

# A Robotic Crack Inspection and Mapping System for Bridge Deck Maintenance

Ronny Salim Lim, Hung Manh La, *Member, IEEE*, and Weihua Sheng, *Senior Member, IEEE*

**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 marks the location of cracks manually. However, the accuracy of the inspection result is low due to the subjective nature of human judgement. We propose a crack inspection system that uses a camera-equipped mobile robot to collect images on the bridge deck. In this method, the Laplacian of Gaussian (LoG) algorithm is used to detect cracks and a global crack map is obtained through camera calibration and robot localization. To ensure that the robot collects all the images on the bridge deck, a path planning algorithm based on the genetic algorithm is developed. The path planning algorithm finds a solution which minimizes the number of turns and the traveling distance. We validate our proposed system through both simulations and experiments.

**Note to Practitioners**—This work addresses crack detection and mapping on a bridge deck using a robotic system. Several challenges including coordinate transformation, robot localization and complete coverage path planning for the proposed robot system are tackled. This paper focuses mainly on the overall framework for such a robotic inspection system, therefore some of the techniques for handling shadows, paints, patches on bridges are not addressed. In real-world applications, these issues should be carefully incorporated into the design of the image processing algorithm. Also, there may be vibration caused by the passing traffic, which should be dealt with as well. The positioning of the ROCIM system is critical to crack mapping, hence more accurate robot localization techniques fusing various sensors such as differential GPS, Inertial Measurement Unit (IMU), etc. should be developed. It is also worth noting that the depth and severity of the cracks can be measured by employing advanced nondestructive evaluation (NDE) sensors, such as impact echo and ultrasonic surface wave.

**Index Terms**—Crack inspection, mobile robot, path planning.

Manuscript received January 11, 2013; revised August 16, 2013; accepted October 17, 2013. Date of publication January 09, 2014; date of current version April 03, 2014. This paper was recommended for publication by Associate Editor V. Isler and Editor D. Tilbury upon evaluation of the reviewers' comments. This work was supported in part by the National Science Foundation (NSF) under Grant CISE/CNS MRI0923238 and the Oklahoma Transportation Center under Grant OTCREOS10.1-43.

R. S. Lim and H. M. La are with the Center for Advanced Infrastructure and Transportation, Rutgers University, Piscataway, NJ 08854 USA (e-mail: ronny.lim@rutgers.edu; hung.la11@rutgers.edu).

W. Sheng is with the School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK 74078 USA (e-mail: weihua.sheng@okstate.edu).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TASE.2013.2294687

## I. INTRODUCTION

### A. 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-35 W Mississippi River Bridge that carried Interstate 35 W across the Mississippi River in Minneapolis left 13 dead and more than 100 injured [1], not to mention its big impact on the traffic and businesses 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. The inspectors usually walk though the bridges and measure the crack sizes and locations. This manual approach has several disadvantages. First, it is prone to human errors. Second, it has limited accuracy due to the limited visual capability of human inspectors. Third, it cannot guarantee the full coverage of the whole bridge deck. Additionally, conducting visual crack inspection of a bridge deck is a dangerous job with passing traffic. Therefore, it is highly desirable to develop a robotic crack inspection and mapping (ROCIM) system for accurate assessment of cracks on bridge decks. This system can outperform human inspectors in several ways. First, the ROCIM system can achieve high accuracy of crack detection if equipped with a high-resolution camera. Second, the ROCIM system can localize itself precisely, which facilitates accurate crack localization. Finally, by using a robot, the ROCIM system can greatly reduce the safety risk of human inspectors.

### B. 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, while a camera is used to collect images of the bridge deck surface. The collected images are then processed using

image processing techniques to detect the cracks. We store the crack locations in this 2D map, therefore obtaining a crack map, which can be used to measure, classify and monitor cracks periodically. In order to implement the ROCIM system, there are several challenging problems that should be addressed.

- 1) Crack detection: To detect cracks on the bridge deck using computer vision, we need to develop an effective edge detection algorithm to distinguish cracks and noncracks.
- 2) Coordinate transformation: To create a crack map, the crack location has to be pinpointed in the global coordinate system, or the coordinate system of the 2D map of the bridge deck. Since the cracks are detected in the image coordinate system, we have to map the crack locations from the image coordinate system to the global coordinate system.
- 3) Path planning of the mobile robot: The path planning should ensure the mobile robot can inspect the whole bridge deck surface in an efficient way. Unlike the typical complete coverage path planning of a vacuum cleaning robot, which uses the robot body as the coverage, the mobile robot in the ROCIM system uses the camera field of view as the coverage, which poses some difficulties in path planning.

### C. Literature Review

In this section, we review the existing works of robotic and coverage path planning.

Recent years have witnessed growing research interests in structural health monitoring for bridges, buildings and other civil infrastructures [2]–[9]. Research in structure inspection using robotic devices has resulted in several prototypes. Yu *et al.* [10] presented an automated inspection system using a mobile robot that detects concrete cracks in a tunnel. An illuminator is used to help distinguish cracks from non-cracks. Sinha *et al.* [11] developed a statistical filter for crack detection in pipes. In their system, crack features from the buried pipe images are first extracted, and then the cracks among the segment candidates are detected by a cleaning and linking procedure. Tung *et al.* [12] proposed a mobile manipulator system equipped with a binocular CCD camera for bridge crack inspection. Lee *et al.* [13] and Oh *et al.* [14] proposed a bridge inspection system which consists of a specially designed car, a robotic mechanism and a control system for automatic crack detection. Sohn *et al.* [15] developed a system that monitors crack change in concrete structures. Their system focuses on quantifying the crack change from multi-temporal images during the monitoring period. Ito *et al.* [16] demonstrated an automated measurement system for concrete block inspection by means of fine crack extraction. Their proposed system uses a high-resolution camera to capture images, and the cracks are automatically extracted using an integrated image processing technique. Most of these studies classify, measure, and detect cracks. However, none of these works studies the global mapping of cracks and the optimization problem in inspection path planning.

Coverage path planning for mobile robots has been investigated by many researchers. Choset *et al.* [17], [18] studied complete coverage path planning using boustrophedon motion. Their proposed algorithm allows for obstacle avoidance, and has

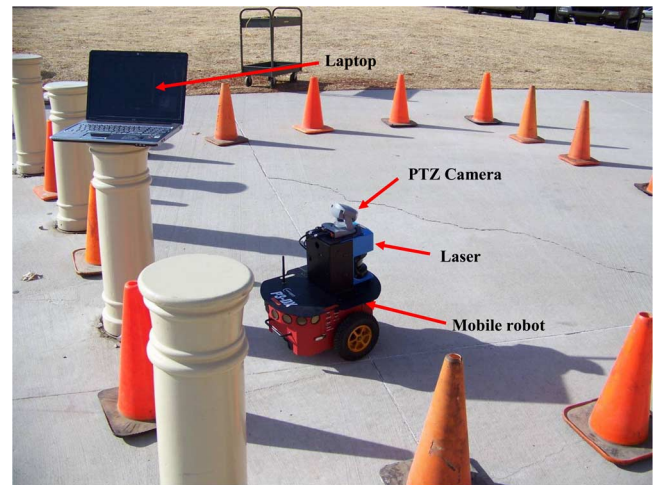


Fig. 1. The ROCIM system in action.

efficient coverage paths. A neural network approach to complete coverage path planning for a vacuum cleaning robot is developed by Simon *et al.* [19]. Their algorithm is capable of planning collision-free complete coverage robot paths. Genetic algorithms are usually used to optimize the efficiency of the coverage path [20]–[22]. For example, Tu *et al.* [20] proposed a path planning method based on genetic algorithms with a variable length chromosome. Their algorithm can generate efficient collision-free paths for a mobile robot in both static and dynamic environments. Jimenez *et al.* [21] utilized a genetic algorithm to find the most efficient coverage path. Their algorithm can find a collision free path between two positions to allow a robot to sweep the whole free space environment. Muzaffer *et al.* [22] developed a genetic algorithm to generate efficient rectilinear coverage paths. They modeled the area using disks representing the sensor's coverage area. Their algorithm can find a path which runs through the center of each disk with minimal cost. Most of the above mentioned works use the robot body to cover the area of interest. In the ROCIM system, we are interested in ensuring the union of the camera field-of-view (FoV) covers the whole bridge deck area. There are challenges in the planning problem due to the configuration of the camera and the mobility of the robot. In this paper, we present a solution to this problem based on genetic algorithms.

The remainder of this paper is organized as follows. In Section II, we discuss the overview of the ROCIM system. Section III presents the technique to derive the coordinate. The crack detection algorithm is described in Section IV. Section V formulates the problem of the complete coverage path planning (CCPP), and then describes the proposed algorithm for this problem. Section VI provides the experimental results. Finally, Section VII proposes some ideas for future work and gives the conclusions.

## II. ROCIM SYSTEM OVERVIEW

This section presents an overview of the ROCIM system. As shown in Fig. 1, the hardware setup of the ROCIM system consists of:

- one Pioneer3-DX mobile robot;
- one PTZ (Pan-Tilt-Zoom) Canon VC50i camera;

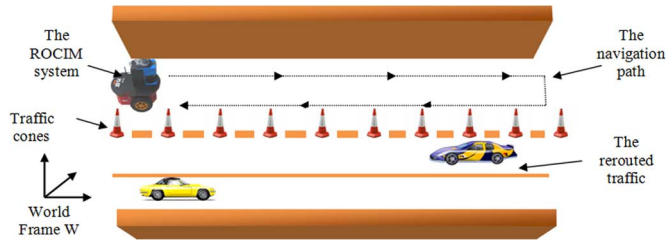


Fig. 2. The scenario of crack inspection and mapping using the ROCIM system.

- one SICK LMS-500 laser sensor;
- one laptop.

We mount the PTZ Canon VC50i camera on the Pioneer3-DX robot. It is a color camera with a resolution of  $860 \times 640$  and an optical-zoom-in factor of 26. We also integrate the laser range finder with the robot for 2D mapping and robot localization. This laser has a field of view of  $190^\circ$ , a maximum scanning frequency of 75 HZ, an operating range of 0–80 m. Its distance resolution is 1 mm and the angular resolution is  $0.25^\circ$ . The average distance error is 7 mm.

To conduct crack inspection and mapping, we will block half of the bridge, as shown in Fig. 2. 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, we assume that the robot captures an image only when it completely stops at the predetermined locations.

There are three steps during 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 surface image data at predetermined locations. The raw image data will be stored in the on-board computer or transferred to a nearby laptop computer using wireless connection.
- 3) Crack map generation: Cracks will be detected through image processing. The crack map will be created by piecing together multiple local crack maps. This step can be performed offline on the laptop computer.

The proposed crack inspection system works according to Fig. 3. To create a navigation map, we use a simultaneous localization and mapping (SLAM) algorithm [23]. The SLAM algorithm estimates the robot locations and creates the 2D map at the same time. We utilize the *Mapper3* software [24] that comes with the Pioneer robot for this purpose. After the map is created, the mobile robot can localize itself based on that map using the Monte Carlo localization (MCL) algorithm [25].

To create the crack map, we need find the relationship between the image coordinate system and the global map coordinate system. The relationship between the camera and the robot coordinate system is assumed to be fixed once the camera orientation is fixed. We denote this relationship as the camera-robot transformation. The crack locations in the image coordinate system can be mapped to the robot coordinate system through this transformation. During the inspection, the robot travels on the bridge deck and localizes itself in the 2D map.

Therefore, the relationship between the robot coordinate system and the global coordinate system can be obtained through robot localization. We denote this relationship as the robot-global coordinate system transformation. Another transformation is the image-camera coordinate system transformation that can be found from camera calibration. In order to create the crack map, we derive an image-global coordinate system transformation which is a multiplication of image-camera, camera-robot and robot-global coordinate system transformation.

After all images are collected, they will be processed using the Laplacian of Gaussian (LoG) algorithm [26], [27] to detect the cracks. The LoG algorithm finds a zero-crossing as an edge in the second differential of the image intensity. Since the cracks are detected using the LoG algorithm, the crack locations are defined in the image coordinate system. In order to create the crack map, we apply the image-global coordinate transformation to map these crack locations to the global coordinate system.

To ensure the mobile robot collects all the images on the bridge deck in an efficient manner, we formulate a complete coverage path planning (CCPP) problem for the ROCIM system. We propose robotic inspection path planning based on the genetic algorithm (RIP-GA) to solve this CCPP problem.

The RIP-GA algorithm will run off-line based on the 2D bridge deck map. The output of RIP-GA is a path which consists of a sequence of inspection configurations which include the robot locations, orientations, and camera poses.

### III. COORDINATE TRANSFORMATION

In this section, we discuss the coordinate transformation that maps the crack location from the image coordinate system to the global coordinate system. The ROCIM system involves five coordinate systems, as shown in Fig. 4. 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 ( $F_G$ ). As we mentioned before, the mobile robot travels in the inspection area and collects the images of the bridge deck surface. The location and orientation of the mobile robot can be estimated using the Monte Carlo localization (MCL) algorithm [25] which is a sampling-based method to estimate a probability density distribution of the robot locations. Each sample consists of a possible robot location and a probability that the robot currently is at that location. We use the *advanced range navigation laser* (ARNL) library [24] to implement the MCL algorithm.

Camera calibration finds the extrinsic and intrinsic parameters of a camera. Subsequently, we utilize those parameters to map all crack locations to the 2D map by assuming that the transformation between the camera coordinate system and the robot coordinate system is fixed, for a given camera orientation.

We use the Matlab Camera Calibration Toolbox to conduct the calibration [28]. After we finish 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 ( $\mu_0, \nu_0$ ) as described in (1)

$$\mathbf{P} = \begin{bmatrix} sf & 0 & \mu_0 & 0 \\ 0 & f & \nu_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \quad (1)$$

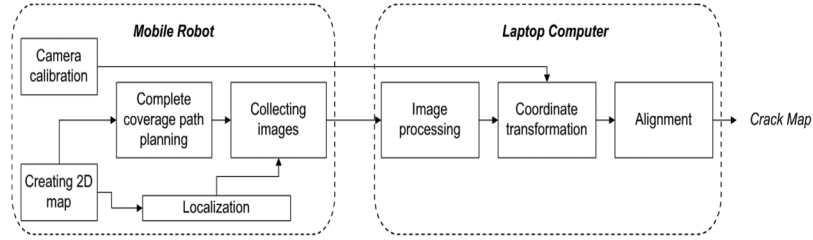


Fig. 3. The working principle of the ROCIM system.

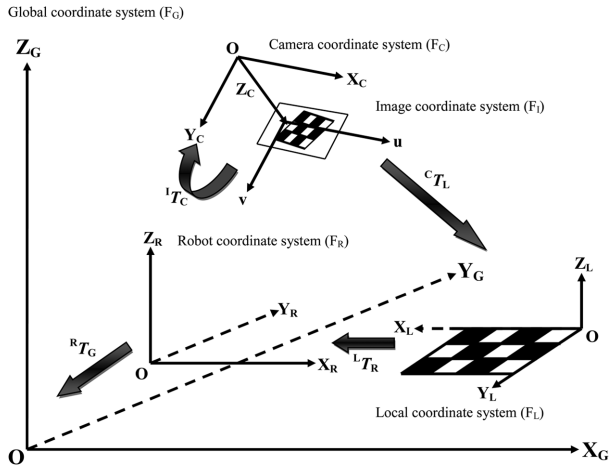


Fig. 4. Coordinate systems in the ROCIM system.

The extrinsic parameters are rotation ( $\mathbf{R}$ ) and translation ( $\mathbf{t}$ ) matrices as expressed in (2) [29]

$$\mathbf{M} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}. \quad (2)$$

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 mark all the crack locations in the global coordinate system. To do this, we start with 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 matrices as below

$${}^C T_R = {}^C T_L {}^L T_R \quad (3)$$

$${}^L T_R = (D_L^T D_L)^{-1} D_L^T D_R. \quad (4)$$

The transformation  ${}^L T_R$  can be calculated using pseudo-inverse as in (4). Here,  $D_L$  and  $D_R$  are a set of locations in the local coordinate system and the robot coordinate system, respectively. We apply another transformation from the robot coordinate system to the global coordinate system using the transformation  ${}^R T_G$ , which is defined as

$${}^R T_G = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & x_r \\ \sin(\theta) & \cos(\theta) & y_r \\ 0 & 0 & 1 \end{bmatrix} \quad (5)$$

here  $(x_r, y_r, \theta)$  are the position and heading of the robot.

Finally, from the image coordinate system to the global coordinate system, we have the following transformation:

$${}^I T_G = {}^I T_C {}^C T_R {}^R T_G. \quad (6)$$

#### IV. CRACK DETECTION

In this section, we describe how to detect cracks in each image captured by the mobile robot. 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 second derivative is very sensitive to noise. Hence, this noise should be filtered out. To achieve this, the Laplacian of Gaussian (LoG) algorithm [26] is used.

The Laplacian is a 2D isotropic measure of the second spatial derivative of an image. The Laplacian of an image highlights regions of fast intensity change and is often used for edge detection. It is often applied to an image that has first been smoothed with something approximating a Gaussian smoothing filter in order to reduce its sensitivity to noise. The Laplacian operator normally takes a single gray level image as input and produces another gray level image as output. Namely, the Laplacian  $L(x, y)$  of an image with pixel intensity values  $I(x, y)$  is given by

$$L(x, y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}. \quad (7)$$

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 different kernels 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 smoothed using a Gaussian filter before the Laplacian filter is applied. This preprocessing 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, then convolve this hybrid filter with the image to achieve the required result. By doing this we have the following 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 precalculated in advance so only one convolution needs to be performed at runtime on the image.

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

$$\text{LoG}(x, y) = -\frac{1}{\pi\sigma^4} \left[ 1 - \frac{x^2 + y^2}{2\sigma^2} \right] e^{-\frac{x^2 + y^2}{2\sigma^2}} \quad (8)$$

where  $\sigma$  is the standard deviation and it is selected based on a trial and error method. The details of the selection of  $\sigma$  will be discussed in Section VI-A.

If the image is presmoothed by a Gaussian low-pass filter, then we have the LoG operation defined as

$$(K_{\nabla^2} ** (G_{\sigma} ** I)) = (K_{\nabla^2} ** G_{\sigma}) ** I = (\nabla^2 G_{\sigma}) ** I \quad (9)$$

where  $K_{\nabla^2}$  is the Laplacian kernel with a size of  $15 \times 15$ , and  $G_{\sigma}$  is the Gaussian kernel with a size of  $17 \times 17$ , and  $\nabla^2 G_{\sigma}(x, y)$  is computed as

$$\nabla^2 G_{\sigma}(x, y) = \frac{1}{2\pi\sigma^4} \left[ \frac{x^2 + y^2}{\sigma^2} - 2 \right] e^{-\frac{x^2 + y^2}{2\sigma^2}}. \quad (10)$$

To find the places where the value of the Laplacian passes through zero points, we use the zero crossing detector. We know that such points often occur at edges where the intensity of the image changes rapidly. Zero crossings always lie on closed contours, 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. To avoid detection of insignificant edges, only the zero crossings with corresponding first derivative above a certain threshold  $T$  are selected as edge points. The edge direction is obtained by using the direction in which zero crossing occurs. The selection of  $T$  is discussed in Section VI-A for real crack detection. The initial input to 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 filter used for the smoothing stage of this operator. As the level of smoothing is increased, then less zero crossing contours were found, and those that remain correspond to features of larger and larger scale in the image.

## V. COMPLETE COVERAGE PATH PLANNING

### A. Problem Formulation

The goal of complete coverage path planning (CCPP) is to ensure the union of camera FoV covers the whole bridge deck surface. There are several issues that we need address in this CCPP problem. First, the camera is typically mounted on top of the mobile robot, as shown in the top part of Fig. 5. Therefore, there is a blind spot between the center location of the robot body and the center of the FoV. Second, to allow the mobile robot to detect small cracks, the camera should zoom in, which makes the size of the FoV smaller than the size of the robot body. The bottom part of Fig. 5 illustrates these two issues. To simplify the problem, the area of interest is partitioned into cells which have the same size as the FoV.

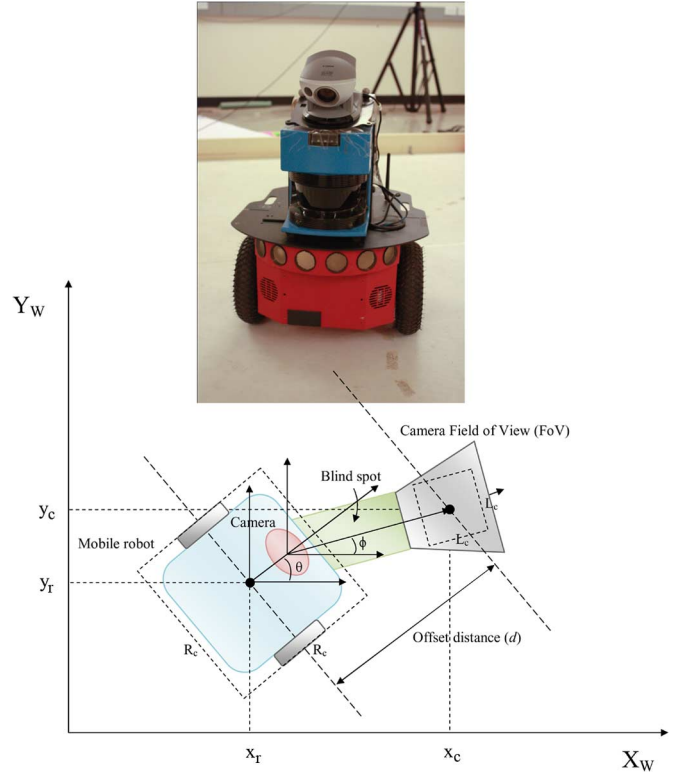


Fig. 5. Projection of mobile robot and camera FoV to an  $XY$  plane.

We define several concepts to formulate the CCPP problem and develop the solution to it. These definitions are described as follows.

*Definition 1: Configuration of Inspection (CoI):* As shown in Fig. 5, let  $(x_r, y_r) \in \mathbf{R}^2$  be the center location of the mobile robot;  $\theta \in [0, 2\pi]$  be the heading of the mobile robot; and  $\phi \in [-\pi/2, \pi/2]$  be the pan angle of the camera. Then, a *configuration of inspection (CoI)* is defined as

$$\text{CoI}^i \triangleq (x_r, y_r, \theta, \phi)$$

where  $i$  is the index of a cell.

*Definition 2: Motion Template:* A motion template,  $\text{MT}(i, j)$ , is the motion plan of the mobile robot and the camera which realizes the transition from  $\text{CoI}^i$  to  $\text{CoI}^j$ :

$\text{MT}(i, j) \triangleq \text{CoI}^i \rightarrow \text{CoI}^j$ ,  $j \in \mathbf{N}(i)$ , here  $\mathbf{N}$  is the neighborhood of cell  $i$ , as shown in Fig. 7(a).

*Definition 3: Inspection Path:* An inspection path, IP, is a sequence of motion templates that ensure the union of camera FoVs cover the whole area of interest. An IP is defined as:

$$\text{IP} \triangleq [\text{MT}(i_1, i_2), \text{MT}(i_2, i_3), \dots, \text{MT}(i_{n-1}, i_n)]$$

where  $(i_1, i_2, \dots, i_n)$  is the camera path (a sequence of all the cells). An example camera path is shown in Fig. 7(b). In Fig. 6, we show the 12 CoIs, where each CoI has a different robot heading ( $\theta$ ) and camera pan angle ( $\phi$ ). Here,  $\theta \in \{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$  and  $\phi \in \{-45^\circ, 0^\circ, 45^\circ\}$ . We encode the CoIs using the alphabet from A to L:  $\mathbf{S} \in \{A, B, C, D, E, F, G, H, I, J, K, L\}$ , as shown in Fig. 6. The red (or darker) square is the mobile robot and the green (or

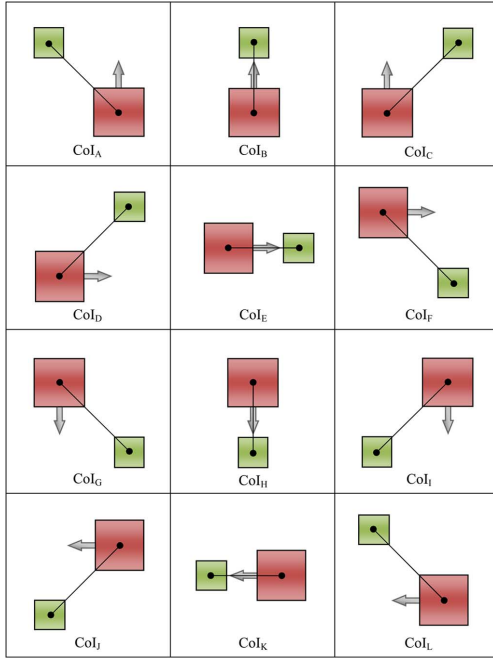


Fig. 6. Twelve configuration of inspections.

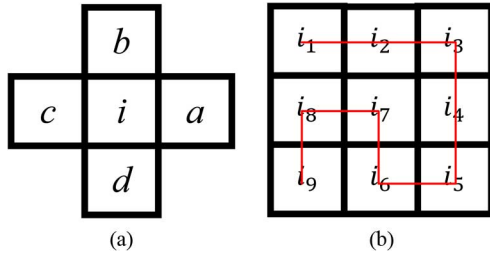


Fig. 7. (a) Four directions of camera path define four neighbor cells. (b) An example camera path to cover the whole area.

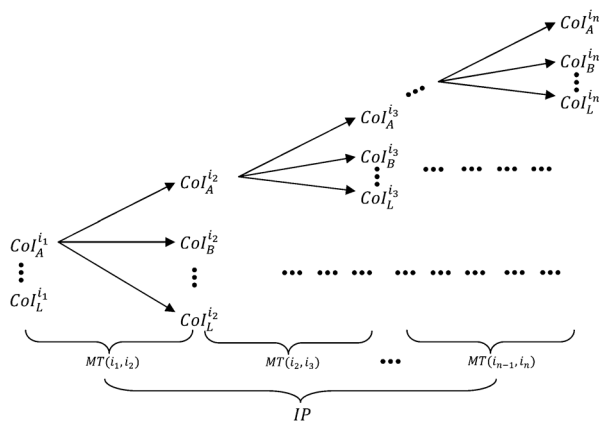


Fig. 8. Relationship between CoI, motion templates (MT), and IP.

lighter) square is the FoV. The relationship between the three definitions is shown in Fig. 8. In a free space, all the 12 CoIs can be applied to cover a cell  $i$ .

The problem of complete coverage path planning (CCPP) for the ROCIM system is concerned about how to find an efficient inspection path in order to have complete coverage of the cells. This can be done by selecting a combination of CoIs which

minimizes an objective function which consists of three components: traveling distance, robot turns, and camera turns. The last component (camera turns) can be safely ignored because the camera works in parallel to the mobile robot. Therefore, 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.

Finding the optimal solution to this CCPP problem is NP-hard and the computational cost increases exponentially with the size of the problem. Therefore, in this paper, we are interested in finding a suboptimal solution.

### B. Robotic Inspection Path Planning Based on Genetic Algorithm (RIP-GA)

Existing approaches cannot be directly applied to solve the CCPP problem since they only concern about the coverage of the robot body [17], [22], [21], [30], [31]. We present our RIP-GA approach which gives a suboptimal solution to the CCPP problem.

The RIP-GA approach finds an inspection path that minimizes the total traveling distance and the number of robot turns by selecting a sequence of CoIs. The proposed algorithm extends the idea of the proposed RPP-CP (robot path planning based on the camera path) in our previous work [27]. Both approaches assume that the camera path is planned first to completely cover the area of interest. Then, the robot motion and camera motion are planned according to the camera path.

To select the CoIs, we utilize a genetic algorithm, which is an optimization method mimicking the natural evolution. The solution is generated using similar techniques of mutation, selection and crossover as reported in [32]. We model the genetic algorithm as follows.

- The CoI is defined as the gene.
- The chromosome ( $C_r^I$ ) is a sequence of CoIs,  $C_r^I = (CoI^{i_1}, CoI^{i_2}, CoI^{i_3}, \dots, CoI^{i_n})$  and the length of  $C_r^I$  is the total number of cells ( $n$ ).

In Fig. 9, we show an example of calculating the objective function calculation from one CoI to another CoI. Fig. 10(a) illustrates the chromosome which represents a solution to this CCPP problem. For each generation the genetic algorithm tries to find a better chromosome by minimizing the objective function

$$F = \sum_{j=1}^n (\alpha \|q(i_j, i_{j+1})\| + \beta \|u(i_j, i_{j+1})\|). \quad (11)$$

Here, the functions  $q$  and  $u$  are the traveling distance and the number of robot turns, respectively. The variables  $\alpha$ ,  $\beta$  are the corresponding weight values. The pseudocode of the RIP-GA algorithm is shown in Algorithm 1. In this algorithm, the crossover operation is applied to mate the nearest pair, which can produce the offspring ( $\omega^I, \omega^{I+1}$ )

$$\omega^I = c_1 \otimes C_r^{I+1} \oplus (1 - c_1) \otimes C_r^I \quad (12)$$

$$\omega^{I+1} = c_1 \otimes C_r^I \oplus (1 - c_1) \otimes C_r^{I+1}. \quad (13)$$

Here,  $\otimes$  is the crossover operator;  $\oplus$  is the combination operator; and  $c_1 \in [0, 1]$  represents the proportion of the chromosome where the crossover takes place. For example, let  $c_1 = 0.3$ , then the chromosome  $\omega^I$  takes 70% genes from  $C_r^I$  to mate

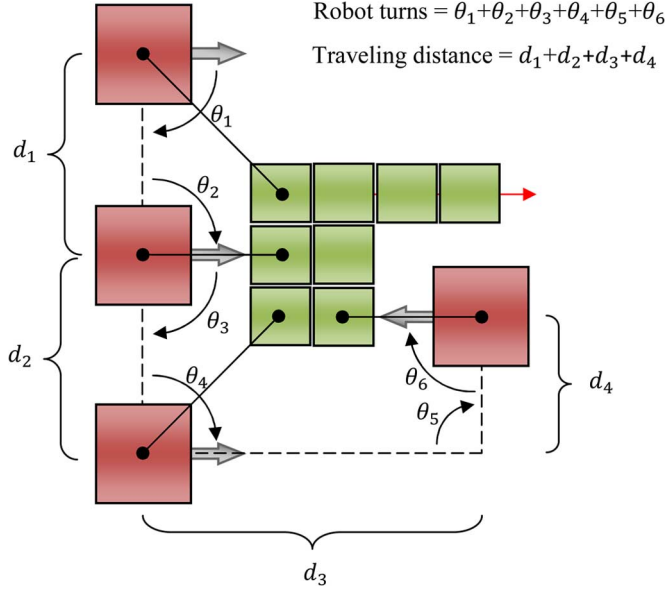
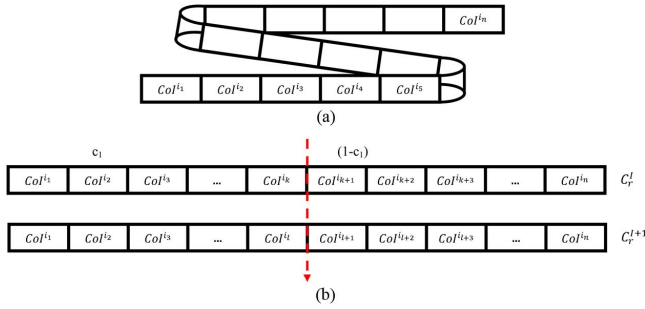


Fig. 9. Calculation of the objective function.

Fig. 10. (a) Chromosome ( $C_r^I$ ). (b) Chromosome crossover.

with 30% of the  $C_r^{I+1}$ . The illustration of the cross-over operation is shown in Fig. 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 are applied, the offsprings ( $\omega^I, \omega^{I+1}$ ) are evaluated with the parents ( $C_r^I, C_r^{I+1}$ ). For each offspring, we calculate and compare the fitness value with that of the parents. We discard two of these chromosomes (offsprings and parents) which have the most fitness value. In step 2.6, we remove some chromosomes in the population which have the most fitness value and new chromosomes are introduced to fill in. The algorithm continuously runs until the fitness value of the top chromosome does not change for  $Q$  iterations.

We compare the genetic algorithm with a Greedy algorithm. The Greedy algorithm searches locally for the minimum fitness value for the next CoI along the camera path. It is robust but usually gets stuck in local minima. The pseudocode of the RIP-Greedy algorithm is shown in Algorithm 2.

## VI. EXPERIMENTAL RESULTS

We conducted both simulations and experiments to validate the proposed ROCIM system and the associated algorithms. In the first experiment, we evaluate the crack detection algorithm.

### Algorithm 1: RIP-GA

- 
- Step 1.** Generate a camera path  $CP(n)$  using boustrophedon motion, where  $CP = (i_1, i_2, i_3, \dots, i_n), n \in \mathbf{Z}$ .  
**for each**  $CP(n)$  **do**  
 └ Find a set of feasible  $CoI_{(S)}$
- Step 2.** Generate a population of  $M$  genetic strings randomly from  $CP$ ;  $P^I = (\omega_1^I, \omega_2^I, \omega_3^I, \dots, \omega_N^I)$ ;  $I = 1, 2, \dots, M$ .  
**while the genetic algorithm is not converged do**  
 1. Compute the objective function using Equation 11  
 2. Rank the genetic strings from top to bottom,  $C^I (I = 1, 2, \dots, M)$   
 3. Put the top chromosome to the next generation (Elitism).  
 4. Mate the nearest pairs.  
 5. Mutate the 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**.
- 

**Step 3.** Generate the inspection path based on the CoI sequences (Inspection Path).

---

### Algorithm 2: RIP-Greedy

- 
- Step 1.** Generate a camera path  $CP(n)$  using boustrophedon motion, where  $CP = (i_1, i_2, i_3, \dots, i_n), n \in \mathbf{Z}$ .  
**for each**  $CP(n)$  **do**  
 └ Find a set of feasible  $CoI_{(S)}$
- Step 2.** Find the motion templates for each cell:  
 $IP = [\min(MT_{ij}(t = 1)), \min(MT_{ij}(t = 2)), \dots, \min(MT_{ij}(t = (N - 1)))]$ .
- Step 3.** Generate the robot trajectory based on the CoI (Inspection Path).
- 

In the second experiment, we evaluate the complete coverage path planning using RIP-GA algorithm (Algorithm 1) in both obstacle-free and obstacle environments. Finally, we evaluate the ROCIM system in both indoor and outdoor environments.

#### A. Real Crack Detection

In order to evaluate the performance of the ROCIM system, we first tested our crack detection algorithm using real crack images collected from a bridge deck. The original image is shown in Fig. 11(a). The LoG algorithm is applied to the image, and the results are shown in Fig. 11(b). The detected crack is superimposed on the original image in Fig. 11(c). Here, the result clearly shows that the LoG algorithm successfully detects the crack in the image.

In the LoG algorithm, we need find the optimal parameters such as  $\sigma$  and  $T$ . In Fig. 12, we show the result of the LoG algorithm using various  $\sigma$  and  $T$  values. For example, as shown in Fig. 12, using  $\sigma \leq 1.5$  and  $T = 2$ , the result shows too much noise and the crack is barely detectable. On the other hand, if  $\sigma$

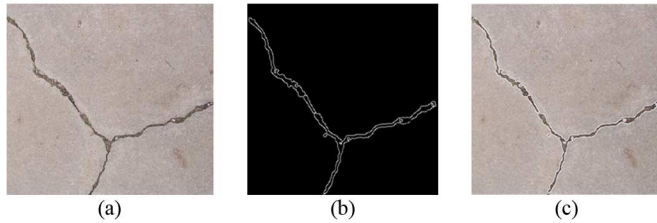


Fig. 11. Crack detection result on a real bridge deck. (a) Original image. (b) Crack detection result. (c) Cracks are superimposed with the original image.

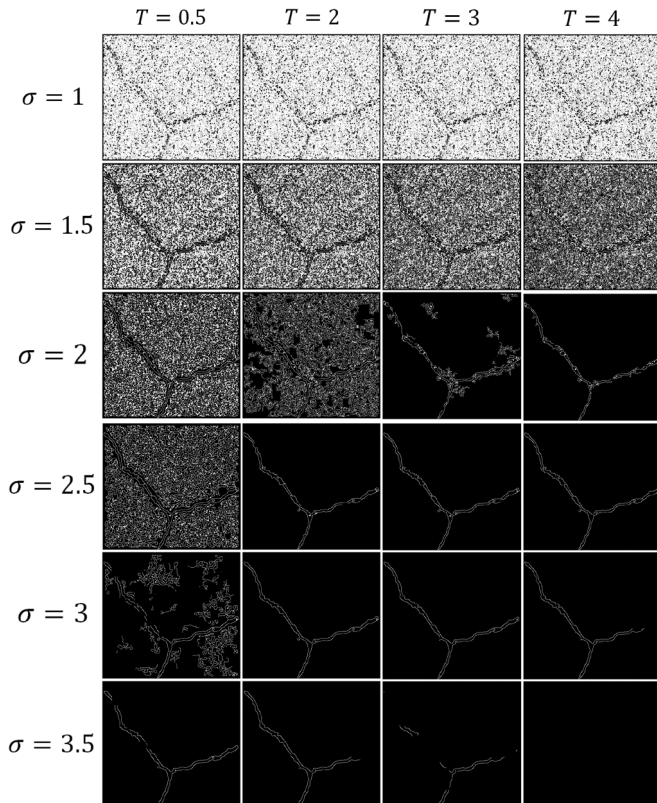


Fig. 12. Comparison of the detection results with different LoG parameters.

and  $T$  are too large, the LoG removes most of the real cracks. Therefore, the parameters we set for the LoG algorithm are:  $\sigma = 2.5$  and  $T = 3$ . With these parameters, we find that the smallest detectable crack is around 1 mm in width.

### B. Validation of the RIP-GA Algorithm

In this section, we test our RIP-GA algorithm in both obstacle-free and obstacle environments and then compare it with the RIP-Greedy algorithm.

1) *Obstacle-Free Environment*: We consider a rectangular environment without obstacles, as shown in Fig. 13(a). There are 55 cells in this environment and each cell has to be covered in order to have complete coverage. First, the camera path is planned using boustrophedon motion from top right to bottom left. We initialized the following parameters for the RIP-GA algorithm: the number of chromosomes  $M = 500$ , the weight parameters  $\alpha = 0.94$ ,  $\beta = 1.1$ , and the stopping condition  $Q = 30$ . The values of  $\alpha$  and  $\beta$  are determined by measuring the time required for traveling a unit distance and making the

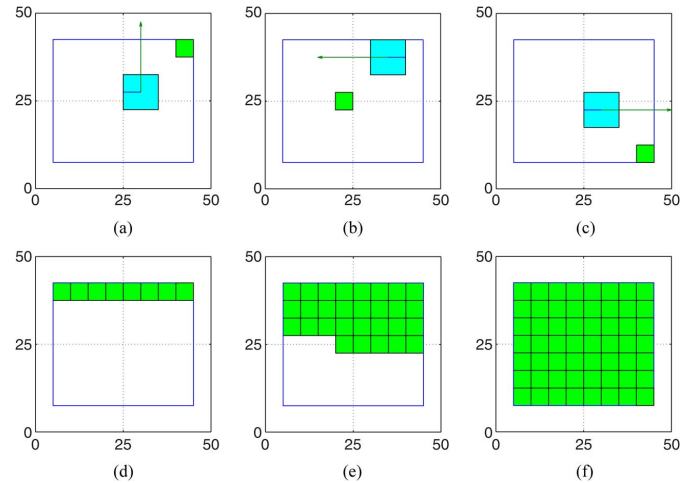


Fig. 13. Camera path planning result in the obstacle-free environment: (a), (b) and (c) are the snapshots of the mobile robot position, and (d), (e) and (f) are the snapshots of the covered camera view.

TABLE I  
THREE SOLUTIONS WITH THE MINIMUM FITNESS VALUE

Cols
LLLLBDDCDBLLLLLALLBDDDDFFHJJJJJJHFFFGGFFHJJJJJJHFFF
LLLLLBDCCDBLLLAALLBDDDCFFHJJJJJJHFFFHJJJJJJHFFF
LLLLBDDCDBLLLLLLLLLBDCCFFHJJJJJJHFFFHJJJJJJHFFF

robot to finish a  $90^\circ$  turns. Fig. 13 shows the camera path planning result. For each generation, we discard 50% of the population which has the least rank and introduce new chromosomes to fill in. We run the RIP-GA algorithm to find the suboptimal solution of the inspection path. After 130 iterations, the RIP-GA algorithm converges at 59.15. Moreover, the RIP-GA algorithm finds multiple solutions that have the same fitness value and they are shown in Table I.

We compare the result of the RIP-GA algorithm with that of the RIP-Greedy algorithm. The comparison is shown in Table II. We can see that the result of the RIP-GA algorithm is clearly better than that of the RIP-Greedy algorithm in terms of the number of robot turns, traveling distance and fitness value.

2) *Obstacle Environment*: In this experiment, we evaluate the proposed algorithm in the obstacle environment, as shown in Fig. 14. Here, the obstacles are represented by red squares. First, we decompose the environment into eight regions. Then, the camera path is planned using boustrophedon motion from bottom to top for each region. In each region, we indicate the starting and ending points by gray/lighter and black/darker bullets, respectively. To cover the whole area, the robot needs to travel all regions. After the robot finishes one region, it goes to the nearest region.

In this experiment, the initialization parameters are the same as those in the obstacle-free environment except the number of chromosomes ( $M$ ). Since the obstacle environment is decomposed into several regions, we run the RIP-GA algorithm separately for each region. We use different values of  $M$  due to the different sizes of the region. Here, we use  $M = 1000$  for region 1 and 3, and  $M = 500$  for the rest of regions. The RIP-GA algorithm initializes the search with a random sample for each

TABLE II  
COMPARISON OF COMPLETE COVERAGE PATH PLANNING

	RIP-GA	RIP-Greedy	RIP-GA	RIP-Greedy	RIP-GA	RIP-Greedy	RIP-GA	RIP-Greedy
	Trial 1	Trial 1	Trial 2	Trial 2	Trial 3	Trial 3	Trial 4	Trial 4
The number of robot turns	26	29	24	27	28	31	25	30
Traveling distance	325	395	322	389	341	402	343	422
Fitness value	59.15	65.73	58.99	63.19	62.01	69.54	65.85	79.04

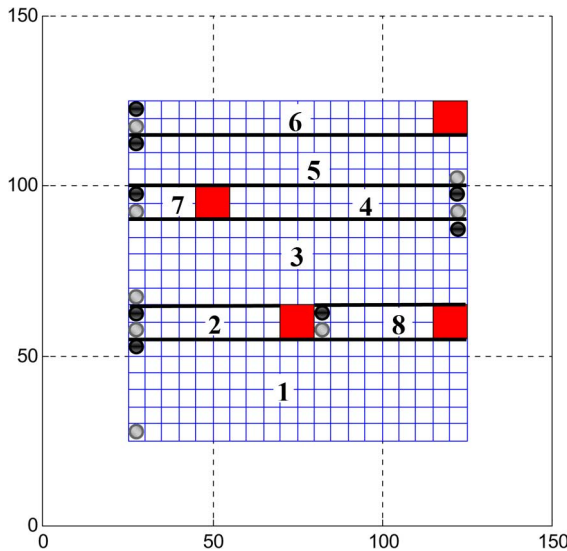


Fig. 14. Map decomposition for an obstacle environment.

TABLE III  
COMPARISON OF RIP-GA AND RIP-GREEDY ALGORITHMS

Region	RIP-GA algorithm	RIP-Greedy algorithm
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

cell. In the obstacle environment, some CoIs cannot be used to cover the cells in the corner or around the boundary. Therefore, we apply a CoI filtering to remove these infeasible CoIs. We run the RIP-GA and RIP-Greedy algorithm to find out the best combination of CoIs to generate the inspection path. The comparison of both algorithms based on the objective function is shown in Table III. It can be seen that the RIP-GA algorithm gives better solution than the RIP-Greedy algorithm in term of fitness value. The camera path planning results of the RIP-GA algorithm in the obstacle environment are shown in Fig. 15.

### C. Overall Evaluation of the ROCIM System

In this section, we evaluate the ROCIM system through both indoor and outdoor experiments. The indoor environment for this experiment is shown in Fig. 16. We use traffic cones to set up the inspection area. We also simulate the cracks on a wooden floor. The procedure of the experiment is described as follows: First, the mobile robot creates a 2D map of this environment using the *ARNL* and *Mapper3* software [24]. Second,

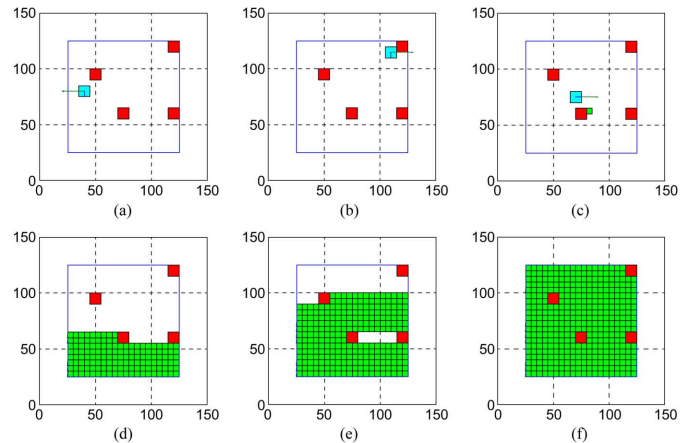


Fig. 15. Camera path planning in the obstacle environment: (a), (b) and (c) are the snapshot of covered camera view, and (d), (e) and (f) are the snapshot of robot position.

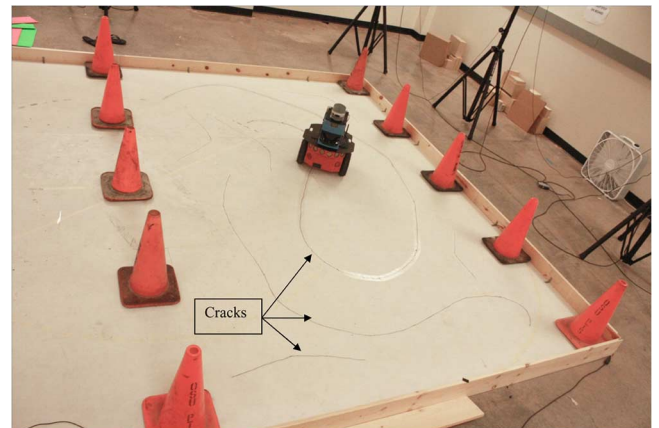


Fig. 16. Laboratory environment to test the ROCIM system.

we initialize the RIP-GA algorithm by defining the size and the number of cells in the 2D map. For this experiment, the inspection area is decomposed into 28 cells and the size of each cell is  $550 \times 550$  mm. Third, we run the RIP-GA algorithm to find the suboptimal inspection path. Fig. 17 shows the result of the RIP-GA algorithm which consists of robot locations (red star) and center of FoV locations (blue star). Last, the robot path is given to the mobile robot to follow using the navigation library [24]. During the experiment, the linear speed of the robot is 0.1 m/s and the angular speed is 15 degree/s. To avoid blurring the images, the robot is programmed to stop for 1 s for capturing an image. For the whole inspection area (4 m by 2.5 m), the total inspection time is 23 min.

Fig. 18 shows the obtained crack map. The blue lines/dots represent the 2D map and the black lines/dots represent the

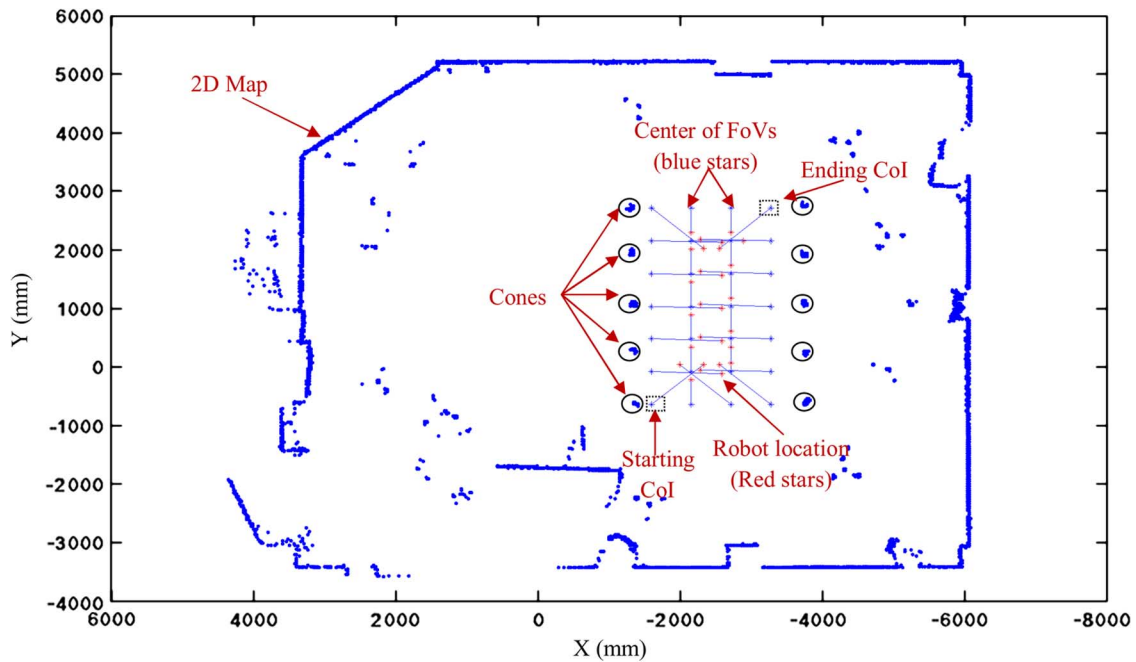


Fig. 17. Robot locations for each cell in the 2D map.

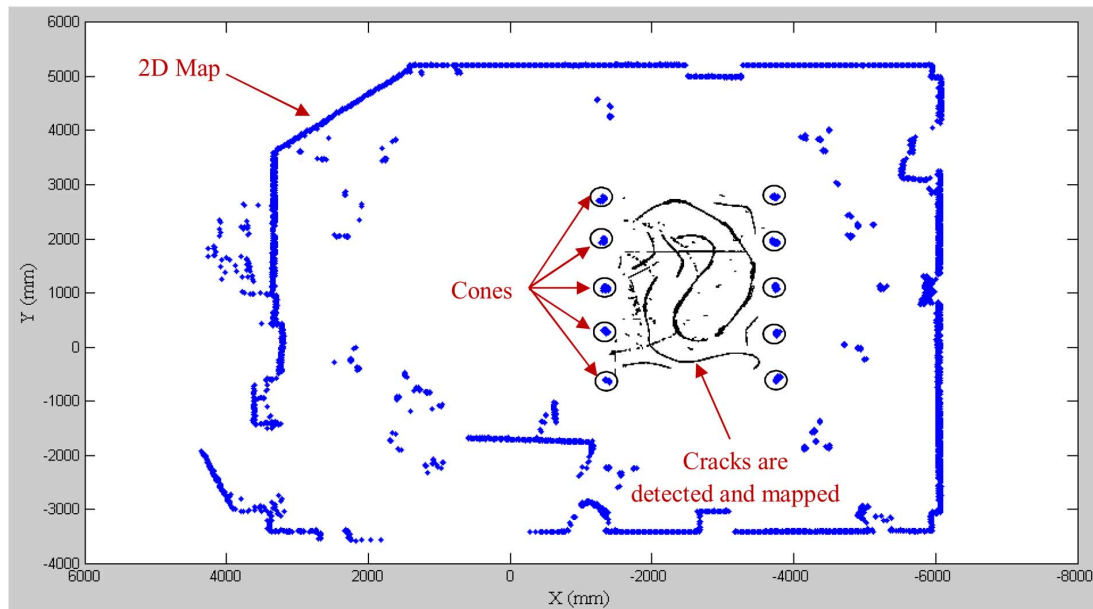


Fig. 18. The global crack map.

cracks. From the results we can see that the proposed ROCIM system successfully detects the individual cracks and builds the global crack map.

In outdoor environment testing, we deploy the mobile robot to detect the cracks on a concrete road surface. Here, the ROCIM system has to handle an open environment and moving objects such walking people. The main problem with this experiment is robot localization. This problem is caused by: 1) the limited sensing range of the laser sensor which is not sufficient to scan the whole area and 2) the dynamic environment which introduces significant noises to the MCL algorithm. To overcome these problems, we use traffic cones to play as landmarks that

can improve the localization and reduce the noise. The experiment setup is shown in Fig. 19(a). The whole inspection area has a size of 3.5 m by 2.5 m and the total inspection time is about 20 min. There are total 30 images collected at the corresponding robot's pose (position and orientation). Similar to the indoor result, the ROCIM system builds a crack map, as shown in Fig. 19(b).

## VII. CONCLUSION AND FUTURE WORK

In this paper, we introduced a robotic crack inspection and mapping (ROCIM) system. The ROCIM system provides an overall solution to bridge deck crack inspection. First, the crack

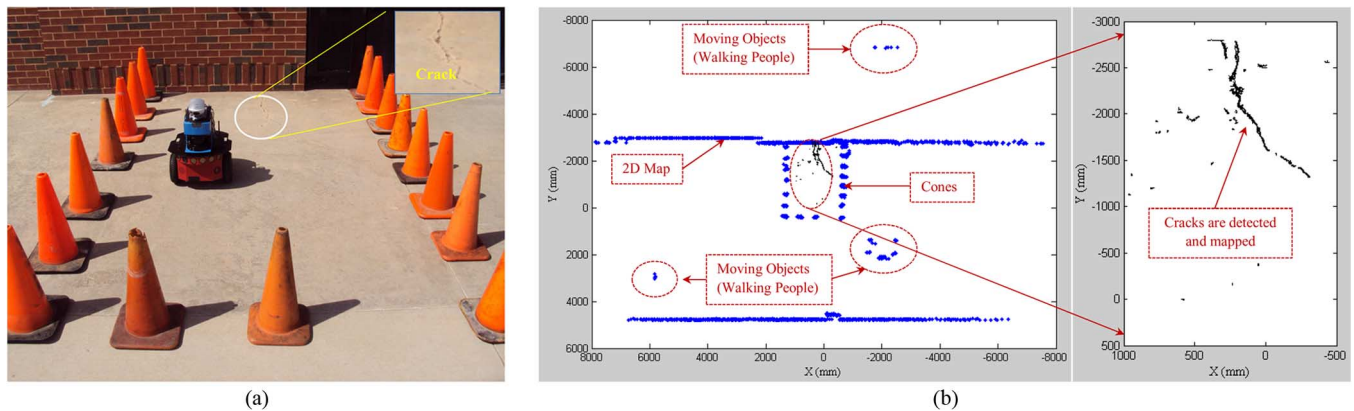


Fig. 19. (a) Experiment setup to evaluate the ROCIM system in the outdoor environment. (b) Crack map for outdoor environment.

detection algorithm works well for real cracks through the experiment and simulation evaluation. Second, we propose robotic inspection path planning based on a genetic algorithm (RIP-GA) to ensure the mobile robot collects all the images efficiently in the area of interest. We also validate the proposed RIP-GA algorithm and then compare it with the Greedy algorithm in obstacle free and obstacle environments. The results show that the genetic algorithm performs better than Greedy algorithm. Third, both indoor and outdoor tests are conducted to validate the proposed ROCIM system. In the future, we will further improve the crack detection algorithm, especially in various ambient lighting conditions. We need enhance the robustness of the crack detection algorithm in such conditions. On the other hand we will utilize Non-Destructive Evaluation (NDE) sensors such as Impact Echo and Ultrasonic Surface Wave to detect vertical cracks (crack depth) and delamination of the bridge deck [33]. Also, we will address the problem of degraded localization accuracy due to moving objects. One of the best solutions is using Extended Kalman Filter (EKF) to fuse various data from different sources such as Differential Global Positioning System (DGPS), Inertial Measurement Unit (IMU), and robot wheel encoders to output the accurate and smooth localization of the robot on the bridge [34], [33].

## REFERENCES

- [1] Wikipedia "Mississippi River Bridge," 2007. [Online]. Available: <http://en.wikipedia.org/wiki/i-35w>
- [2] V. Giurgiutiu, C. A. Rogers, Y. J. Chao, M. A. Sutton, and X. Deng, "Adaptive health monitoring concepts for spot-welded and weld-bonded structural joints," in *Proc. ASME Aerosp. Division*, 1997, vol. 54, pp. 99–104.
- [3] C. R. Farrar, H. Sohn, and S. W. Doebling, "Structural health monitoring at Los Alamos National Laboratory," in *U.S.-Korea Conf. Sci. Technol., Entrepreneurship and Leadership*, Chicago, IL, USA, Sep. 2–5, 2000, pp. 1–11.
- [4] H. Sohn, C. R. Farrar, M. L. Fugate, and J. J. Czarnecki, "Structural health monitoring of welded connections," in *Proc. 1st Int. Conf. Steel Composite Structures*, Pusan, Korea, Jun. 14–16, 2001.
- [5] E. Sazonov, K. Janoyan, and R. Jha, "Wireless intelligent sensor network for autonomous structural health monitoring," in *Proc. SPIE—Int. Soc. Opt. Eng.*, 2004, vol. 5384, no. 1, pp. 305–314.
- [6] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, "A wireless sensor network for structural monitoring," in *Proc. 2nd Int. Conf. Embedded Networked Sensor Syst. (SenSys'04)*, 2004, pp. 13–24.
- [7] D. R. Huston, "Adaptive sensors and sensor networks for structural health monitoring," in *Proc. SPIE—Int. Soc. Opt. Eng.*, 2001, vol. 4512, pp. 203–211.
- [8] W. Sheng, H. Chen, and N. Xi, "Navigating a miniature crawler robot for engineered structure inspection," *IEEE Trans. Autom. Sci. Eng.*, vol. 5, no. 2, pp. 368–373, Apr. 2008.
- [9] Y. Yu, J. Gu, G. K. I. Mann, and R. G. Gosine, "Development and evaluation of object-based visual attention for automatic perception of robots," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 2, pp. 365–379, Apr. 2013.
- [10] S. N. Yu, J. H. Jang, and C. S. Han, "Auto inspection system using a mobile robot for detecting concrete cracks in a tunnel," *Autom. Construction*, vol. 16, pp. 255–261, 2007.
- [11] S. K. Sinha and P. W. Fieguth, "Automated detection of cracks in buried concrete pipe images," *Autom. Construction*, vol. 15, pp. 58–72, 2006.
- [12] P. C. Tung, Y. R. Hwang, and M. C. Wu, "The development of a mobile manipulator imaging system for bridge crack inspection," *Autom. Construction*, vol. 11, pp. 717–729, 2002.
- [13] J. H. Lee, J. M. Lee, J. W. Park, and Y. S. Moon, "Efficient algorithms for automatic detection of cracks on a concrete bridge," in *Proc. 23rd Int. Tech. Conf. Circuits/Syst., Comput. Commun.*, 2008, pp. 1213–1216.
- [14] 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," *Autom. Construction*, vol. 18, pp. 929–941, 2009.
- [15] H. G. Sohn, Y. M. Lim, K. H. Yun, and G. H. Kim, "Monitoring crack changes in concrete structures," *Comput.-Aided Civil and Infrastructure Eng.*, vol. 20, pp. 52–61, 2005.
- [16] A. Ito, Y. Aoki, and S. Hashimoto, "Accurate extraction and measurement of fine cracks from concrete block surface image," in *Proc. IECON'02*, 2002, pp. 77–82.
- [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," in *Proc. Int. Conf. Field Service Robot.*, 1997.
- [19] S. X. Yang and C. Luo, "A neural network approach to complete coverage path planning," *IEEE Trans. Systems, Man, Cybern., Part B: Cybern.*, vol. 34, no. 1, pp. 718–724, Feb. 2004.
- [20] J. Tu and S. X. Yang, "Genetic algorithm based path planning for a mobile robot," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2003, vol. 1, pp. 1221–1226.
- [21] P. A. Jimenez, B. Shirinzadeh, A. Nicholson, and G. Alici, "Optimal area covering using genetic algorithms," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics*, 2007, pp. 1–5.
- [22] K. Muzaffar, O. Metin, Y. Ahmet, and P. Osman, "Pattern-based genetic algorithm approach to coverage path planning for mobile robots," in *Computational Science—ICCS 2009*. Berlin/Heidelberg: Springer, 2009, vol. 5544, pp. 33–42.
- [23] S. Thrun, "Simultaneous localization and mapping," *Robot. Cognitive Approaches to Spatial Mapping*, vol. 38, pp. 113–141, 2008.
- [24] Mobile Robot Inc., 2013. [Online]. Available: <http://www.mobilerobots.com/>
- [25] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust monte carlo localization for mobile robots," *Artif. Intell.*, vol. 128, pp. 99–141, 2001.

- [26] D. A. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*. Englewood Cliffs, NJ, USA: Prentice-Hall, 2003.
- [27] R. S. Lim, H. M. La, Z. Shan, and W. Sheng, "Developing a crack inspection robot for bridge maintenance," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 6288–6293.
- [28] CALTECH Computational Vision Group, "Camera calibration toolbox for Matlab," 2010. [Online]. Available: [http://www.vision.caltech.edu/bouguetj/calib\\_doc](http://www.vision.caltech.edu/bouguetj/calib_doc)
- [29] J. Heikkilä, "Geometric camera calibration using circular control points," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, pp. 1066–1077, 2000.
- [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 *Proc. 7th Asian Control Conf.*, 2009, pp. 1468–1473.
- [31] T. Oksanen and A. Visala, "Coverage path planning algorithms for agricultural field machines," *J. Field Robot.*, vol. 26, pp. 651–668, 2009.
- [32] H. M. La, T. H. Nguyen, C. H. Nguyen, and H. N. Nguyen, "Optimal flocking control for a mobile sensor network based on a moving target tracking," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, San Antonio, TX, USA, Oct. 11–14, 2009, pp. 4801–4806.
- [33] H. M. La, R. S. Lim, B. B. Basily, N. Gucunski, J. Yi, A. Maher, F. A. Romero, and H. Parvardeh, "Mechatronic and control systems design for an autonomous robotic system for high-efficiency bridge deck inspection and evaluation," *IEEE Trans. Mechatronics*, 2013, to be published.
- [34] H. M. La, R. S. Lim, B. B. Basily, N. Gucunski, J. Yi, A. Maher, F. A. Romero, and H. Parvardeh, "Autonomous robotic system for high-efficiency non-destructive bridge deck inspection and evaluation," in *Proc. IEEE Int. Conf. Autom. Sci. Eng. (CASE)*, Madison, WI, USA, Aug. 17–21, 2013.



**Ronny Salim Lim** received the B.S. degree from Pelita Harapan University, Tangerang, Indonesia, in 2008 and the M.S. degree in electrical and computer engineering from Oklahoma State University, Stillwater, OK, USA, in 2011.

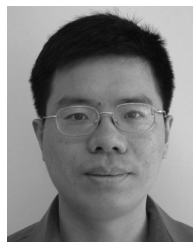
He is currently an Application Developer with the Center for Advanced Infrastructure and Transportation, Rutgers University, Piscataway, NJ, USA. He was a Research Assistant with the Laboratory for Advanced Sensing, Computation and Control, where he conducted research on robotics crack inspection, intelligent transportation systems, and multi-agent control.



**Hung Manh La** (M'09) received the B.S. and M.S. degrees in electrical engineering from Thai Nguyen University of Technology, Thai Nguyen, Vietnam, in 2001 and 2003, respectively, and the Ph.D. degree in electrical engineering from Oklahoma State University, Stillwater, OK, USA, in 2011.

He is currently a Research Associate with the Center for Advanced Infrastructure and Transportation (CAIT), Rutgers University, where he started in September 2011 to August 2012 as a Postdoctoral Researcher. Prior to coming to CAIT, he was a Lecturer in the Electrical Engineering Department, Thai Nguyen University of Technology, Vietnam (2001–2007). His research interests include mobile robotic systems; mobile sensor networks; cooperative control, learning, and sensing; and intelligent transportation systems. He is the author of more than 25 papers in major journals, book chapters, and international conferences.

Dr. La received two Best Paper Awards at the 2009 and 2010 Conferences on Theoretical and Applied Computer Science, and One Best Paper Presentation of the network control session at the 2009 American Control Conference. He has been actively involved in research projects with the Federal Highway Administration, the National Institute of Standards and Technology, the Department of Transportation, the Department of Defense, and the National Science Foundation.



**Weihua Sheng** (SM'08) received the B.S. and M.S. degrees in electrical engineering from Zhejiang University, Zhejiang, China, in 1994 and 1997, respectively, and the Ph.D. degree in electrical and computer engineering from Michigan State University, East Lansing, MI, USA, in May 2002.

He is an Associate Professor with the School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK, USA. During 1997–1998, he was a Research Engineer at the R&D Center, Huawei Technologies Co., China. From 2002

to 2006, he taught in the Electrical and Computer Engineering Department, Kettering University (formerly General Motors Institute). He has participated in organizing various IEEE international conferences and workshops in the area of intelligent robots and systems. He is the author of one U.S. patent and more than 120 papers in major journals and international conferences. His current research interests include wearable computing, mobile robotics, human robot interaction, and intelligent transportation systems. His research is supported by the National Science Foundation, the Department of Defense, DEPSCoR, the Department of Transportation, etc.

Prof. Sheng is currently an Associate Editor for the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING.