossings of the second derivative are used. To avoid detecting of insignificant edges, only the zero crossings whose corresponding first derivative above some thresholds are selected as edge points. The edge direction is obtained by using the direction in which zero-crossing occurs. The Laplacian is a 2D isotropic measure of the 2nd spatial derivative of an image. The Laplacian of an image highlights regions of fast intensity change and is often used for edge detection. The Laplacian is often applied to an image that has first been smoothed with a Gaussian smoothing filter in order to reduce its sensitivity to noise. Hence, the two variants will be described together here. The operator normally takes a single gray level image as input and produces another gray level image as output. The Laplacian L(x,y) of an image with pixel intensity values I(x,y) is given by:

$$L(x,y) = \frac{\delta^2 I}{\delta x^2} + \frac{\delta^2 I}{\delta y^2} \tag{4.1}$$

Since the input image is represented as a set of discrete pixels, we have to find a discrete convolution kernel that can approximate the second derivatives in the definition of the Laplacian.

Here we use two kernels with sizes 15x15 and 17x17 in combination with the changing thresholds and variances to check the effect of edges in the images. Using these kernels, the Laplacian can be calculated using standard convolution methods. Because these kernels are approximating a second derivative measurement on the image, they are very sensitive to noise. To counter this, the image is often Gaussian smoothed before the Laplacian filter is applied. This pre-processing step reduces the high frequency noise components prior to the differentiation step. Because the convolution operation is associative, we can convolve the Gaussian smoothing filter with the Laplacian filter, and then convolve this hybrid filter with the image to achieve the required result. Doing things this way has two advantages:

- Since both the Gaussian and the Laplacian kernels are usually much smaller than the image, this method usually requires far fewer arithmetic operations.
- The LoG kernel can be pre-calculated in advance so only one convolution needs to be performed at run-time on the image.

The 2D LoG function centered on zero and with Gaussian standard deviation has the form:

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\ 1 - \frac{x^2+y^2}{2\sigma^2}\ e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{4.2}$$

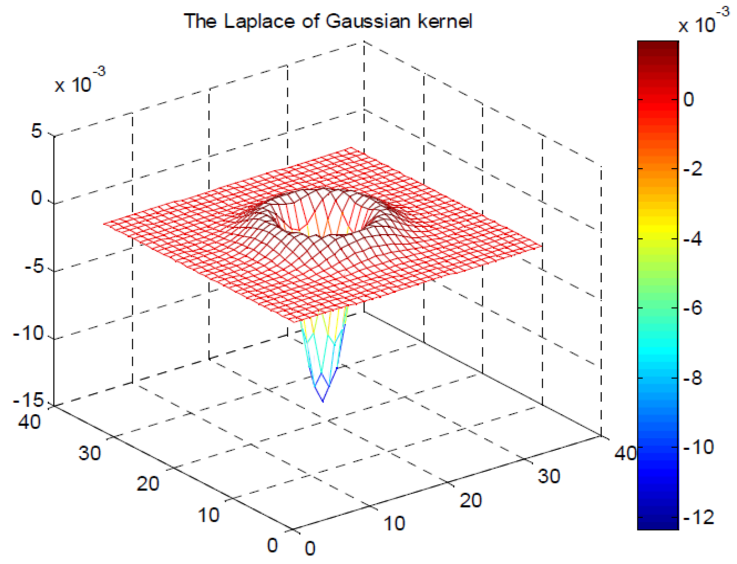


Figure 4.2TheLoG kernel.

Figure 4.2 shows the LoG kernel with σ equal to 2.5. If the image is pre-smoothed by aGaussian low-pass filter, then we have the LoG operation that is defined as

$$(K_{\nabla^2} ** (G_\sigma ** I)) = (K_{\nabla^2} ** G_\sigma) ** I = (\nabla^2 G_\sigma) ** I \tag{4.3}$$

Where

$$\nabla^2 G_\sigma(x,y) = \frac{1}{2\pi\sigma^4}\ \frac{x^2+y^2}{\sigma^2} - 2\ e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{4.4}$$

To find the places where the value of the Laplacian passes through zero points and also changes sign, we use the zero-crossing detector. We know that such points often occur at edges where the intensity of the image changes rapidly, but they also occur at places that are not easy to associate with edges. It is better to think of the zero-crossing detector as some sort of feature detector rather than as a specific edge detector. Zero crossings always lie on closed contours, and so the output from the zero-crossing detector is usually a binary image with single-pixel lines showing the positions of the zero-crossing points.

The initial input for the zero-crossing detector is an image which has been filtered using the Laplacian of Gaussian filter. The resulting zero crossings are strongly affected by the size of the

Gaussian used for the smoothing stage of this operator. As the smoothing is increased, then fewer and fewer zero-crossing contours will be found, and those that remain will correspond to features of larger and larger scale in the image. The summary of the crack detection algorithm is shown in Algorithm 1.

Table 1 The crack detection algorithm

```
Algorithm 1: Crack Detection

Step 1: Apply the LoG to the image.

Step 2: Apply the zero-crossing detector in the image

Step 3: Filter the zero-crossings to keep only those strong ones (large difference between the positive
maximum and the negative minimum).

Step 4: Suppress the weak zero-crossings most likely caused by noise.
```

.

4.2 Operation Interface for Crack Detection Based on the LoG algorithm

In this subsection, we discuss the operation interface for crack detection by using a GUI shown in Figure 4.3. The input images captured by the robot are collected and stored on the computer. To find out the crack in each image, we can select the image in Browse, then the Manual-Run button will start the crack detection algorithm and the crack results (white lines) are superimposed on the output image. In the Auto-Run mode. This mode allows us to detect the crack on all images automatically. After the crack on each image is detected, the crack locations in the image coordinate are mapped to the crack map.

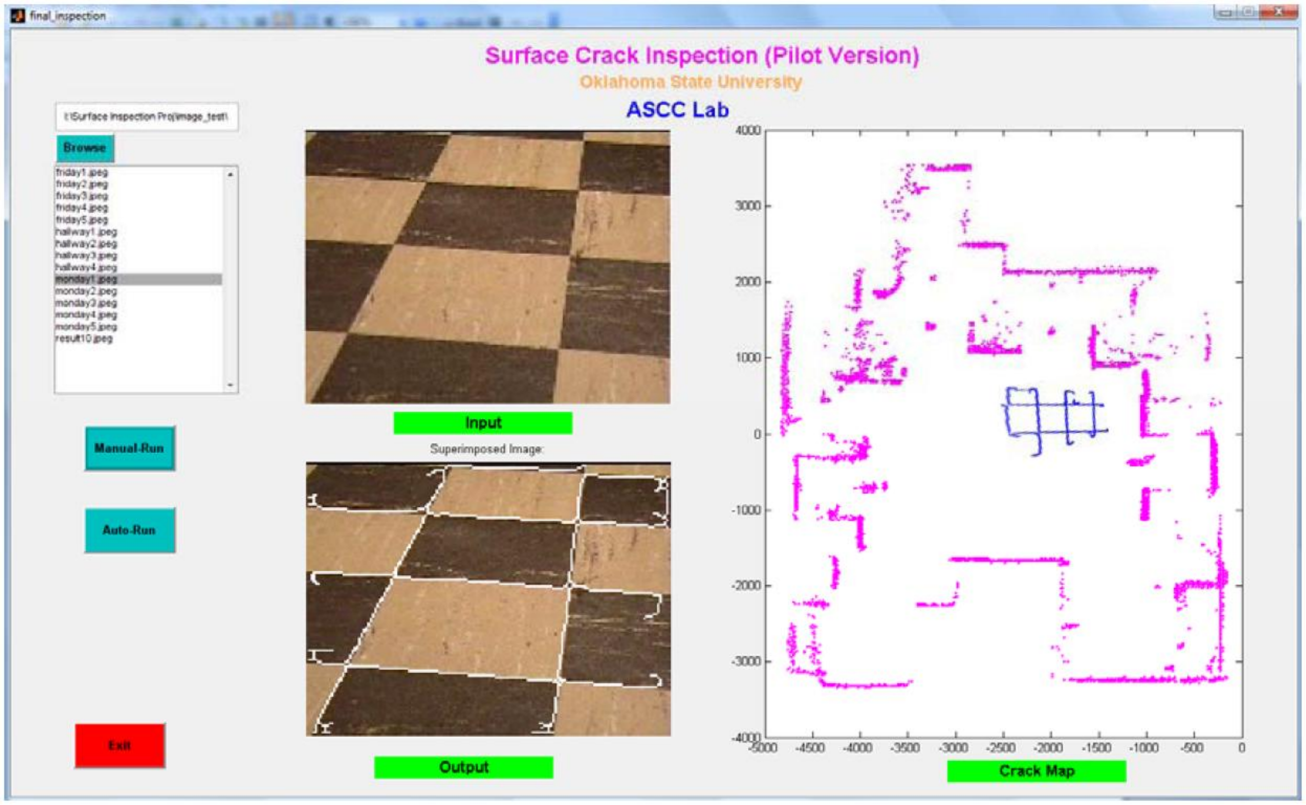


Figure 4.3 Graphical User Interface for Crack Detection and Mapping.

CHAPTER 5

COMPLETE COVERAGE PATH PLANNING

In this chapter, we present a new path planning algorithm that allows the robot to travel efficiently during the inspection. The path planning is important for the ROCIM system to save time and energy. We first formulate the problem then we propose two approaches to solve this problem.

5.1 Problem formulation

The purpose of complete coverage path planning (CCPP) is to ensure the camera field of view (FoV) covers the whole bridge deck surface. There are several issues that we need to address in this CCPP problem. First, a typical camera installation is mounted on top of the mobile robot as shown in Figure 5.1. Therefore, there is a blind spot which is caused by the offset distance between the robot body and the FoV. Second, to allow the mobile robot to detect small cracks, the camera should zoom in. Hence, it makes the size of FoVsmaller than the size of robot body. To illustrate these issues, we project the robot body and FoV to an XY-plane as shown in Figure 5.2.

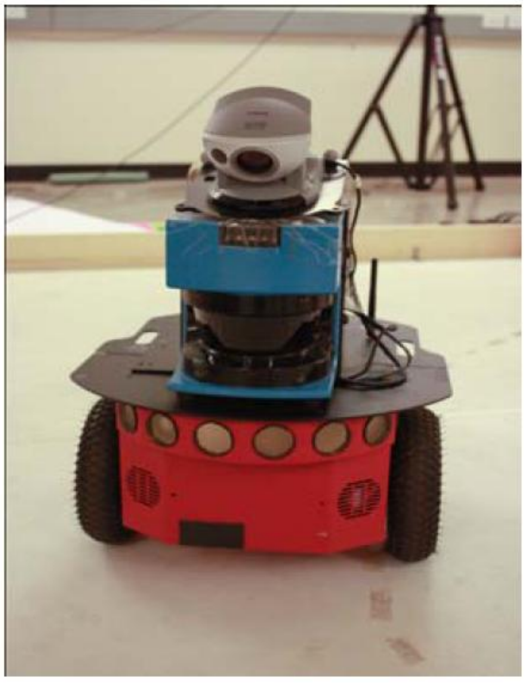


Figure 5.1 Pioneer3-DX with the camera on top of it.

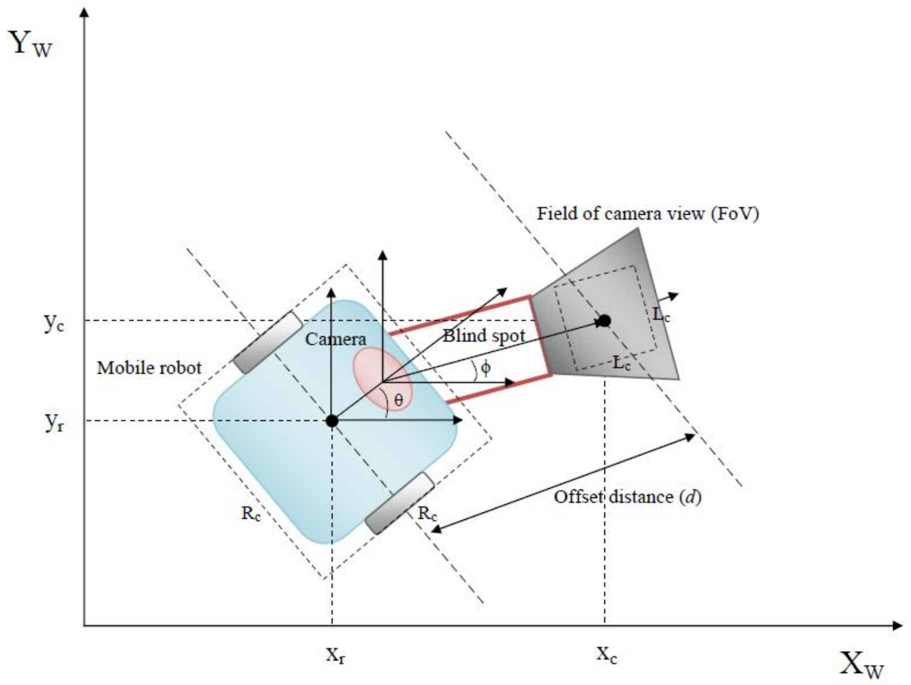


Figure 5.2 Projection of the mobile robot and camera field of view (FoV) to the XY plane.

To simplify the problem, the area of interest is partitioned into cells which have the same size as the FoV. We define several concepts todevelop the solution for this problem. These definitions are as follows:

Definition 1: Configuration of Inspection

Let $((x_r, y_r) \in \mathbf{R}^2)$ be the center location of the mobile robot; $\theta \in [0,2\pi]$ be the orientation of the mobile robot; and $\varphi \in [-\pi/2, \pi/2]$ be the pan angle of the camera. Then, a *configuration of inspection* (CoI^i) is defined as:

$\mathrm{CoI}^i = (x_r, y_r, \theta, \varphi)$ where *i*denotes a cell.

Definition 2: Motion Template

A motion template, MT(*i*, *j*), is the mobile robot and camera motion plan that transitions from $\mathrm{CoI}^i$to CoI^j:

$\mathrm{MT}(i, j) = \mathrm{CoI}^i \rightarrow \mathrm{CoI}^j$, $j \in \mathbf{N}(i)$, here **N** is the neighborhood of cell *i*as shown in Figure 5.4(a).

Definition 3: Inspection Path

An inspectio*n path*, IP, is a sequence of motion templates that ensure the camera FoVs*cover the whole area of interest. An inspection path*is defined as:

IP=[MT(i_1,i_2), MT(i_2,i_3),..., MT($i_{n-1}$$i_n$)] where ($i_1$,$i_2$,..$i_n$) is the camera path (a sequence of all the cells). An example of camera path is shown in Figure 5.4(b).

In Figure 5.3, we show the twelve CoIs where each CoI has a different robot orientation (θ) and camera pan angle (φ) where θ =(0^0,90^0, 180^0, 270^0) and φ =(-45^0,0^0,45^0). We use twelve CoIs because the robot heading (θ) is discretized into four values and the camera pan angle (φ) is discretized into three values. We encode the CoIs using alphabet from A to L: **S** = (A, B, C, D, E, F, G, H, I, J, K, L) as shown in Figure 5.3. The red square (darker square) is the mobile robot and the green square (lighter square) is the FoV. The relationship between the three definitions is shown in Figure 5.5. In a free space, all the twelve CoIs can be applied to cover a cell *i*as shown in Figure 5.6.

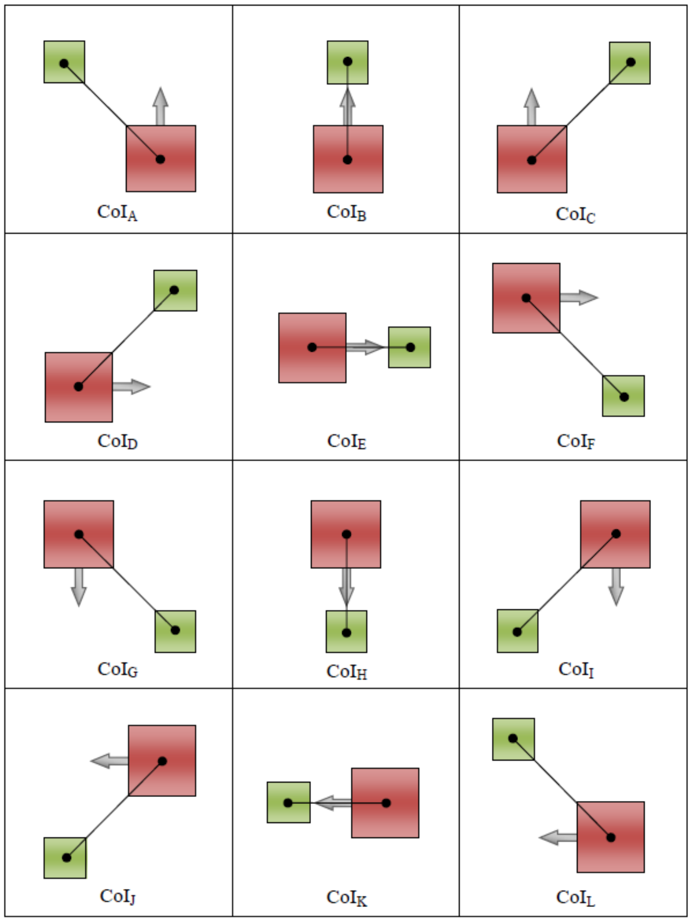


Figure 5.3 Twelve configuration of inspections (CoI).

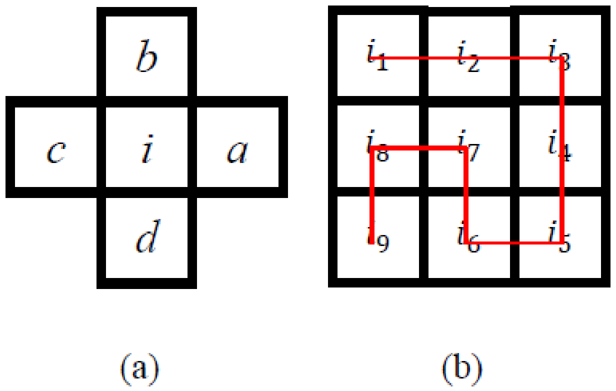


Figure 5.4 (a) Four directions defined by four neighbor cells. (b) An example of camera path to cover the whole area.

In Figure 5.4(a) we show that cell *i*has four neighbors because we assume the camera FoV moves only in four directions such as up, down, left and right. Therefore, there are 512 (12×12×4) different motion templates base on twelve CoIs and four camera path directions.

The problem of complete coverage path planning for the ROCIM system is how the mobile robot plans a path (inspection path) in order to have complete coverage of the cells in an efficient manner. This can be done by selecting the combination of CoIs which minimize a cost function. The cost function consists of three components: traveling distance, robot turns, and camera turns.

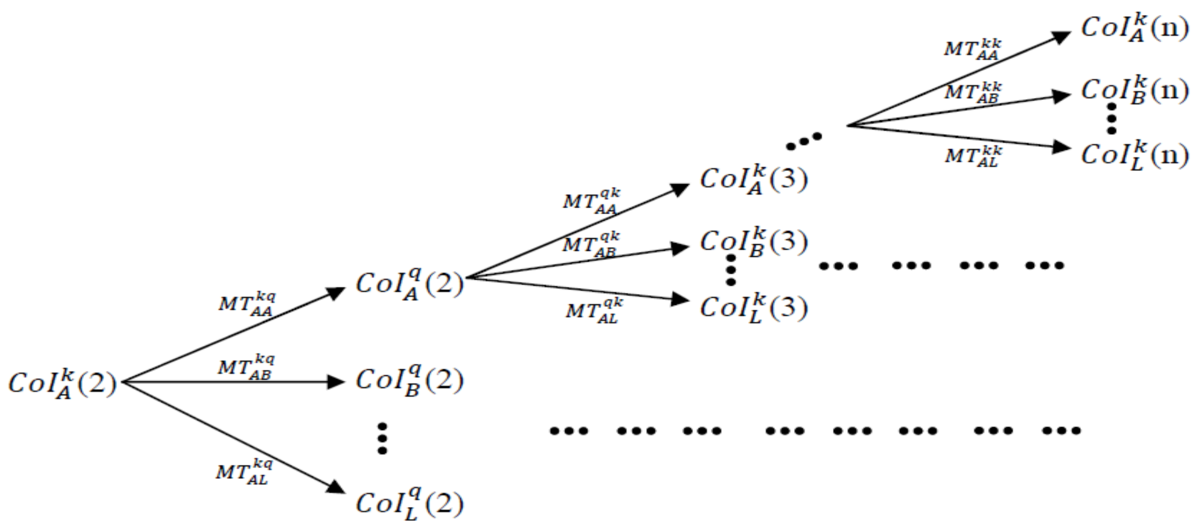


Figure 5.5 Relationship between configuration of inspection (CoI), motion templates (MT) and inspection path (IP).

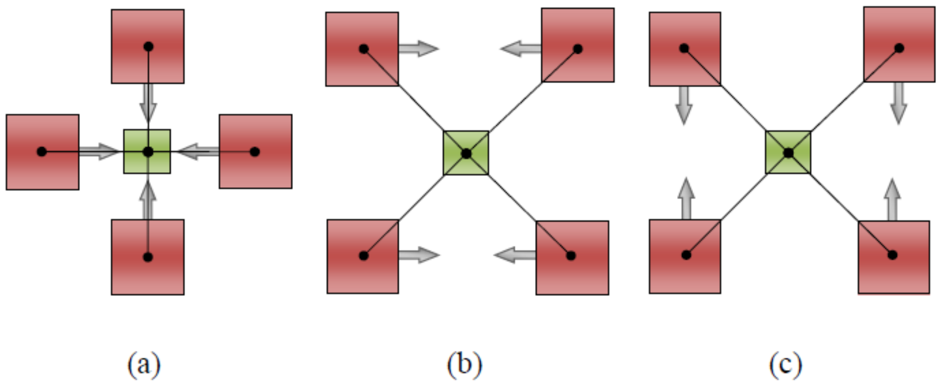


Figure 5.6 In free space, to inspect a cell, there are twelve CoIs that can achieve it.

In this paper, the RIP-GA (Robotic Inspection Planning based on Genetic Algorithm) algorithm is proposed to minimize the number of robot turns and traveling distance. The last component (camera turns) is not considered because the camera works in parallel to the mobile robot.

In summary, the CCPP problem can be stated as follows:

Given an area of interest, which is partitioned into cells, find an inspection path that covers all the cells while minimizing the numbers of robot turns and traveling distance.

In this project, we are interested in finding the sub-optimum solution because finding the optimum solution is a NP-hard problem and the computation increases exponentially with the size of environment.

5.2 Two approaches to complete coverage path planning

Many existing approaches cannot be directly applied to solve the CCPP problem because in these approaches, the robot body is used for complete coverage [17] [19] [20] [30] [31]. In this section, we discuss two different approaches to the CCPP problem. First, we present a heuristic approach which provides a full coverage of cells, however the traveling distance and the number of robot turns are not optimized. Second, we present a RIP-GA (Robotic path planning based on genetic algorithm) approach which gives a better solution than the heuristic approach.

5.2.1 Heuristic Approach

In this approach, the robot plans the inspection path based on the robot body movement. Typical motion planning such as boustrophedron motion [17], wall-following motion, and spiral motion can be used to plan the path. After the robot covers the entire area using the robot body, the coverage of FoV is considered. If the complete coverage is not achieved, the robot needs to inspect the uncovered cells by adding more movements. Pseudo-code of this approach is shown in Algorithm 1. We show some examples using this approach that can give a full coverage of the cells however the number of robot turns and traveling distance are not minimized. The examples are presented as follows:

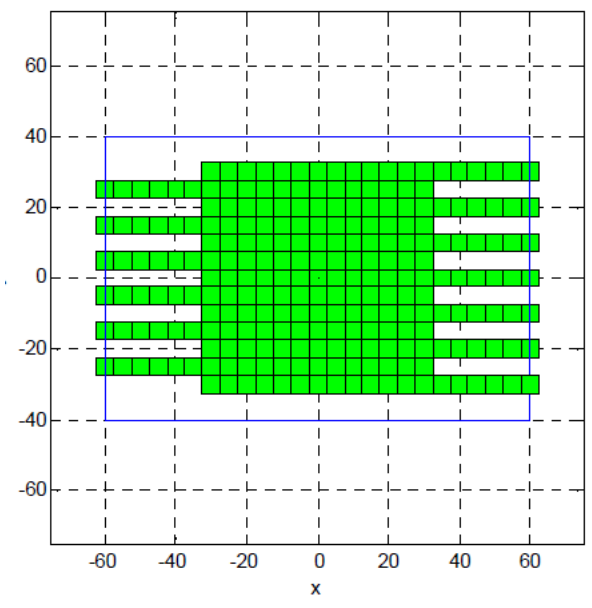
Example 1

We consider an empty rectangular map without any obstacle with size 80×120 as in Figure 5.7(a). The map is discretized into square cells with the same size as the FoV. The boustrophedron motion from top to bottom is applied to the mobile robot with camera facing forward, i.e., *CoI_B*. The mobile robot starts moving from (−50, 30) to (50, −30). These are missing areas especially along the edges as shown in Figure 5.7(a). The trajectory of this robot motion is shown in Figure 5.7(b). To cover the missing areas, the robot needs to re-plan the motion which results in a lot of overlapping path. Therefore, the solution is not very good.

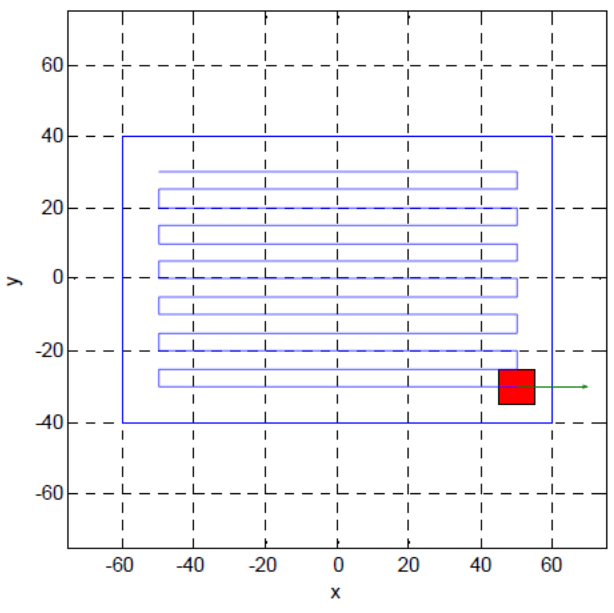
Table 2The heuristic approach.

```
Algorithm 2: Heuristic Approach

Step 1. Generate a mobile robot path to cover the area.

if there is any uncovered cell then
Step 2. Find out the uncovered cells and let the robot go back to cover these cells

else
Step 3. Finish.
```

In this example we change the pan-orientation of camera to 45 degree(CoIC) which allows therobot to inspect the cells along the edges. Then, we apply wall-following motion for the mobile robot to cover the entire area [26]. The coverage of this motion planning is shown in Figure 5.8(a) and the trajectory of the robot motion is shown in Figure 5.8(b). The result shows that full coverage is achieved. However there is unnecessary overlap in the coverage and robot paths. Therefore the traveling distance and the number of robot turns are not satisfying. As the result, this heuristic approach does not provide a good solution to the CCPP problem.

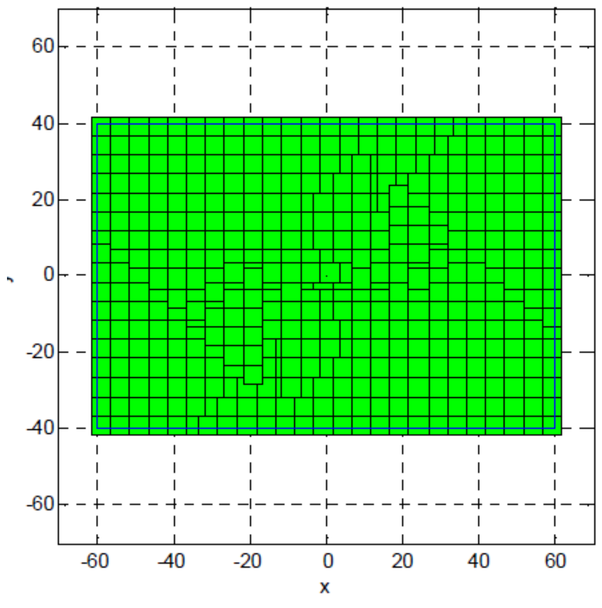


(a) Field of camera view

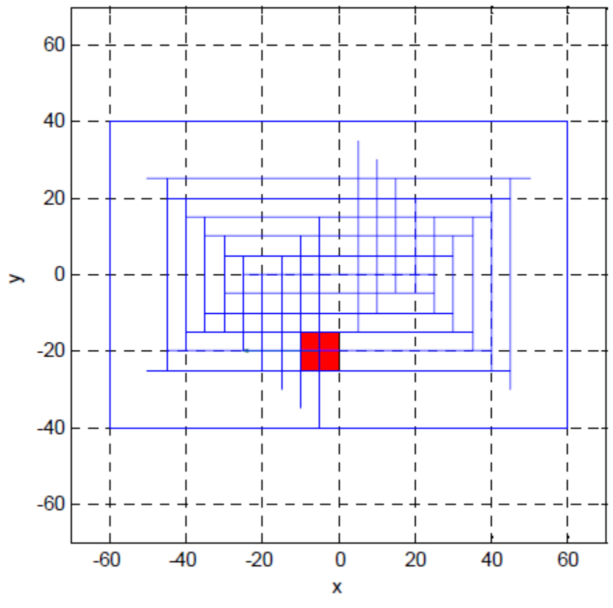


(b) Trajectory of robot path

Figure 5.7. Mobile robot trajectory with 0 degree camera orientation.



(a) Field of camera view



(b) Trajectory of robot path

Figure 5.8 Mobile robot trajectory with 45 degree camera orientation.

5.2.2 RIP Approach

The RIP approach finds a inspection path (IP) that minimizes the total traveling distance and the number of robot turns by selecting the motion templates (MT). The proposed algorithm extends the idea of the proposed RPP-CP (Robot path planning based on the camera path) [26]. The RPP-CP and RIP approaches assume that the camera path is planned in the first step in order to have complete coverage. Then, the robot and camera motion is optimized according to the camera path.

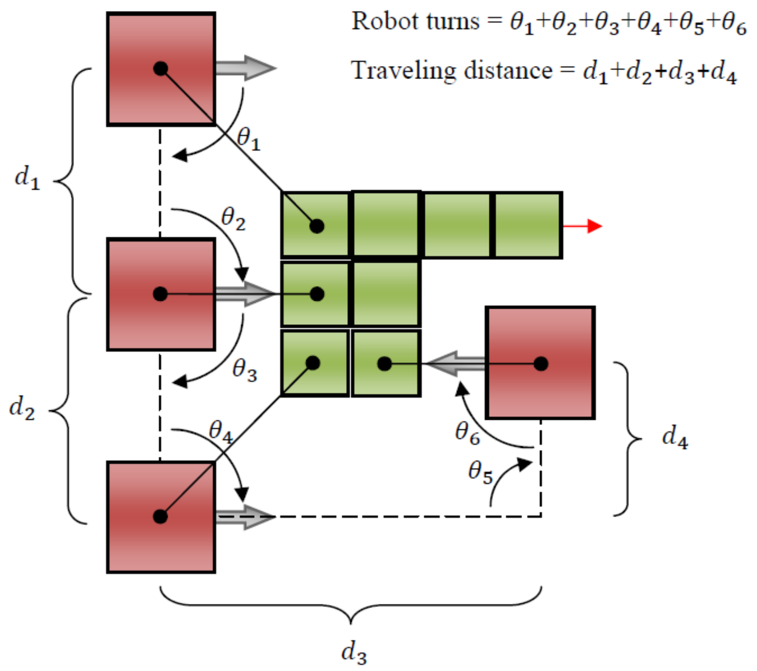
There are many ways to select the CoI, we utilize a genetic algorithm as the global search and greedy algorithm as the local search to do this. Here, we denote the RIP-GA as the robotics inspection path planning based on genetic algorithm and RIP-greedy as the robotics inspection path planning based on greedy algorithm. The genetic algorithm is an optimization method which mimics the natural evolution. The solution is generated using techniques such as mutation, selection and crossover [32]. We model some parameters to fit the genetic algorithm as follows:

- The gene is defined as the CoI
- The chromosome (C_r^I) is a sequence of CoIs, C_r^I=(CoI^{i1}, CoI^{i2}, CoI^{i3}, ..., CoI^{i_n}) and the length of C_r^I is the same as the number of cells (n).

In Figure 5.10(a) shows the illustration of the chromosome and it represents the solution to this CCPP problem. For each generation the genetic algorithm tries to find a better chromosome by minimizing the cost function.

$$F = \sum_{j=1}^{n}(\alpha \ q(i_j, i_{j+1}) \ + \beta \ u(i_j, i_{j+1}) \ ) \tag{5.1}$$

Here, the functions q, u basically are the traveling distance and robot turns, respectively. These functions are defined based on the existing path planning algorithm of the mobile robot. The variables α, βare the weight value for distance and time. In Figure 5.9 we show the calculation of the cost function from one CoI to another CoI.



Robot turns = $\theta_1+\theta_2+\theta_3+\theta_4+\theta_5+\theta_6$
Traveling distance = $d_1+d_2+d_3+d_4$
θ_1
θ_2
θ_3
θ_4
θ_5
θ_6
d_1
d_2
d_3
d_4

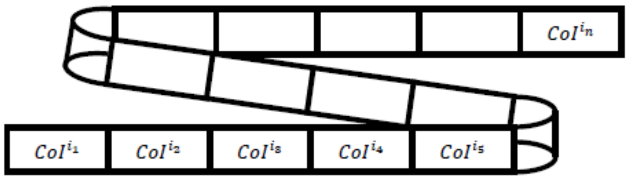
Figure 5.9 Calculation of the cost function.

The general idea of the RIP-GA algorithm is shown in Algorithm 2. In the step 4 of Algorithm 2, the multiple cross-over operation is applied. The nearest pair is mated toproduce the new offspring (ωI, ωI+1).

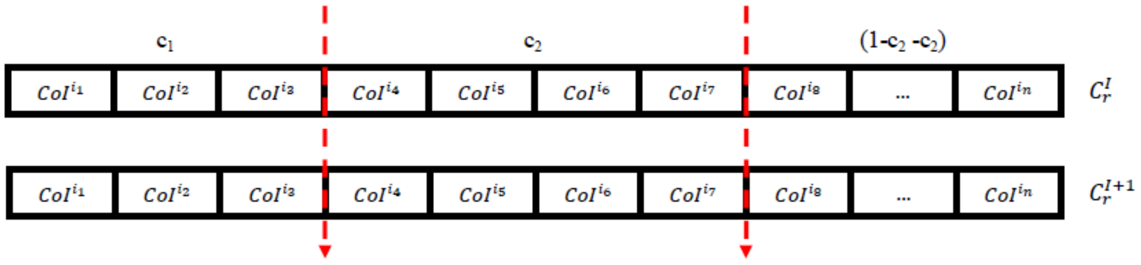
$$\omega^I = c_1 C_r^I + c_2 C_r^{I+1} + (1 - c_1 - c_2) C_r^I \tag{5.2}$$

$$\omega^{I+1} = c_1 C_r^{I+1} + c_2 C_r^I + (1 - c_1 - c_2) C_r^{I+1} \tag{5.3}$$

Equation 5.2 and 5.3 have to satisfy the following conditions as follows: c1, c2 ∈[0,1] and c1+c2 ≤ 1. Here, c1, c2 is a percentage of the CoI from one chromosome. For example, let c1 = 0.2, c2 = 0.3 then the chromosome ωItakes 20% genes from CrIto mate with 30% of the CrI+1. The illustration of the cross-over operation is shown in Figure 5.10(b). Then, the mutation operation is applied in order to avoid the genetic algorithm getting stuck in local minimum. After the crossover and mutation, the new offspring is evaluated with the old offspring. If the new offspring has less cost, it replaces the old offspring. Next, we remove the bottom chromosome and a new population is introduced. The algorithm continuously runs until the cost value of the top chromosome does not change for Q iterations.



(a)



(b)

Figure 5.10 (a) Chromosome. (b) Multiple chromosome crossover.

The greedy algorithm basically searches locally for the minimum cost function for the next CoI along the camera path. The greedy algorithm is robust to select the CoI but it always gets stuck in local minimums. The Pseudo-code of the greedy algorithm is shown in Algorithm 4.

Table 3 The RIP-GA algorithm.

```
Algorithm 3: RIP-GA
    Step 1. Generate a camera path CP(n) where CP = (x,y), CP ϵ Rn = 1: N,nϵ Z.

    foreach CP(n) do
    L Find a set of applicable $CoI_{(S)}$

    Step 2. Generate a population of M genetic strings randomly from CP:

    $P^I = (\omega^I{}_1, \omega^I{}_2, \omega^I{}_3 \ldots\ldots, \omega^I{}_N); I = 1, 2 \ldots, M.$

      while the genetic algorithm is not converged do
        1. Compute the cost function using Equation (5.1)
        2. Rank the genetic strings from top to bottom. $C^I (I = 1, 2, ,M)$
        3. Put the top chromosome to the next generation (Elitism).
        4. Mate the nearest pairs.
        5. Mutate the new offspring.
        6. Kill the bottom genetic strings ($B < M$) and keep the top $K$ parents.
        7. New parents = $K \cup R$. and $R$ is a random population to replace the bottom genetic strings.
        8. Evaluate the best chromosome
        if the genetic algorithm is converged then
          Go to Step 3.
        else
          Go to 1.

    Step 3. Generate the inspection path based on the CoI sequences.
```

Table 4 The RIP-Greedy algorithm.

```
Algorithm 4: RIP-Greedy
Step 1. Generate a camera path $CP(n)$ where $CP = (x,y)$, $CP \in \mathbf{R}$ $n = 1 : N, n \in \mathbf{Z}$.

    foreach CP(n) do
    Find a set of applicable $CoI_{(s)}$

Step 2. Find the minimum motion templates for each cell
Step 3. Generate the robot trajectory based on the Col (Inspection path).
```

CHAPTER 6

EXPERIMENT RESULTS

We conducted both simulations and experiments to validate the proposed ROCIM system and the associated algorithms. In the first experiment, we detect real cracks on the bridge deck. In the second experiment, we demonstrate the working of the ROCIM system in the indoor and outdoor environment. Finally, we evaluate the complete coverage path planning algorithm in both simulation and real robot implementation.

6.1 Real Crack Detection

In order to evaluate the performance of the ROCIM system, we first test our crack detection algorithm using real crack images which are collected from a bridge deck. The original image is shown in Figure 6.1(a). The Laplacian of Gaussian (LoG) algorithm is applied to the image, and the results are shown in Figure 6.1(b). The detected crack is superimposed on the original image as shown in Figure 6.1(c). Here, the result clearly shows the LoG algorithm successfully detects the crack in the image. The parameters of the LoG algorithm are: $\sigma = 2.5$ and threshold $T = 3$.

The limitation of the LoG algorithm is hard to find the optimal parameters such as σ and T. For example, using $\sigma \leq 1.5$ and $T = 2$, the result shows too much noise and the crack is hardly determined (See Figure 6.2). On other hand, if we set σ or T too large, the LoG removes most of the crack information. In Figure 6.2, we show the result of the LoG algorithm using various σ and T to detect the crack. In addition, our initial test indicates that the smallest crack that can be detected is around ± 0.4 mm.

6.2 Validation of the ROCIM system

In this section, we demonstrate the working of the ROCIM system. Three types of environments are evaluated in this experiment. The first is a simple indoor environment. We define the grids on the floor as the crack. The hardware setup for this environment is shown in Figure 6.3(a). The second is a complex indoor environment. We create artificial cracks by drawing curves on the ground as shown in Figure 6.3(b). The cones are used to set the inspection area. The third is an outdoor environment. We investigate the system to detect the crack on the concrete surface as shown in Figure 6.3(c).

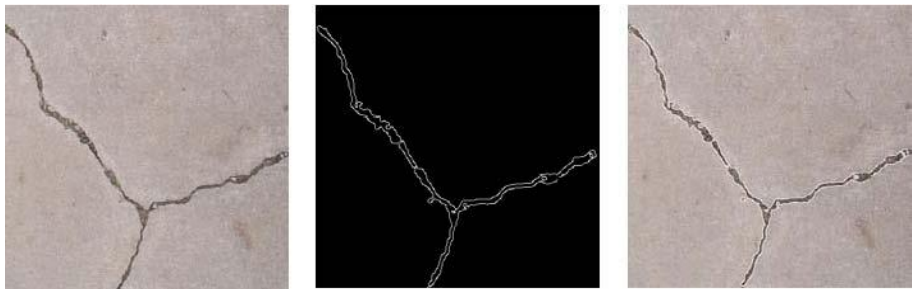


Figure 6.1: Crack detection result on a real bridge deck.

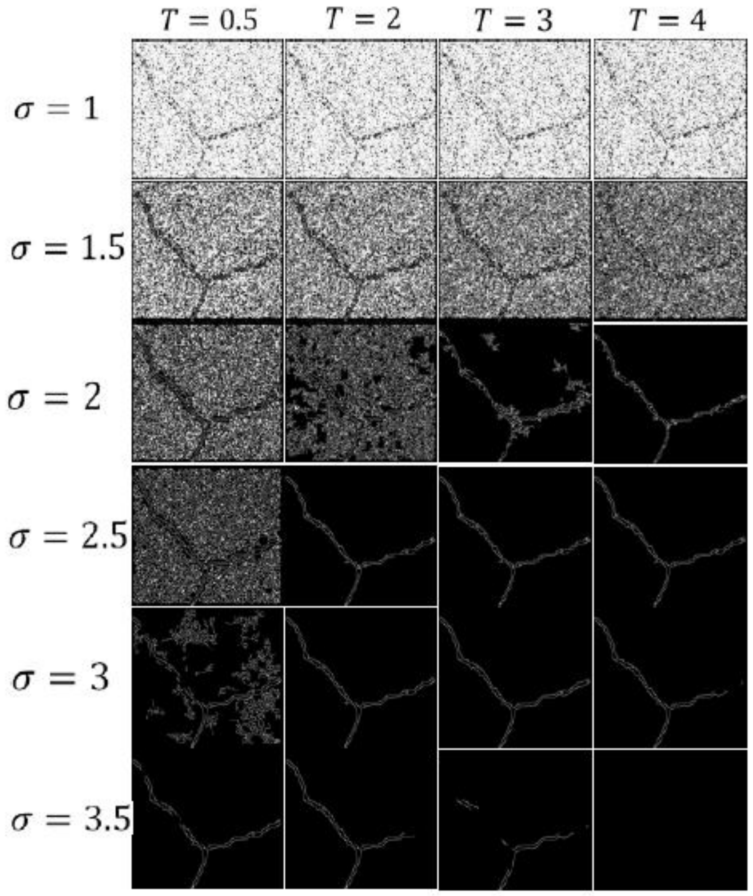
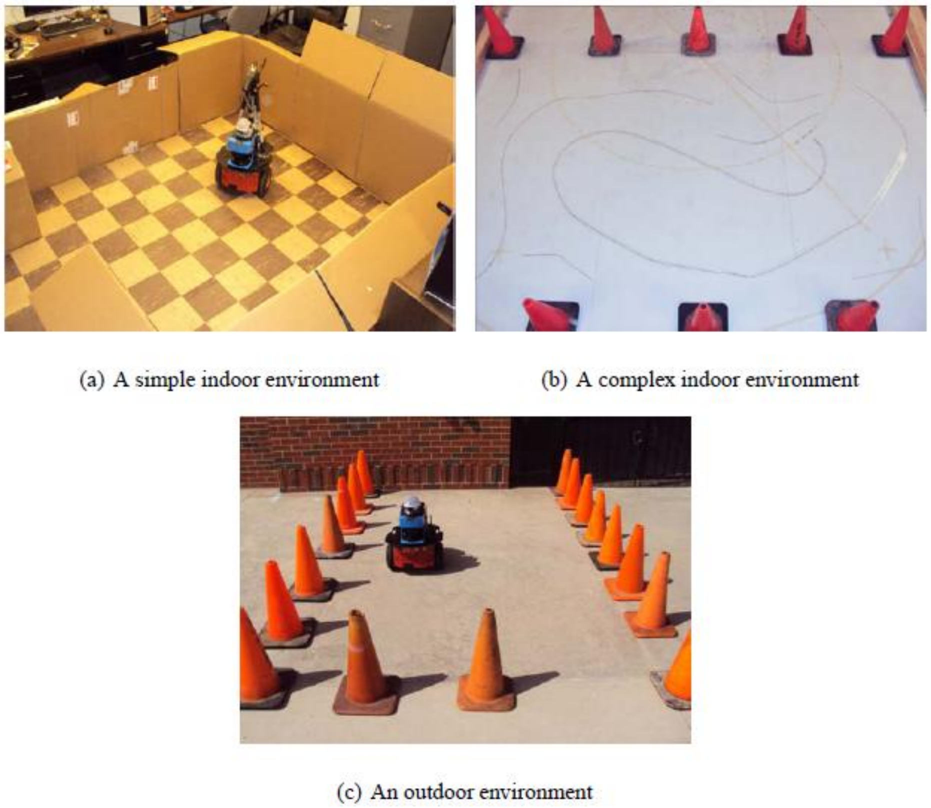


Figure 6.2 Comparison of the LoG parameters for crack detection.


(a) A simple indoor environment (b) A complex indoor environment

(c) An outdoor environment

Figure 6.3 Experiment setup to validate the ROCIM system.

6.2.1 A simple indoor environment

We use the Advanced Range Navigation Laser (ARNL) and the Mobile Eyes software from Mobile robot Inc. [1] to create a 2D map of the environment. Next, the 2D map is used to localize the mobile robot during the inspection using Monte Carlo Localization (MCL) algorithm. Before we run the inspection, we calibrate the camera using grids on the floor. The intrinsic and extrinsic parameters for the camera calibration with respect to the robot coordinate system are obtained as follows:

$$L = \begin{bmatrix} 220.3092 & 0 & 79.5000 \\ 0 & 220.3092 & 59.5000 \\ 0 & 0 & 1.0000 \end{bmatrix} M = \begin{bmatrix} 0.0370 & 0.9993 & 0.0102 & -414 \\ 0.7316 & -0.0202 & -0.6814 & -1157 \\ -0.6807 & 0.0327 & -0.7318 & 1924 \end{bmatrix}$$

There are six images collected from different places. Then, the LoG algorithm is applied to detect the crack in each image as shown in Figure 6.4. Next, we map all the crack locations to the global coordinate system. Figure 6.5 gives the overall crack map based on the six images. The blue and black lines represent the crack and the 2D global map, respectively. The result shows that the transformation of the crack locations from image coordinate system to the global coordinate system works well and the crack map is successfully created. However, there are some misalignments in the crack map.

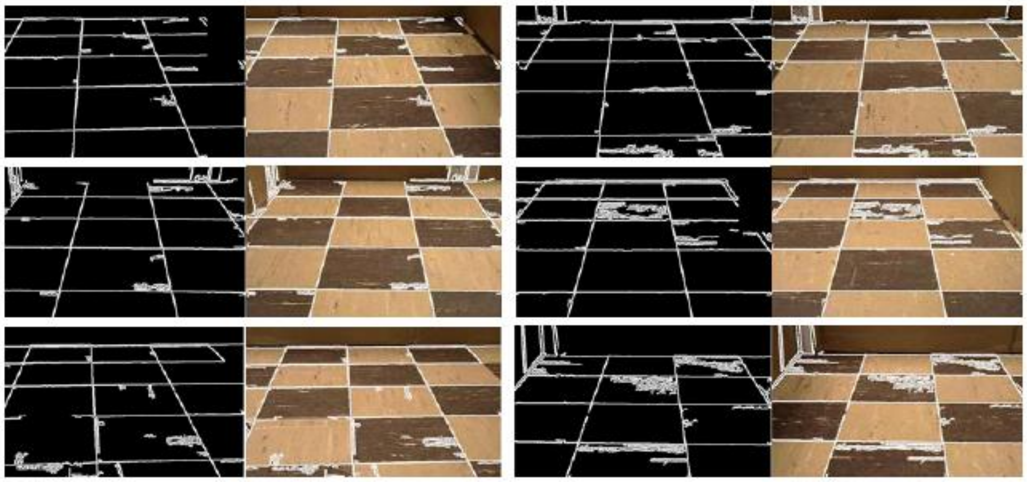


Figure 6.4 Crack detection results for a lab floor.

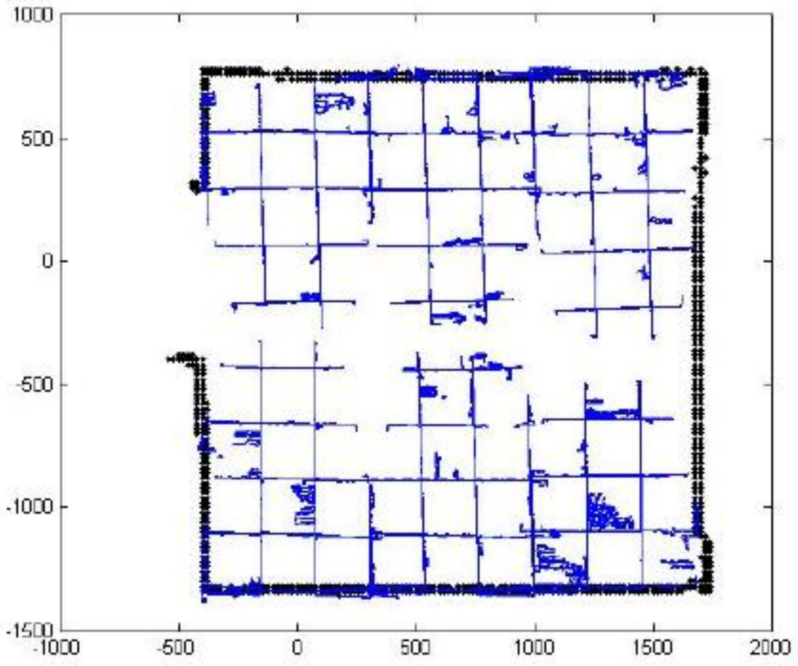


Figure 6.5 Crack map of the first experiment setup.

6.2.2 A complex indoor environment

In the second environment, the robot collects 18 images to cover the whole inspection area. The procedure to create the crack map is the same as the previous experiment. The result of the crack map is shown in Figure 6.6. The black line and blue lines represent the crack and the 2D map, respectively. Figure 6.7 shows some misalignment of the crack location in the crack map. This is mostly caused by the camera calibration and robot localization errors. Overall the accuracy of the crack detection can be up to 0.4mm with the proper camera.

6.2.3 An outdoor environment

In this experiment, we deploy the mobile robot to detect the crack on a concrete surface that is very similar to bridge decks. The experiment setup is shown in Figure 6.8. Here, the ROCIM system has to handle an open environment and the moving obstacle such as walking people or bypassing cars. The methodology to produce the crack map is the same as in indoor experiment. The problem with the outdoor experiment is mainly caused by the shadow. Figure 6.9 shows the crack map of this experiment. The misplacement of the crack map is a bit larger because of the robot localization is poorer. For all the results of crack map, we conclude that the proposed framework works well to create the crack map. We find that the robot can finish a typical bridge deck within about 30 minutes.

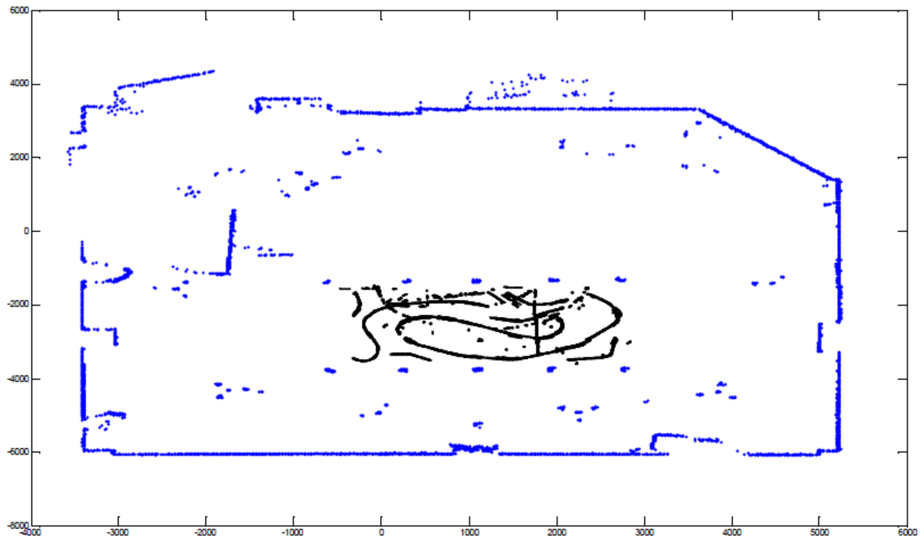


Figure 6.6 Crack map of the second experiment setup.

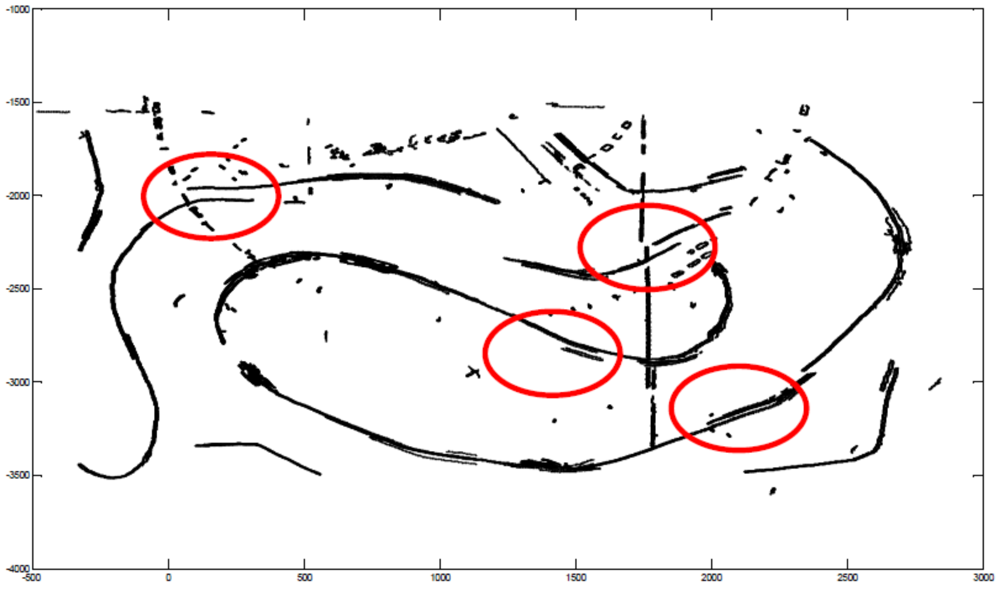


Figure 6.7Misalignment of the crack location.

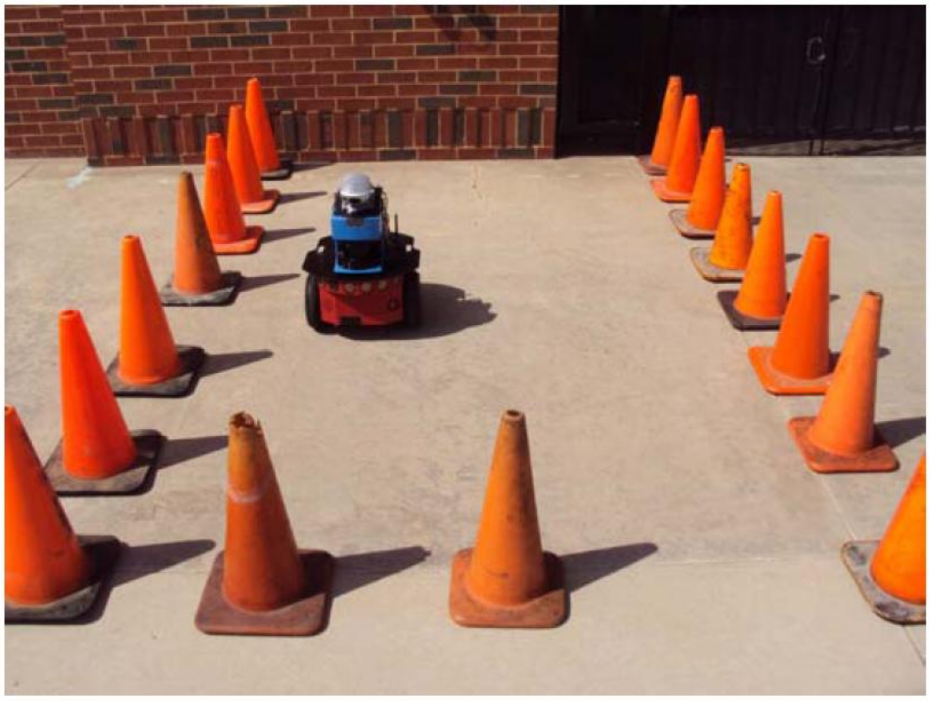


Figure 6.8 Experiment setup to evaluate the ROCIM system in the outdoor environment.

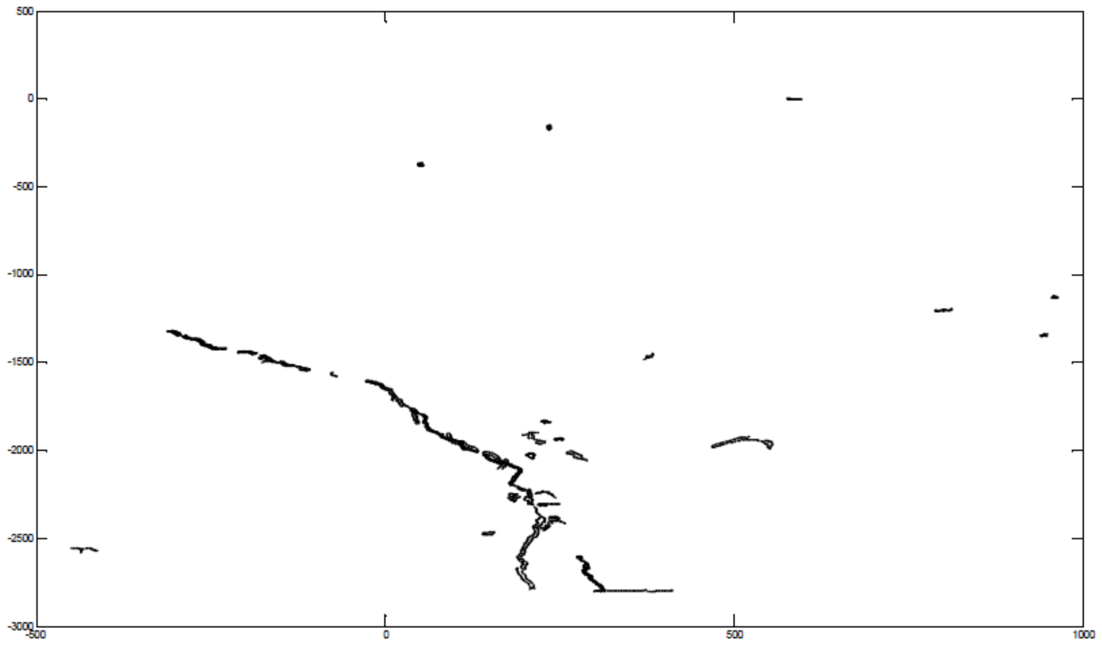


Figure 6.9 Crack map of the outdoor environment.

6.3 Complete coverage path planning

In this section we test our proposed Robotics Inspection Planning based on Genetic Algorithm (RIP-GA) algorithm in both simulation and real experiment. In the simulation, we implement our algorithm in obstacle and obstacle-free environments. In the real experiment, we implement our algorithm using the real mobile robot, Pioneer 3DX.

6.3.1 Obstacle-free environment

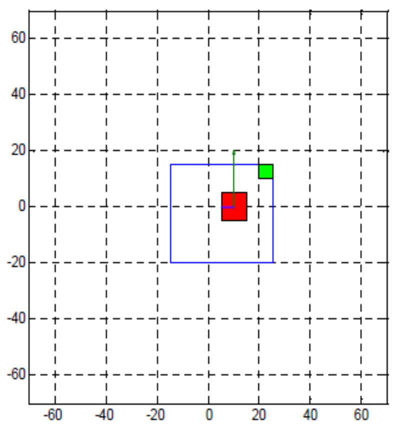
We define an environment without obstacles as in Figure 6.10(a). First, the camera path is planned using boustrophedron motion from top (left) to bottom. There are 55 cells in this environment and each cell have to be covered in order to have complete coverage of FoV. There are several parameters are needed to be initialized before we run the RIP-GA algorithm such as the number of population (M) = 500, the weight parameters $\alpha = 0.94$, $\beta = 1.1$, $\gamma = 0$, and $Q = 30$. The value of α and β are found by measuring the trade-off of traveling distance and robot turns into time in the simulation. We use $\gamma = 0$ since the camera operation works independently to the robot.

For each generation, we remove 50% of the bottom chromosome. We run the RIPGA algorithm to find the sub-optimum solution of the Inspection Path (IP). After certain iteration, the RIP-GA converged at 59.15. The performance of the RIP-GA algorithm for each generation is shown in

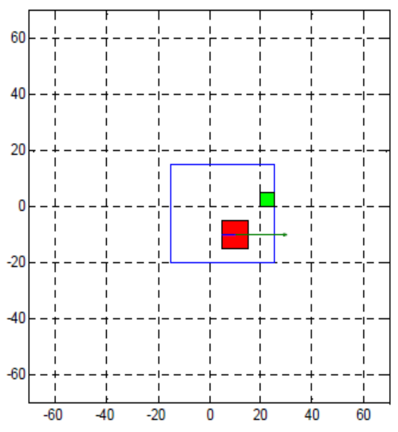
Figure 6.11. Moreover, RIP-GA finds more than one solution that have the same cost function and they are shown in Table 5. We compare the RIP-GA with RIP-greedy and the result is shown in Table 6. Here, the result of RIP-GA is slightly better than the RIP-greedy. We show some snapshots of the result by the RIP-GA approach and they are shown in Figure 6.10.

6.3.2 Obstacle environment

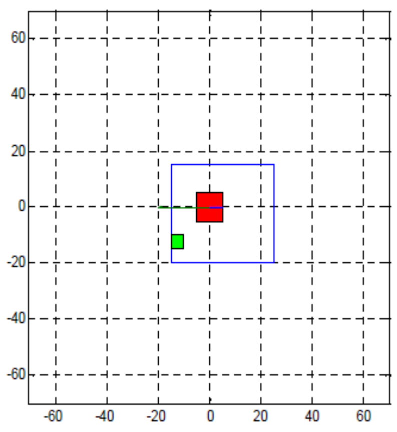
In this experiment, we evaluate the proposed algorithm in the obstacle environment. The obstacle environment is shown in Figure 6.12. Here, the obstacles are described by red squares. First, we decompose the environment into eight regions. Then, the camera path is planned using boustrophedron motion from bottom to top for each region. For each region, we indicate the starting and ending point are indicate by gray and black bullets, respectively. In this experiment, the initialization parameters are the same as in obstacle-free environment except the number of population (M). Since the obstacle environment is decomposed into several regions, we run separately the RIP-GA algorithm for each region. We use different value of M due to the difference size of the region. We use M = 1000 for region 1 and 3, and M = 500 for the rest regions.



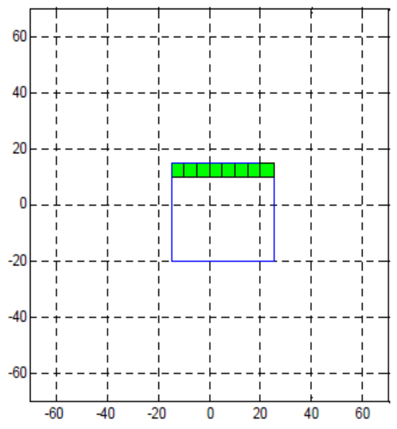
(a)



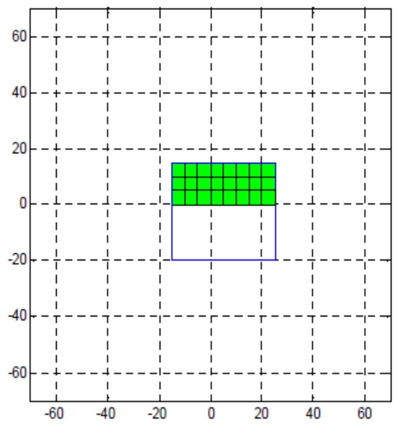
(b)



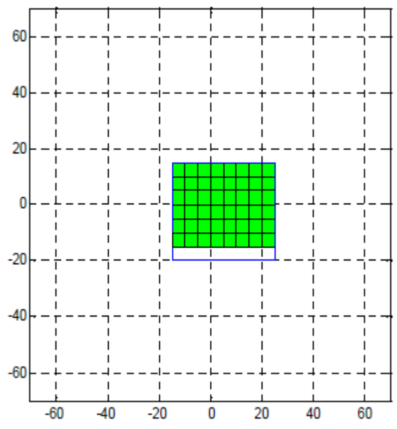
(c)



(d)



(e)



(f)

Figure 6.10 Snapshot of the simulation in obstacle-free environment. The Figure (a), (b) and (c) are the snapshot of the mobile robot position. The Figure (d), (e) and (f) are the snapshot of the covered camera view.

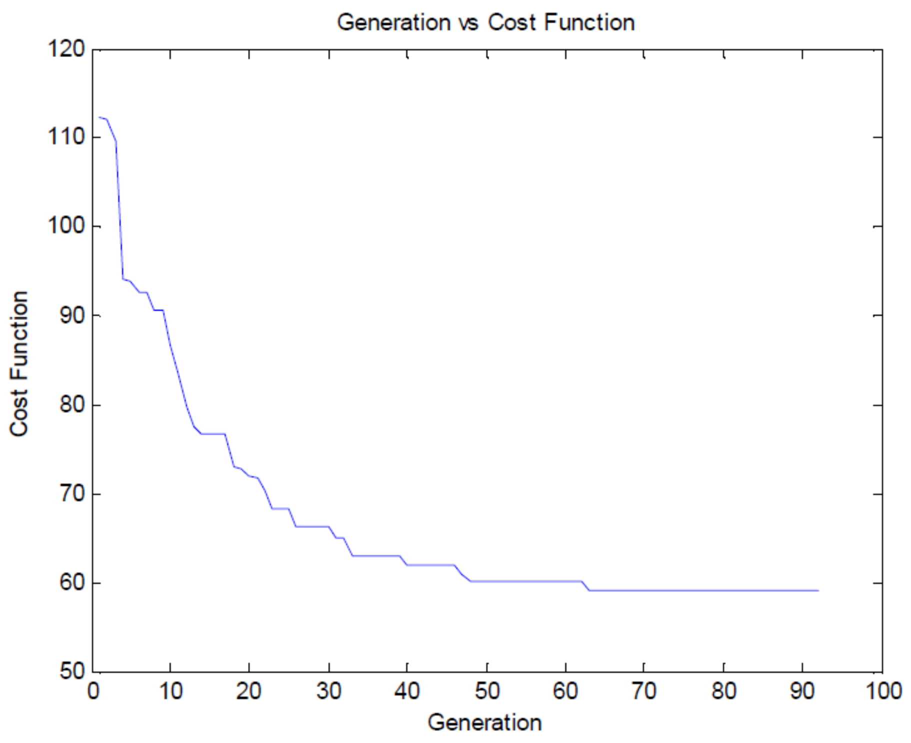


Figure 6.11 Performance of RIP-GA for obstacle-free environment.

Table 5Several solutions with the minimum cost function

CoIs
LLLLBDDCDDBLLLLALLLBDDDDFFHJJJJIJJJHFFFGGFFFHJJIIJJJHFFF
LLLLLBDDCDDBLLLAALLBDDDCFFHJJJJIJJJJHFFFFFFHJJJJIIJJJHFFF
LLLLBDDCDDBLLLLLLLLLBDDCFFHJJJJJJJJHFFFGFFFFHJJJJJJJHFFF

Table 6 Comparison of complete coverage path planning

	RIP-GA	RIP-Greedy
The number of robot turns	26	26
Traveling distance	325	395
cost value	59.15	65.73

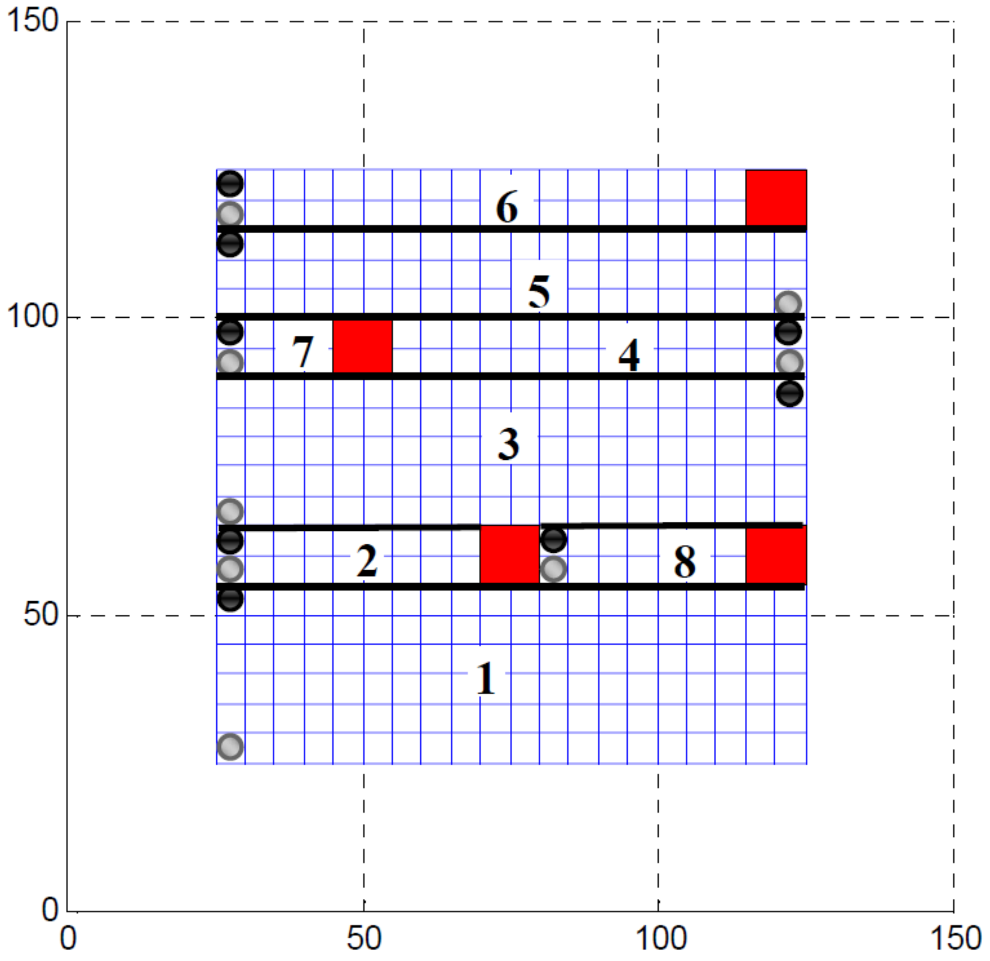


Figure 6.12. Map decomposition for obstacle environment.

The RIP-GA algorithm initials the search from a random sample for each cell. The random sample is basically CoIs. In the obstacle environment, several CoIs can't be used to cover the cell which locates in the corner or in the boundary. Therefore, we apply a CoI filtering to remove this inapplicable CoI. In Table 7, we show the result of CoI filtering for the first region. The subscript indicates the order of camera path and the letters indicate the applicable CoIs. For the first region, the RIP-GA generates the random population based on this table and the RIP-greedy calculates the least cost for each CoI according to this table too. We run the RIP-GA and RIP-greedy to find out the best combination of CoI to generate the IP. The comparison of both algorithm based on the cost function can be seen in Table 8. We see that the RIP-GA gives better solution than the RIP-greedy. The snapshots of the RIP-GA in simulation are provided in Figure 6.13.

Table 7Applicable CoIs in region 1.

IJ_{1}	IJK_{21}	IJK_{41}	$AIJKL_{61}$	$AIJKL_{81}$	$AIJKL_{101}$
HIJ_{2}	$HIJK_{22}$	$HIJK_{42}$	$ABHIJKL_{62}$	$ABHIJKL_{82}$	$ABHIJKL_{102}$
HIJ_{3}	$HIJK_{23}$	$HIJK_{43}$	$ABHIJKL_{63}$	$ABHIJKL_{83}$	$ABHIJKL_{103}$
$FGHIJ_{4}$	$EFGHIJK_{24}$	$EFGHIJK_{44}$	$ABCDEFGHIJKL_{64}$	$ABCDEFGHIJKL_{84}$	$ABCDEFGHIJKL_{104}$
$FGHIJ_{5}$	$EFGHIJK_{25}$	$EFGHIJK_{45}$	$ABCDEFGHIJKL_{65}$	$ABCDEFGHIJKL_{85}$	$ABCDEFGHIJKL_{105}$
$FGHIJ_{6}$	$EFGHIJK_{26}$	$EFGHIJK_{46}$	$ABCDEFGHIJKL_{66}$	$ABCDEFGHIJKL_{86}$	$ABCDEFGHIJKL_{106}$
$FGHIJ_{7}$	$EFGHIJK_{27}$	$EFGHIJK_{47}$	$ABCDEFGHKL_{67}$	$ABCDEFGHKL_{87}$	$ABCDEFGHL_{107}$
$FGHIJ_{8}$	$EFGHIJK_{28}$	$EFGHIJK_{48}$	$ABCDEFGHKL_{68}$	$ABCDEFGHKL_{88}$	$ABCDEFGHL_{108}$
$FGHIJ_{9}$	$EFGHIJK_{29}$	$EFGHIJK_{49}$	$ABCDEFGKL_{69}$	$ABCDEFGKL_{89}$	$ABCDEFGL_{109}$
$FGHIJ_{10}$	$EFGHIJK_{30}$	$EFGHIJK_{50}$	$ABCDEFGIJKL_{70}$	$ABCDEFGIJKL_{90}$	$ABCDEFGIJKL_{110}$
$FGHIJ_{11}$	$EFGHIJK_{31}$	$EFGHIJK_{51}$	$ABCDEFGIJKL_{71}$	$ABCDEFGIJKL_{91}$	$ABCDEFGIJKL_{111}$
$FGHIJ_{12}$	$EFGHIJK_{32}$	$EFGHIJK_{52}$	$ABCDEIJKL_{72}$	$ABCDEIJKL_{92}$	$ABCDIJKL_{112}$
$FGHIJ_{13}$	$EFGHIJK_{33}$	$EFGHIJK_{53}$	$ABCDEHIJKL_{73}$	$ABCDEHIJKL_{93}$	$ABCDHIJKL_{113}$
$FGHIJ_{14}$	$EFGHIJK_{34}$	$EFGHIJK_{54}$	$ABCDEHIJKL_{74}$	$ABCDEHIJKL_{94}$	$ABCDHIJKL_{114}$
$FGHIJ_{15}$	$EFGHIJK_{35}$	$EFGHIJK_{55}$	$ABCDEFGHIJKL_{75}$	$ABCDEFGHIJKL_{95}$	$ABCDEFGHIJKL_{115}$
$FGHIJ_{16}$	$EFGHIJK_{36}$	$EFGHIJK_{56}$	$ABCDEFGHKL_{76}$	$ABCDEFGHKL_{96}$	$ABCDEFGHL_{116}$
$FGHIJ_{17}$	$EFGHIJK_{37}$	$EFGHIJK_{57}$	$ABCDEFGHKL_{77}$	$ABCDEFGHKL_{97}$	$ABCDEFGHL_{117}$
FGH_{18}	$EFGH_{38}$	$EFGH_{58}$	$BCDEFG_{78}$	$BCDEFG_{98}$	$BCDEFG_{118}$
FGH_{19}	$EFGH_{39}$	$EFGH_{59}$	$BCDEFG_{79}$	$BCDEFG_{99}$	$BCDEFG_{119}$
FG_{20}	EFG_{40}	EFG_{60}	$CDEFG_{80}$	$CDEFG_{100}$	$CDEFG_{120}$

Table 8Comparison of RIP-GA and RIP-greedy

Region	*RIP-GA*	*RIP-Greedy*
1	*84.3600*	*88.1300*
2	*10.190*	*11.7600*
3	*70.0900*	*76.0600*
4	*14.8900*	*16.4600*
5	*47.4250*	*53.0000*
6	*19.1200*	*23.8300*
7	*5.4900*	*7.0600*
8	*8.3100*	*9.8800*

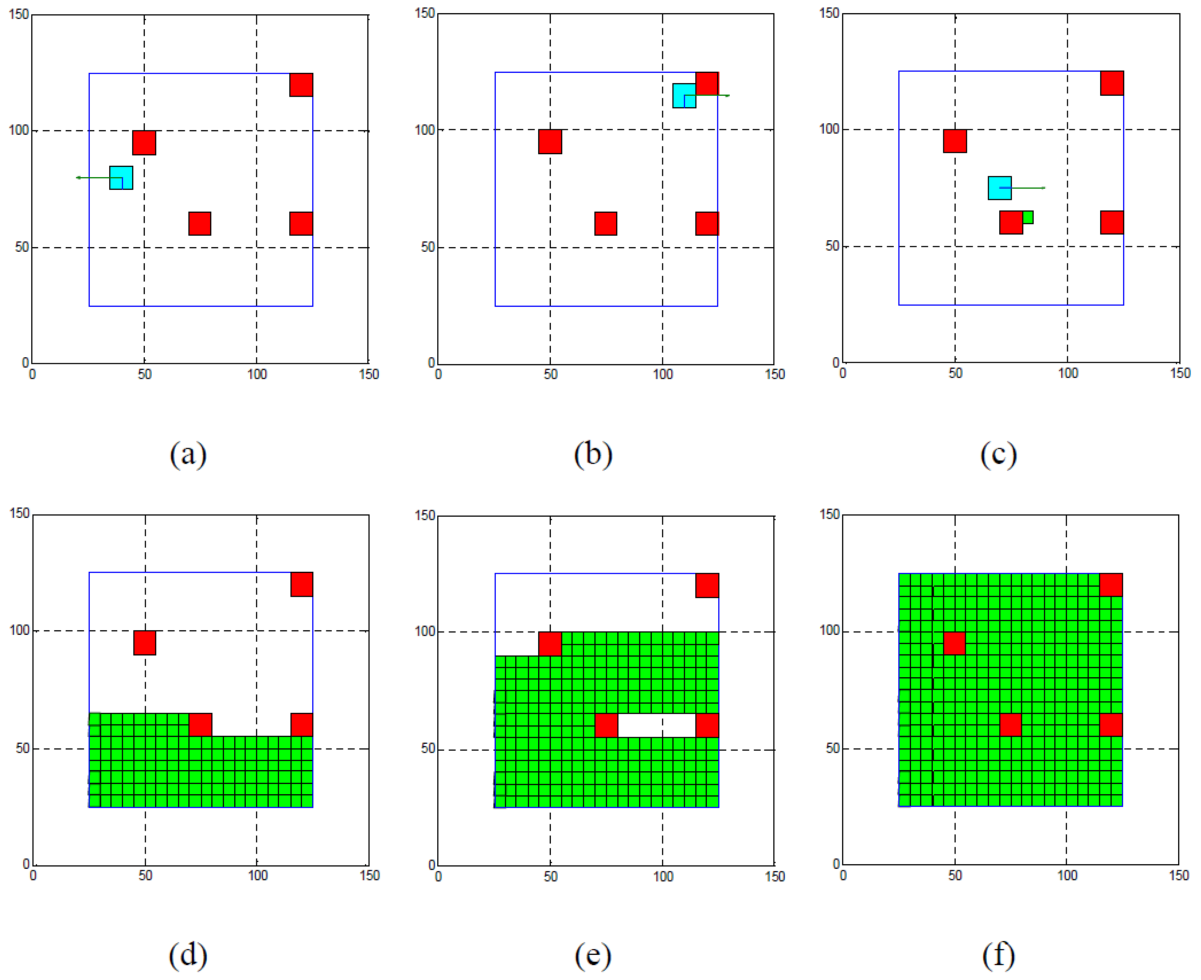


Figure 6.13 Snapshot of the simulation for obstacle environment. The Figure (a), (b) and (c) are the snapshot of covered camera view. The Figure (d), (e) and (f) are the snapshot of robot position.

6.3.3 Real robot implementation

In the simulation, the RIP-GA algorithm gives a satisfactory result to solve the CCPP problem. In this section we discuss how to implement the proposed algorithm in the real robot, Pioneer3-DX. After the sub-optimum solution for the Inspection Path (IP) is found, the robot need to travel from one location to another location where a particular robot and camera pose are maintained for each location.

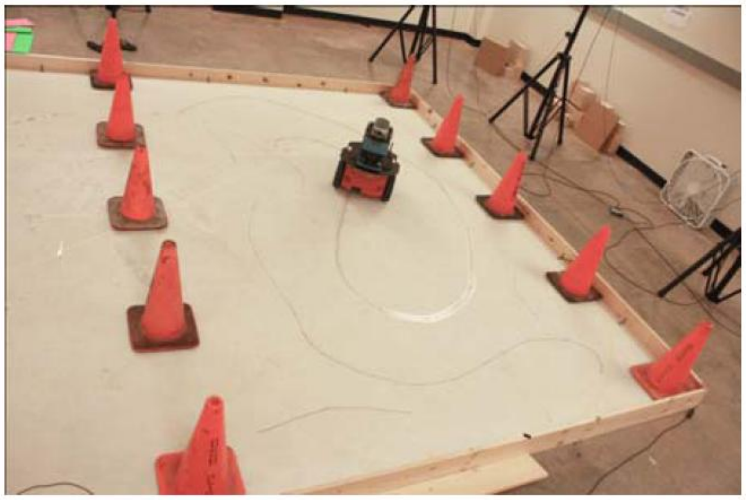


Figure 6.14 Laboratory environment to test real robot implementation.

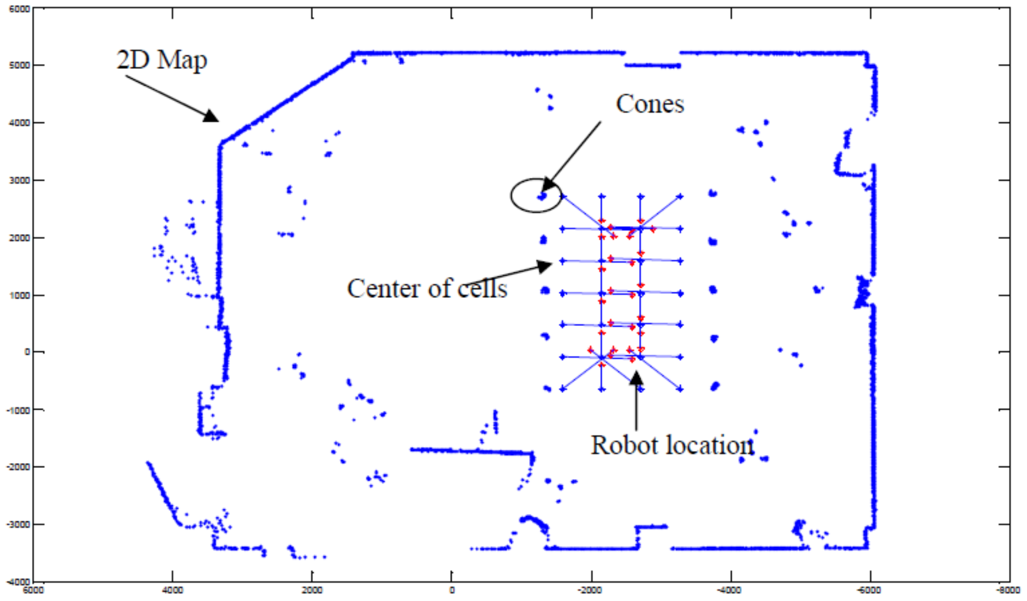


Figure 6.15 Robot locations for each cell in the 2D map.

The set of CoI of this experiment is the same as in the simulation. The laboratory environment for this experiment is shown in Figure 6.14. We use cones to make a boundary for the inspection area. First, the mobile robot creates a 2D map of this environment using ARNL (Advanced Range Navigation Laser) library [1]. Second, we define the size and the number of cells in the 2D map. For this experiment, the inspection area is decomposed into 28 cells. We run the RIP-GA algorithm to find the optimum IP. The Figure 6.15 presents the 2D map with the output of the RIP-GA algorithm.

The output of the RIP-GA algorithm is the center of camera path location (blue star) and the IP location (red star). Third, the IP is given to the mobile robot to follow this path using ARNL [1]. For each location in the IP, the robot changes the camera pose and collects the image as illustrated in Figure 6.16. The number indicates the sequences of CoI which is the sub-optimum solution from RIP-GA algorithm. Here, the complete coverage can be guaranteed because the robot localizationis very good for indoor environment.

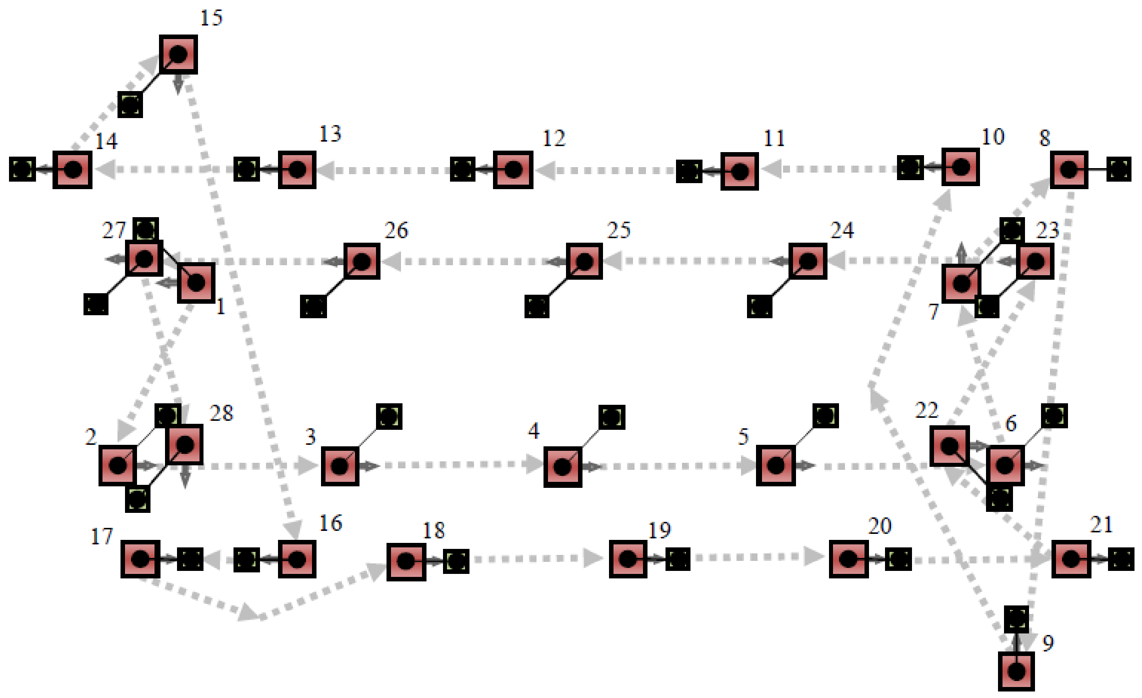


Figure 6.16 Illustration of the inspection path with the CoIs.

CHAPTER 7

CONCLUSIONS AND RECOMMENDATIONS

7.1 Summary

In this project, we developed a Robotic Crack Inspection and Mapping (ROCIM) system to replace human operators in bridge deck crack inspection. This ROCIM system creates a crack map which is useful for measuring, classifying, and analysis of crack growth. This project is conducted by a team of interdisciplinary researchers with expertise in the area of mobile robotics, remote sensing and civil engineering. The project consists of three major tasks: 1) the development of the ROCIMplatform; 2) the development of robot autonomous navigation (localization and path planning) for inspection; 3) the development of crack detection and crack map generation. We validated our proposed system through both indoor and outdoor experiments. The proposed research can significantly improve the accuracy of bridge deck inspection and reduces the inspection time. Timely awareness of bridge structural health can make our transportation system much safer while man power will be significantly saved.

7.2 Research findings

Through this project, we have the following findings:

- The proposed ROCIM system meets the project goals in terms of the accuracy and time of inspection. The ROCIM system can achieve an accuracy of 0.016 inch and the time required for a typical bridge is about 30 minutes, given the right camera sensors are used.
- We have successfully developed the ROCIM hardware system and the software system. The overall cost of the ROCIM system is about 20K$. Such a ROCIM system can be quickly duplicated since most of the parts are commercial of the shelf.
- We developed algorithms for camera calibration which facilitates the mapping of the crack location in the images to the crack locations in the global frames. We used Matlab toolbox to develop the calibration algorithms.
- We developed the robot localization and mapping algorithms to create the global map of the

bridge. The MCL algorithm works fine for robot localization. However the accuracy of the localization affects the accuracy of the crack mapping. There need to be sufficient landmarks for a robot to accurately localize itself. Therefore it would be better to put more road cones during the actual application.

- There are many algorithms for crack detection. We find that the Laplacian of Gaussian algorithm works finein most scenarios. However, to handle shadows caused by sun light and other weather conditions, we may need explore other algorithms for crack detection. We tested the ROCIM system in both indoor and outdoor environments, and the result shows that the ROCIM system works well.
- For the complete coverage path planning, we introduce the Robotics Inspection Path based on the Genetic Algorithm (RIP-GA) to ensure the mobile robot collects all the images in an efficient way. The RIP-GA algorithm proves to be better than the RIP-greedy algorithm in terms of robot turns, traveling distance and inspection time. We implemented the proposed algorithm on the real mobile robot, Pioneer3-DX. The implementation result shows the reliability and robustness of the proposed algorithm.

7.3 Recommendations

In the future work, we can further improve the ROCIM system. Some of the improvements are as follows:

- On-line complete coverage path planning: The proposed RIP-GA algorithm is developed for static environment and is an off-line path planning algorithm. To deploy the ROCIM system in the passing traffic, we need to extend the RIP-GA algorithm for dynamic environments.
- Multi-robot cooperation: It would be desirable to develop a multi-robot cooperation algorithm that allows multiple ROCIM systems to perform the inspection in order to save more time for the crack inspection.
- The user interface should be further improved and simplified for an average user. The procedure involved in the setting up, preparing and running the ROCIM system should be simplified as well. Otherwise it will cause a significant amount of burden on the operator side which may make them reluctant to use this system.

- The system should be further tested in different weather conditions, as well as road traffic scenarios. The vibration of the bridge may be an issue that may affect the crack detection through cameras.
- The robot platform used in the ROCIM system can be further improved, or a different mobile platform can be used to improve the speed and reliability of the ROCIM system.

7.4 Implementation and Technology Transfer

This project has been implemented and tested in both indoor and outdoor environments. The tests show that we can achieve the goals we originally set. Currently the ROCIM system can achieve an accuracy of 0.016 inch and the time required for a typical bridge is about 30 minutes. However, to deploy this ROCIM system in real world applications, there are still several issues that should be addressed. First, the user interface should be further improved and simplified for an average user. The procedure involved in the setting up, preparing and running the ROCIM system should be simplified as well. Otherwise it will cause a significant amount of burden on the operator side which may make them reluctant to use this system. Second, the system should be further tested in different weather conditions, as well as road traffic scenarios. The vibration of the bridge may be an issue that may affect the crack detection through cameras. Third, the robot platform used in the ROCIM system can be further improved, or a different mobile platform can be used to improve the speed and reliability of the ROCIM system.

The result of this project has been published in conference papers and a news article by the technical news website called *Visions Systems Design*. A link to the news article is at:

http://www.vision-systems.com/articles/print/volume-16/issue-9a/departments/technology-trends/vision-guided-robotics-intelligent-robot-performs-bridge-integrity-analysis.html

REFERENCES

[1] Mobile robotics, Inc., http://www.mobilerobots.com/, Dec. 2012.

[2] J. Heikkil, "Geometric camera calibration using circular control points," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 1066–1077, 2000.

[3] Wikipedia, "http://en.wikipedia.org/wiki/i-35w mississippi river bridge," 2007.

[4] C. R. Farrar, H. Sohn, and S. W. Doebling, "Structural health monitoring at los alamos national laboratory," *The 13th International Congress and Exhibition on Condition Monitoring and Diagnostic Engineering Management (COMADEM 2000)*, 2000.

[5] H. Sohn, C. R. Farrar, M. L. Fugate, , and J. J. Czarnecki., "Structural health monitoring of welded connections," *The First International Conference on Stell and Composite Structures*, 2001.

[6] E. Sazonov, K. Janoyan, and R. Jha., "Wireless intelligent sensor network for autonomous structural health monitoring," *Proceedings of the SPIE -The International Society for Optical Engineering*, vol. 5384(1), pp. 305–314, 2004.

[7] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, andD. Estrin, "A wireless sensor network for structural monitoring," *In SenSys'04*, 2004.

[8] D. R. Huston, "Adaptive sensors and sensor networks for structural health monitoring," *Proceedings of SPIE -The International Society for Optical Engineering*, vol. 4512, pp. 203–211, 2001.

[9] S. N. Yu, J. H. Jang, and C. S. Han, "Auto inspection system using a mobile robot for detecting concrete cracks in a tunnel," *Automation in Construction*, vol. 16, pp. 255– 261, 2007.

[10] S. K. Sinha and P. W. Fieguth, "Automated detection of cracks in buried concrete pipe images," *Automation in Construction*, vol. 15, pp. 58–72, 2006.

[11] P. C. Tung, Y. R. Hwang, and M. C. Wu, “The development of a mobile manipulator imaging system for bridge crack inspection,” *Automation in Construction*, vol. 11, pp. 717–729, 2002.

[12] J. H. Lee, J. M. Lee, J. W. Park, and Y. S. Moon, “Efficient algorithms for automatic detection of cracks on a concrete bridge,” *The 23rd International Technical Conference on Circuits/Systems, Computers and Communications*, pp. 1213–1216, 2008.

[13] J. K. Oh, G. Jang, S. Oh, J. H. Lee, B. J. Yi, Y. S. Moon, J. S. Lee, and Y. Choi, “Bridge inspection robot system with machine vision,” *Automation in Construction*, vol. 18, pp. 929–941, 2009.

[14] H. G. Sohn, Y. M. Lim, K. H. Yun, and G. H. Kim, “Monitoring crack changes in concrete structures,” *Computer-Aided Civil and Infrastructure Engineering*, vol. 20, pp. 52–61, 2005.

[15] A. Habib and D. Kelley, “Single photo resection using the modified hough transformation,” *Photogrammetric Engineering and Remote Sensing*, vol. 67, no. 8, pp. 909–914, 2001.

[16] A. Ito, Y. Aoki, and S. Hashimoto, “Accurate extration and measurement of fine cracks from concrete block surface image,” *Proceedings IECON2002*, pp. 77–82, 2002.

[17] H. Choset, “Coverage of known spaces: the boustrophedon cellular decomposition,” *Autonomous Robots*, vol. 9, pp. 247–253, 2000.

[18] H. Choset and P. Pignon, “Coverage path planning: The boustrophedon decomposition,” *Proceedings of the International Conference on Field and Service Robotics*, 1997.

[19] M. Kapanoglu, M. Ozkan, A. Yazici, and O. Parlaktuna, “Pattern-based genetic algorithm approach to coverage path planning for mobile robots,” vol. 5544, pp. 33–42, 2009. 10.1007/978-3-642-01970-8 4.

[20] P. A. Jimenez, B. Shirinzadeh, A. Nicholson, and G. Alici, “Optimal area covering using genetic

algorithms," *Proceedings of the 2007 IEEE /ASME International Conference on Advanced Intelligent Mechatronics*, pp. 1–5, 2007.

[21] S. Yang and C. Luo, "A neural network approach to complete coverage path planning," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 34, pp. 718 – 724, February 2004.

[22] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *J. Physiol. Lond*, vol. 117, pp. 500–544, 1952.

[23] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust montecarlo localization for mobile robots," Artificial Intelligence, Volume 128, Issues 1–2, Pages 99–1412001.

[24] S. Thrun, "Simultaneous localization and mapping," *Robotics and Cognitive Approaches to Spatial Mapping*, vol. 38, pp. 113–41, 2008.

[25] D. A. Forsyth and J. Ponce, "Computer vision: A modern approach", *Prentice Hall*, Upper Saddle River, New Jersey, 2003.

[26] R. S. Lim, H. M. La, Z. Shan, and W. Sheng, "Developing a crack inspection robot for bridge maintenance," *Proceedings of IEEE International Conference on Robotics and Automation*, 2011.

[27] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 1999.

[28] Natural Point Inc., http://www.naturalpoint.com/optitrack/, Dec, 2012.

[29] D. A. Forsyth and J. Ponce, "Computer vision: A modern approach," *Prentice Hall*, Upper Saddles, New Jersey, 2003.

[30] Y. Mao, L. Dou, J. Chen, H. Fang, H. Zhang, and H. Cao, “Combined complete coverage path planning for autonomous mobile robot in indoor environment,” in *Asian Control Conference, 2009. ASCC 2009. 7th,* pp. 1468 –1473, Aug. 2009.

[31] T. Oksanen and A. Visala, “Coverage path planning algorithms for agricultural field machines,” *Journal of Field Robotics,* vol. 26, pp. 651–668, 2009.

[32] Wikipedia, “http://en.wikipedia.org/wiki/genetic algorithm,”

[33] I. ActivMedia Robotics, “Aria reference manual 1.1.10,” 2002.

[34] S. A. I. Abuhadrous and F. Nashashibi, “Digitization and 3d modeling of urban environments and roads using vehicle-borne laser scanner system,” *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems,* 2004.

[35] S. I. Schwartz, “Catch me, i’m falling,” *New York Times*, 2007.

[36] V. Giurgiutiu, C. A. Rogers, Y. J. Chao, M. A. Sutton, , and X. Deng, “Adaptive health monitoring concepts for spot-welded andweld-bonded structural joints,” *Proceedings of the ASME Aerospace Division,* vol. 54, pp. 99–104, 1997.

[37] G. W. Reich and K. C. Park, “A use of substructural transmission zeros for monitoring,” *AIAA Journal,* vol. 38, pp. 1040–1046, 2000.

[38] Z. L. Cao, Y. Y. Huang, and E. L. Hall, “Region filling operations with random obstacle avoidance for mobile robots,” *J. Robotic Syst.*, pp. 87–102, 1988.

[39] E. M. Arkin and H. Refael, “Approximation algorithms for the geometric covering salesman problem,” *Discrete Applied Mathematics,* vol. 55, pp. 197–218, 1994.

[40] H. Choset, “Coverage for robotics.,” *Annals of Mathematics and Artificial Intelligence,* vol. 31, pp. 113–126, 2001.

[41] C. Luo and S. Yang, “A bioinspired neural network for real-time concurrent map building and complete coverage robot navigation in unknown environments,” *IEEE Transactions Neural Networks*, vol. 19, pp. 1279–1298, 2008.

[42] A. Zelinsky, R. A. Jarvis, J. C. Byrne, and S. Yuta, “Planning paths of complete coverage of an unstructured environment by a mobile robot,” *Proceedings of International Conference on Advanced Robotics*, pp. 533–538, 1993.

[43] S. O. Joon, H. C. Yoon, B. P. Jin, and Y. F. Zheng, “Complete coverage navigation of cleaning robots using triangular-cell-based map,” *IEEE Trans. Ind. Electron.*, Vol. 51, Issue 3, Pages 718 – 726, 2004.

[44]Complete Camera Calibration Toolbox for Matlab®,

“http://www.vision.caltech.edu/bouguetj/calib_doc/index.html,”

[45] H. B. D. M. T. Hagan and M. Beale, “Neural network design,” *Boston: PWS Publishing Co.*, 1996.

[46] A. Hoover, V. Kouznetsova, and M. Goldbaum, “Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response,” *Medical Imaging, IEEE Transactions on*, vol. 19, pp. 203 –210, march 2000.