Concrete Defects Inspection and 3D Mapping Using CityFlyer Quadrotor Robot

Liang Yang, Bing Li, *Member, IEEE*, Wei Li, Howard Brand, Biao Jiang, and Jizhong Xiao, *Senior Member, IEEE*

***Abstract*—The concrete aging problem has gained more attention in recent years as more bridges and tunnels in the United States lack proper maintenance. Though the Federal Highway Administration requires these public concrete structures to be inspected regularly, on-site manual inspection by human operators is time-consuming and labor-intensive. Conventional inspection approaches for concrete inspection, using RGB image-based thresholding methods, are not able to determine metric information as well as accurate location information for assessed defects for conditions. To address this challenge, we propose a deep neural network (DNN) based concrete inspection system using a quadrotor flying robot (referred to as CityFlyer) mounted with an RGB-D camera. The inspection system introduces several novel modules. Firstly, a visual-inertial fusion approach is introduced to perform camera and robot positioning and structure 3D metric reconstruction. The reconstructed map is used to retrieve the location and metric information of the defects. Secondly, we introduce a DNN model, namely AdaNet, to detect concrete *spalling* and cracking, with the capability of maintaining robustness under various distances between the camera and concrete surface. In order to train the model, we craft a new dataset, i.e., the concrete structure *spalling* and cracking (CSSC) dataset, which is released publicly to the research community. Finally, we introduce a 3D semantic mapping method using the annotated framework to reconstruct the concrete structure for visualization. We performed comparative studies and demonstrated that our AdaNet can achieve 8.41% higher detection accuracy than ResNets and VGGs. Moreover, we conducted five field tests, of which three are manual hand-held tests and two are drone-based field tests. These results indicate that our system is capable of performing metric field inspection, and can serve as an effective tool for civil engineers.**

***Index Terms*—3D reconstruction, concrete inspection, deep neural network, quadrotor flying robot, visual-inertial fusion.**

Manuscript received December 16, 2019; revised March 8, 2020; accepted April 5, 2020. This work was supported in part by the U.S. National Science Foundation (IIP-1915721), and the U.S. Department of Transportation, Office of the Assistant Secretary for Research and Technology (USDOTOST-R) (69A3551747126) through INSPIRE University Transportation Center (http://inspire-utc.mst.edu) at Missouri University of Science and Technology. Recommended by Associate Editor MengChu Zhou. *(Corresponding author: Jizhong Xiao.)*

Citation: L. Yang, B. Li, W. Li, H. Brand, B. Jiang, and J. Xiao, "Concrete defects inspection and 3D mapping using cityflyer quadrotor robot," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 4, pp. 991–1002, Jul. 2020.

L. Yang is with the CCNY Robotics Lab, Electrical Engineering Department, City College of New York, NY 10031 USA, and also with the University of Chinese Academy of Sciences, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110000, China (e-mail: lyang1@ccny.cuny.edu).

B. Li and H. Brand are with the Clemson University, SC 29607 USA (e-mail: bli4@clemson.edu; hbrand@clemson.edu).

W. Li is with the Amazon AWS AI, Seattle, Washington 98170 USA (e-mail: wli3@ccny.cuny.edu).

B. Jiang is with the CCNY Robotics Lab, Electrical Engineering Department, City College of New York, NY 10031, and also with the Hostos Community College, NY 10451 USA (e-mail: bjiang@ccny.cuny.edu).

J. Xiao is with the CCNY Robotics Lab, Electrical Engineering Department, City College of New York, NY 10031 USA (e-mail: jxiao@ccny.cuny.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/JAS.2020.1003234

I. Introduction

STRUCTURAL health monitoring (SHM) plays a significant role in performance evaluation and condition assessments for the nation's highway transportation assets. SHM can augment the operational safety and longevity of highway transportation assets based on data-driven analysis and decision-making. The Federal Highway Administration (FHWA) of the U.S. Department of Transportation (DOT) has launched a Long-Term Bridge Performance (LTBP) program in 2015 to facilitate the SHM by collecting critical performance data [1]. According to the FHWA's latest bridge element inspection manual [2], New York Bridge inspection manual [3], and tunnel operations, maintenance, inspection, and evaluation (TOMIE) manual [4], it is crucial to identify, measure, and evaluate condition state during a routine inspection on bridges and tunnels. Such condition states include concrete spall (delamination, patched area), exposed rebar, cracking, abrasion (wear), and other damages.

There are several robotic inspection systems that have been developed for automated concrete inspection. Lim *et al.* [5] proposed a visual pavement *crack* inspection and mapping system using a mobile robot platform. The robot used a camera to perform visual inspection using an edge detection algorithm with a machine learning method. Lidar was used for location tagging and mapping. Under the support of the FHWA LTBP program, Prasanna *et al.* [6], [7] proposed an autonomous bridge deck inspection mobile robotic system using a mono-visual camera, ground penetrating radar, and acoustic sensors. The robot was developed to perform pavement *crack* detection which are relatively planar surfaces. Unmanned aerial vehicles (UAVs) have also been deployed for bridge visual inspection [8]. UAVs are able to perform remote inspection for areas that are not accessible to human operators. However, none of these robotic inspection systems were able to retrieve metric information of the defects such as width, length, and area information. Also, though these robotic systems used GPS to obtain location information, they were not accurate enough to build a 3D map for visualization nor were they applicable in GPS-denied areas.

To facilitate automatic inspection, acoustic sensors [9], [10], ground penetrating radar [11], and visual cameras [12]–[14] are the three most commonly used sensors in the civil engineering community over the past decade. For visual

camera-based inspection, previous researches were mainly focused on using entropy or intensity thresholding methods by highlighting high contrast distinct visual areas. These methods include edge detection, fast Fourier transform (FFT), and fast Haar transform (FHT). Besides using pure thresholding methods, researchers also introduced new detection algorithms by combining image segmentation, image thresholding (such as OSTU's method [15]), and morphology operations [15] to produce high-quality detection results. Histogram analysis and automatic peaks detection approaches were also used for visual inspection [16]. The *crack*-defragmentation approach for fragment grouping and fragment connection was proposed in [17], and an artificial neural network (ANN) was introduced for *crack* detection classification. However, these methods only work well on a simple clear surface and are not able to indicate defect categories.

In this paper, we propose an automatic robotic system for concrete structure visual inspection, using an RGB-D camera with a deep neural network and RGB-D reconstruction method to build a 3D map with defects highlighted. This is illustrated in Fig. 1. Unlike the previous research which only performed *crack* or *spalling* detection using pure RGB images, we introduce an RGB-D visual simultaneous localization and mapping (SLAM) method for structure reconstruction and combine a deep neural network to recognize and highlight defects. The defects are registered and labeled in the 3D map to reveal the physical location in the 3D structure model, facilitating condition assessment. Furthermore, we introduce a depth adaptive windows size predictor based on depth-inpainting to effectively predict the optimized sliding window size. Then, a sliding window based multi-resolution detection model is used to detect the defect area. Finally, to visualize the defects, we introduced a conditional random field (CRF) method to perform 2D to 3D registration and fusion.

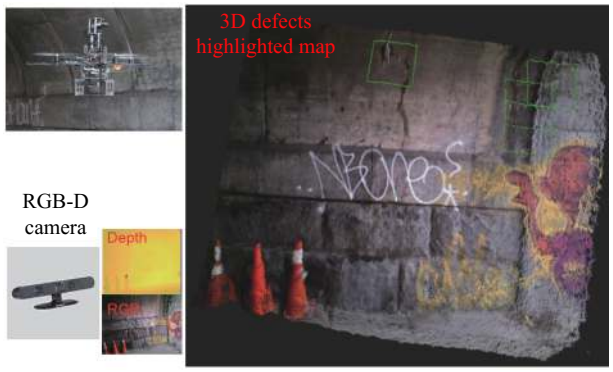


Fig. 1. Illustration of our robotic inspection system, it is developed based on the CityFlyer quadrotor robot aiming to perform the concrete structure inspection. The right side image shows the inspection result that we got under a bridge located at Riverside Drive & West 155th Street, New York. The robot is equipped with an RGB-D camera which is used for localization and 3D mapping with defect visualization.

Extending our preliminary work [18], [19], instead of using the VGGs with fixed sliding windows size to solve the detection problem, we proposed a depth adaptive model to optimize the detection. To summarize, our main contributions are:

1) A high-quality labeled dataset for *crack* and *spalling* detection, which is the first publicly available dataset for visual inspection of concrete structures. It has 522 (labeled) *crack* images and 298 *spalling* images, and over 10 000 field-collected images from the concrete structure.

2) A robotic inspection system with visual-inertial fusion to obtain pose estimation using an RGB-D camera and an IMU. The visual-inertial system has a 100 Hz pose estimation rate to enable online navigation and 3D mapping.

3) A depth in-painting model that allows depth hole in-painting in an end-to-end approach with real-time performance.

4) A multi-resolution model that adapts to image resolution changes and allows accurate defect detection in the field.

II. System Architecture

We propose a novel robot inspection system using the CityFlyer [20] which consists of a control and mission module (CMM), a visual-inertial positioning module, and a deep inspection and 3D registration module as illustrated in Fig. 2. The CMM implements autonomous navigation which is developed under the Robot Operating System (ROS) platform. The CMM receives visual inertial odometry (VIO) as feedback to navigate the CityFlyer. The VIO has a 100 Hz frame rate that meets state control requirements and also decreases the frame-to-frame pose estimation error (within 10 cm). Meanwhile, the concrete defects prediction output is registered to 3D space using the depth information and the target defect's 3D location and surface normal [21] are used to navigate the CityFlyer to the best viewing angle. By navigating the CityFlyer to the front view perspective of the target defect area, our system can achieve better inspection data acquisition.

The visual-inertial positioning module fuses the output of visual odometry and IMU propagation to achieve real-time pose estimation of the CityFlyer. We use ASUS Xtion Pro RGB-D camera as the visual perception unit to perform pose estimation and 3D perception. Its data sheet is listed in Table I. The IMU sensor is Phidgets Spatial 3/3/3 sensor. For VIO fusion, it follows the following steps. First, RGB and depth images are used to estimate the pose of the UAV, using feature matching and optimization approaches [22]. Second, we implement a multi-state extended Kalman filter (MS-EKF [23]) to fuse IMU state propagation and the visual odometry observation, allowing real-time positioning and control at a 100 Hz. It should be noted that we perform an off-line calibration to obtain the transformation, $T^{VP}_{\text{CityFlyer}}VP$, between the camera and CityFlyer body.

The adaptive defect detection and 3D registration module is proposed to solve the significant problem of providing metric information during inspection, allowing civil engineers to perform condition evaluation [4] and have a context on the spatial characteristics and location of the defects. An AdaNet with depth in-painting and multi-resolution approach is proposed to augment defect detection accuracy. We first introduced a depth-varying sliding window size optimizer. Then, the detection result is registered and fused in a 3D map

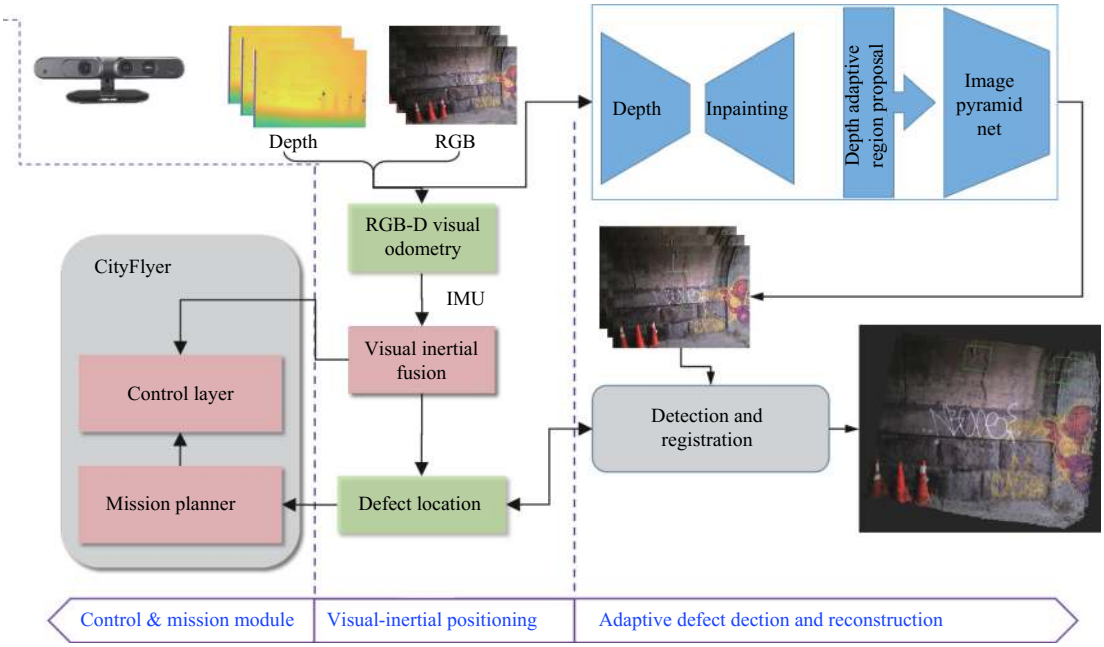


Fig. 2. The CityFlyer inspection system uses an RGB-D camera and an IMU to perform online visual positioning and navigation. We propose a F-ResNet 34 model to perform defects detection. Afterward, 3D reconstruction and registration is performed to visualize the inspection result.

TABLE I
ASUS XTION PRO DATA SHEET

Field of view	Horizontal: 58, Vertical: 45, Diagonal: 70
Depth range	0.35 → 8.0 meters (we tested)
Frame rate	30 Hz
Image resolution	VGA (640 × 480)
Depth resolution	1 (mm) (but accuracy varies with distance)

for visualization.

III. CONCRETE INSPECTION METHOD

This section discusses the DNN model-based concrete inspection method, which is able to tell the defects' 2D region information by taking RGB images as inputs. Inspired by feature pyramids [24], we propose a *Multi-resolution DetectionNet* taking multi-resolution RGB image inputs to detect the concrete defects. Moreover, we introduce a depth adaptive sliding-window size selection method, with the capability to adjust bounding box size based on the distance to the surface. In the rest of this section, we provide comprehensive theoretical analysis of the model, and we also compare the detection performance between our AdaNet and ResNets [25], VGGs [26], and AlexNet [27].

For visual inspection, we treat the concrete defects detection task as a multi-class classification problem. For all input images $X = \{x_1, x_2, ..., x_N\}$, N denotes the number of the images falling in three categories, e.g., *crack, spalling,* and *background*. Each image $x_i \in R^3$ is associated with a ground truth label $y_i \in Y$, where $Y \subset N$ is natural number starting from 0. The detection goal is to find a mapping function $f : X \to Y$ that minimizes a pre-defined loss $Loss(x, y)$. For the label Y, we encode the label of each image as an integer from $\{0, 1, ..., n-1\}$, n denotes the number of classes. In this paper, we define the *crack* images' label as 1, the *spalling* images' label as 2, and the background images' with label 0.

A. Data Preparation and Augmentation

There is no publicly accessible concrete defect dataset available to train our model, let alone an RGB-D dataset with depth information. In order to train the inspection model for defects detection, we developed a new concrete structure *spalling* and cracking (CSSC) dataset for training. We met with and organized discussions with civil engineers to catalog the terminology used in concrete defect assessment applications. This provided key terms for image-based search engines and allowed us to mine images from image search results. The following terms used for web-based datamining are listed below:

1) Concrete spalling/Rebar: Concrete *spalling*, concrete rebar, concrete delamination, concrete bridge *spalling*, concrete column *spalling*, concrete *spalling* from fire, concrete *spalling* repair, and concrete wall.

2) Concrete Crack: Concrete *crack*, *crack* repair, concrete scaling, concrete crazing, and concrete crazing texture.

We searched the image data through Google, Yahoo, Bing, and Flickr. Then, we collected a total of 954 concrete *crack* images and 278 concrete *spalling* images. For *spalling* images, we further added 20 images collected from the field, obtaining a total of 298 *spalling* images for training and validation purposes.

After assembling the *crack* and *spalling* images, we annotated them using Photoshop. An illustration of some of the annotated images are shown in Figs. 3 and 4. For *spalling* images, we annotated the exposed rebars and annotated the regions (contours) of *spalling* damage. These are two regions of interest for civil engineering diagnosis with areas of exposed rebar being areas of more serious degradation. Examples of exposed rebar and *spalling* contour annotation are shown in Fig. 4. For concrete *cracks*, the annotators were

Fig. 3. Illustration of raw *crack* images and the annotated binary images in CSSC dataset.

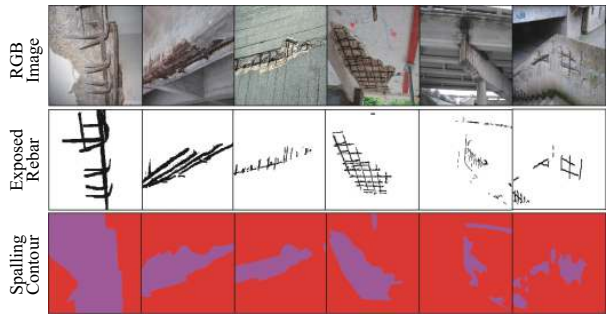


Fig. 4. Illustration of raw *spalling* images and the annotated images in CSSC dataset. We annotated the exposed rebars and *spalling* contour. In the third row, the red pixels denote the background, and the pink pixels denote the *spalling* area.

asked to carefully annotate the entire *crack* areas in order to develop a binary mask as a ground truth (shown in Fig. 3).

Since our AdaNet is a sliding-window detector, we randomly crop the images around the regions of interests (ROIs) using two size settings: 100×100 and 130×130. This is illustrated in Fig. 5. For each cropped image output, we determine whether it is a defect or background image via the rule defined in (1). We first count the number pixels, $n(I)$, located inside of the defect region in its corresponding label image (with a total $N(I)$ pixels). Then, if the defect pixel number is greater than or equal to a pre-defined threshold condition, $n(I)\geq N(I)\times k$ (where k represents an empirical percentage threshold value), we claim the cropped image as a defect image and label them with 1 (as *crack*) or 2 (as *spalling*). If there are no defected pixels, i.e., $n(I)==0$, we classify the cropped image as background and label with 0. It should be noted that a cropped image will be discarded if the number of defected pixels is between zero and the threshold, i.e., $0<n(I)<N(I)\times k$.

$$flag=\begin{cases}0, & \text{if } n(I)==0\\ discard, & \text{if } 0<n(I)<N(I)\times k\\ 1\ or\ 2, & \text{if } n(I)\geq N\times k\end{cases} \tag{1}$$

where $flag$ denotes the category of the image and *discard* denotes the image is not used for training. In this paper, we set $k=0.04$ for *crack* if cropped size is 100×100, and $k=0.06$ for *crack* if cropped size if 130×130. For concrete spalls, we set $k=0.8$ to obtain spalls sub-images. These values are selected to consider the constraints of the dataset size and the data quality for better detection accuracy.

B. Depth In-Painting Model

Commercial RGB-D cameras normally output incomplete

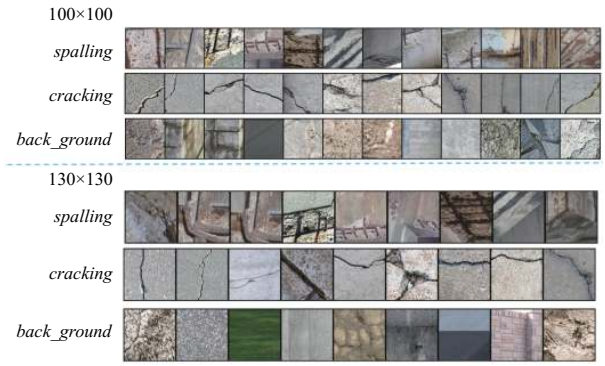


Fig. 5. Examples of generated positive training images based on the proposed selection criteria. The *spalling* images are presented with size of 100×100 with $k=0.8$, and 130×130 with $k=0.8$ for training. The *crack* images are also presented with 100×100 with $k=0.04$, and 130×130 with $k=0.06$ for training.

depth images if there is no reflected ray from certain viewing angles. The regions with missing depths in the image are referred to as "holes" [28]. Holes degrade the quality of the 3D reconstruction of a structure and the 3D metric measurements. Fig. 6 illustrates some examples of the occurence of empty regions in depth image data within a sliding window. Inspired by [29], we introduce a depth inpainting model (named **InpaintNet**) which is illustrated in Fig. 7. **InpaintNet** is developed based on U-Net [30] which has an auto-encoder framework work with five groups of down-convolutions for the encoder and five groups of up-convolutions for the decoder. Each group has two convolutional layers and each layer has the same number of channels as U-Net. **InpaintNet** is composed two U-Net frameworks connected in parallel, one of which learns a surface normal embedding from RGB images and the other one performs depth inpainting from depth and surface normal embeddings. The depth inpainting framework inputs depth images to an encoder to forms depth embedding. The depth embeddings are then concatenated with surface normal embeddings. The decoder portion of the depth inpainting networks decodes complete depth images from the combined depth and surface normal embeddings.

In this paper, we do not have the ground truth depth nor the

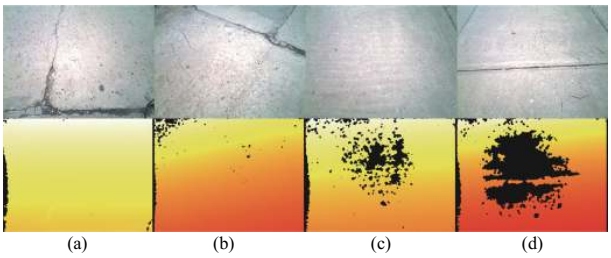


Fig. 6. Illustration of raw depth images (represented by *hot*) from RGB-D sensors, and percentage of the missing depth increases from the left to the right. It is almost impossible to obtain the depth of a box if it is located in the black area as shown in (d).

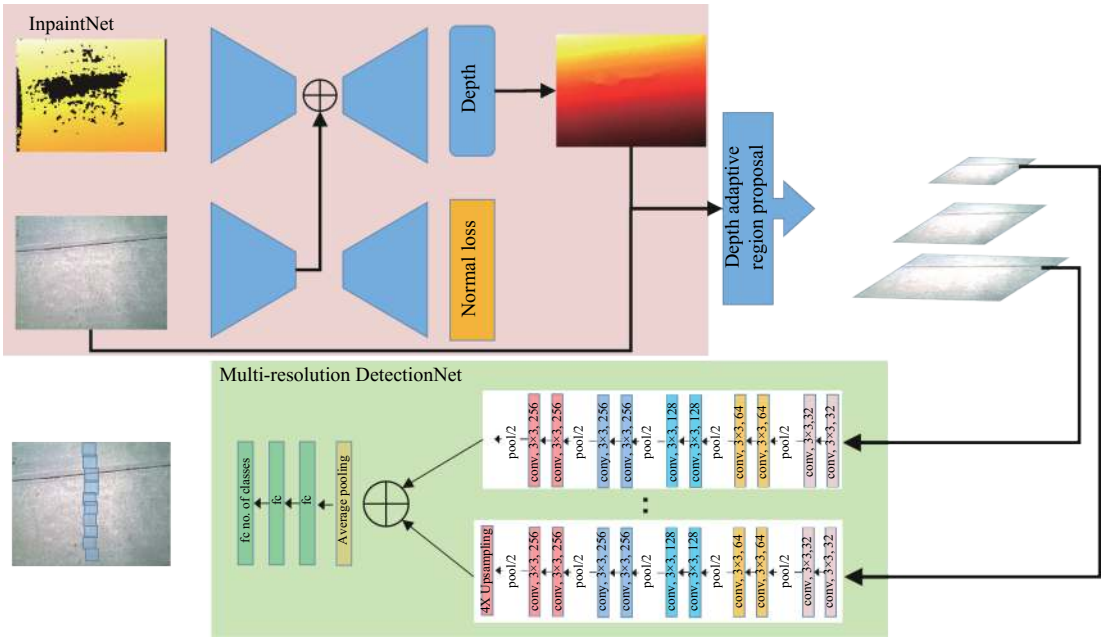


Fig. 7. AdaNet was proposed to perform depth adaptive defect detection. AdaNet uses depth information to estimate the scale of the slide-box, then performs defect detection over the target image with a multi-resolution sliding window detector. Based on our experience, we select a three resolution image pyramid for our detection task.

surface normal data for training. We therefore use classical, computationally expensive approaches to develop estimates of the complete depth and surface normal images. These approaches, though they can not be implemented in real-time, are able to generate accurate estimates of the complete depth and surface normal images to use as a ground truth for the neural network. The neural network then has the benefit of being able estimate complete depth image and surface normal images in real-time. Inspired by [31], we introduce a bilateral filter with color guiding to complete the depth images. The bilateral filtering approach is 10 times slower compared to using a neural network model. For the surface normal, we first introduce a Sobel filter to estimate the gradient of the estimate depth images in the x and y directions.

$$\Delta(x)=\begin{bmatrix}-1&0&1\\-2&0&2\\-1&0&1\end{bmatrix}\quad\Delta(y)=\begin{bmatrix}-1&-2&-1\\0&0&0\\1&2&1\end{bmatrix}\tag{2}$$

then the surface normal of each pixel is $N^I=\Delta(x)\times\Delta(y)$.

C. Multi-Resolution Detection

For robotic on-the-fly defect inspection in the field test, especially using a drone, it is quite challenging to keep a consistent distance between the camera and the surface image aquisition. Since we can obtain the depth aligned with each RGB frame, we can easily using this information to adjust the sliding window size based on the depth measurement. Thus, the detection model should be robust to images taken at any distance.

It has been discussed in our previous research [18] that a fine-tuned VGG model is not able to perform well in field tests, achieving an average of 70.05% detection accuracy due to the spatial resolution of a region depending on the inspection distance. To tackle this problem, this research further introduces a *Multi-resolution Detection* model, inspired by [24], by implementing a multi-resolution input image feature pyramid. Given a sliding window cropped input I, we resize to 1/2 and 1/4 and perform feature extraction in a parallel framework, that is

$$\begin{cases}X^1=CNN(wI+b)\\X^{\frac{1}{2}}=CNN(wI^{\frac{1}{2}}+b)\\X^{\frac{1}{4}}=CNN(wI^{\frac{1}{4}}+b)\end{cases}\tag{3}$$

where $CNN(wI^{\frac{1}{i}}+b), i=1,2,4$ denotes the CNN feature encoder, w and b are symbolic representations of the convolution kernel and bias respectively. $I^{1,2,4}$ denotes the input image where the superscript denotes the corresponding scale. X represents the corresponding output to $I^{1,2,4}$. Because all levels of the pyramid use the same network architecture, the output also differs with the size. We further up-sample the size of the coarse output feature for 1/2 and 1/4 sized images with a factor of 2 and 4, respectively. In this paper, we take the raw sliding window input and resize to 224×224, that is, $size(I)=224\times224$.

To reduce the channel dimension after concatenation, we introduce a convolutional layer with kernel size 1×1 to reduce the channel dimension to 256, then apply an average pooling operation

$$f^i=\frac{\sum(C^i)}{size(C^i)}\tag{4}$$

where C^i is channel $i\in\{0,1,...,255\}$ and f^i is the output after average pooling. A three-layered fully connected convolution is used to regress and predict whether the current region is a defected area or not.

D. Loss Design and Training

AdaNet is designed to perform concrete defect detection and classification with distance adaptable capability, and is trained through a joint approach for finding the optimal weight to regress to an expected prediction. For **InpaintNet**, it is a per pixel value prediction model where we evaluate the model performance using a *perpixel* photometric loss as

$$Loss = \arg\min(L1(N^I, N_G^I) + L1(D^I, D_G^I)) \tag{5}$$

where N^I is the predicted normal, N_G^I is the normal ground truth, D^I is the predicted depth, D_G^I is the depth ground truth. We jointly optimize over both loss terms.

For the multi-resolution detection model, aims at determining the existence of defects in an image frame. $f()$ predicts the probability of the class y_i given x_i as an input. Thus, the loss can be simplified to a cross-entropy style.

$$Loss(x, y) = -\sum_{i=0}^{2} y_i \log(f(x_i, W^{\text{AdaNet}})) \tag{6}$$

where x_i is the ith input patch, W^{AdaNet} denotes the convolutional kernel, and $y_i \in \{0, 1, 2\}$ is the label of each class. In this paper, we use the above cross entropy loss to perform detection regression for our AdaNet model.

Training: The training dataset for multi-resolution DetectionNet contains three classes, as discussed in Section III-A, and we annotate the labels as $0: back\_ground$, $1: crack$, $2: spalling$, respectively. Besides initializing the model with pre-trained model parameters, we also augment the dataset using 1) random rotation with several pre-defined angles; 2) gamma correction ranging from 0.5 to 2.0. For training, we split all the images into three sub-datasets: the training dataset (75% of all the images), validation dataset (15% of all the images), and testing dataset (10% of all the images). For **InpaintNet**, we used all 10 000 field collected data. The ground truth depth and normal are obtained through the method proposed in Section III-B.

IV. Pose Estimation And 3D Semantic Registration

Our final goal is to reveal where the defects are in a 3D map by registering concrete defects to the 3D map. In this section, we discuss using an RGB-D camera to perform 3D positioning and semantic 3D reconstruction based on the conditional random field (CRF) method to highlight the concrete defects in the map.

A. Visual Positioning and Association

The 3D model of the concrete structure is widely used for structure analysis in civil engineering. Moreover, metric defects could be registered to the 3D model with color coded overlays. This provides further assistance to civil engineers to perform concrete structure condition comprehensive assessments [32]. In this paper, we propose a 3D mapping system taking advantage of visual-inertial (VI) SLAM and deep defect detection.

As discussed in Section II, the CityFlyer requires high localization accuracy and update frequency to enable stable navigation. In this paper, we introduce an MS-EKF [33] to fuse high-frequency IMU propagation and low-frequency visual odometry (VO) towards real-time pose estimation. For VI fusion, the IMU measurement is used to predict the state transition and VO observations were used to update the state. The difference in measurement frequency allows us to accommodate the fusion of multiple sensors. For the IMU, its evolving state vector is

$$X_{\text{IMU}} = \begin{bmatrix} {}^W P_I^T & {}^W V_I^T & {}^W a_I^T & {}_W^I q & b_a & b_g \end{bmatrix} \tag{7}$$

where ${}^W P_I^T$ denotes the position of the IMU in the world frame W. ${}_W^I q$ is the unit quaternion that represents the rotation from the world frame W to the IMU frame I. ${}^W V_I^T$ and ${}^W a_I^T$ are the IMU linear velocity and linear acceleration with respect to the world coordinate system. b_a and b_g denote the biases affecting the accelerometer and gyroscope measurements. The system derivative form can be partially represented as following in an east-north-up (ENU) coordinate system (partly referred in [33], [34]):

$$\begin{aligned} {}^W \dot{P}_I &= {}^W V_I + {}^W a_I \times \Delta T \\ {}^W \dot{V}_I &= {}^W a_I \times \Delta T = {}_I^W C(a_m - b_a) + g \\ \dot{a}_I &= j + \chi \times V + w_m \times a_m \\ {}_W^I \dot{q} &= \tfrac{1}{2} \Omega(w_m) {}_W^I \dot{q} \end{aligned} \tag{8}$$

where ${}_I^W C$ is the translation from the IMU frame to the world frame, a_m is the acceleration measurement, w_m is the angular velocity measurement, ΔT denotes the time interval, and g denotes the gravity. The acceleration, a_I, is subject to rotation and translation in the IMU frame. w_I denotes the angular velocity and Ω is the matrix product referred to in [33].

Meanwhile, the VO performs pose estimation using the RGB-D measurement, and outputs the pose $P_{vo} = [r, t]$ (where r denotes the rotation, t denotes the translation). Once VO finished pose estimation for each frame, we can update the state based on the measurement model.

$$P_{\text{vo}} = H X_{\text{IMU}} + V \tag{9}$$

V denotes the measurement noise. H denotes the measurement matrix which represents the mapping between IMU state and the VO pose. Then, the prediction from IMU propagation can be corrected by updating using the EKF filter, achieving a 100 Hz pose estimation rate.

The state estimation error of the VIO will continue to drift as there is no loop-closure to correct the pose if there exists an overlap between views (observations). To further correct the pose, we record the key-frames ${}^F K = \{{}^k I_i, {}^k P_i | i \in (1, 2, ..., m)\}$ (i.e., vertex) based on a motion threshold, where ${}^k I_i$ and ${}^k P_i$ denote the key-frame image and the key-frame pose of a frame i, respectively. VIO propagation and update allow us to obtain the transformation between two consecutive frames i, j, and the relative transformation $T_{i,j}$ can also be derived at the same time. In order to reduce the drift of the visual odometry, this paper introduces graph-optimization to correct the pose drift based on [35]. To perform graph optimization, the following procedures have to be followed: 1) record the key-frames, ${}^F K$, based on motion threshold method; 2) use image features to facilitate loop-closure detection to find the edges (correlation) between any pair of key-frames; 3) perform graph optimization to update all poses simultaneously.

$$({}^{k}\tilde{P}_i, {}^{k}\tilde{P}_j) = \arg\min_{{}^{k}P_i} ({}^{k}\hat{P}_j - T_{i,j} \times {}^{k}\hat{P}_i)^T \times \Omega_{i,j} \times ({}^{k}\hat{P}_j - T_{i,j} \times {}^{k}\hat{P}_i) \quad (10)$$

where $\Omega_{i,j}$ denotes the information matrix that describes the correlation between parameters and ${}^{k}\tilde{P}_{i,j}$ denotes the optimized poses. Equation (10) is able to update all frame's new poses at the same time. Here we just use i, j as an example. Once the graph optimization is done, we take the pose error of the last key frame, ${}^{K}T_{\text{error}} = {}^{K}\tilde{P}_i \times {}^{K}\hat{P}_i^{-1}$, to correct the current the VIO propagation. Then, we correct the current VIO output using the correction, $\hat{P}_{\text{current}} = {}^{K}T_{\text{error}} \times P_{\text{current}}$, where P_{current} is the VIO output.

B. Spalling and Cracking Fusion Using CRF

After VIO pose estimation, we have the pose $p_I = r, t$ of each RGB-D frame, and the corresponding region detection results R_I of each image frame I. Each depth frame ${}^{d}I$, has *millimeter* accuracy. In this paper, we aim to perform a metric reconstruction and superimpose the defect class on the 3D map for better visualization. For each RGB-D frame, we can perform a backward-projection to register the current view measurement to the 3D world.

$$[X, Y, Z] = [r, t]^{-1} \mathcal{K} [u, v] \quad (11)$$

where $[u, v]$ denotes the pixel coordinate in the image, $[X, Y, Z]$ is the corresponding 3D position in world coordinate system, $\mathcal{K}$ is the inverse camera intrinsic parameter, and $[r, t]^{-1}$ denotes the transformation from the camera coordinate system to world coordinate system. The output of the multi-resolution DetectionNet is defect region information, allowing the defected regions in the 3D model to be labeled with specific colors. In this paper, the defects detection in an image is performed using a sliding window approach. Each sliding window defines a region bounding box $U = (u_{\min}, u_{\max}, v_{\min}, v_{\max})$ where the network can output the corresponding class probability distribution $P_C = \{P_{c_i} | C_i, i = 1, ..., n\}$ on n classes. One very important hypothesis we claim is that we assume each pixel in a defect region should have the same probabilistic distribution P_C, i.e., $P_{(u_l, v_l)} = P_c$ for each pixel $(u_l, v_l) \in U$.

In order to fuse a sequence of inspection results, we introduce conditional random fields (CRF) to perform spatial fusion based on our previous work [36]. For each image frame I, the prediction $P_{(u_k, v_k)}$ on region R_I is performed via AdaNet, where (u_k, v_k) is the image coordinate. The fusion involves the following procedures: 1) we build a voxel map and each voxel $\Gamma(i)$ is initialize with equal label probability, i.e., $P_{\Gamma(i)} = \{c_i = 1/n, i = 1, ..., n\}$; 2) each new RGB-D frame will have a new probabilistic image using the detection model, and we perform fusion using CRF [37], [38] to fuse the label probabilistic distribution.

For each pixel (u_k, v_k), we first perform a warping operation to find the association between the voxel map and current pixel, and check whether the corresponding voxel is initialized or not. If not, we first initialize it with an equal distribution, $P_{\Gamma(i)} = \{c_i = 1/n, i = 1, ..., n\}$. Then, with the next frame overlapping the region, we perform a warping via deployment of a general homogeneous transformation to get the voxel index in the voxel map.

$$\Gamma(j) = \pi((u_k, v_k), D(u_k, v_k), T_I) \quad (12)$$

$D(u_k, v_k)$ is the depth measurement of pixel (u_k, v_k), $\Gamma(j)$ is the corresponding voxel in the world, T_I is the transformation from world coordinate frame to the current view, and π is the warping operator that maps the current view to the world coordinate system. With the AdaNet output the class probability prediction $\mathcal{A}(u_k, v_k)$ of pixel (u_k, v_k), we have the conditional probability distribution, $P(u_k, v_k) = \{P(\mathcal{A}(u_k, v_k) == c_i), i = 0, 1, 2\}$. Then, we can update the global probabilistic distribution of each voxel following a recursive Bayesian update procedure [38]:

$$\begin{aligned} &P(\Gamma(j)_{k+1} | \mathcal{A}(u_k, v_k)_{k+1}, \Gamma(j)_k) \\ &= P(\Gamma(j)_k) P(\mathcal{A}(u_k, v_k)_{k+1}) \\ &= P(\Gamma(j)_k) P(u_k, v_k) \end{aligned} \quad (13)$$

where $P(\Gamma(j)_{k+1} | \mathcal{A}(u_k, v_k)_{k+1}, \Gamma(j)_k)$ denotes the probabilistic prediction of voxel $\Gamma(j)$ at time $k+1$ using AdaNet $\mathcal{A}(u_k, v_k)_{k+1}$, and then update its probabilistic distribution. $P(\Gamma(j)_k)$ denotes the probabilistic distribution at time k. Because the prediction between each frame is independent, the update becomes a simple dot operation between each class. The posterior update is performed over all visible voxels, and is finally normalized to obtain $P(\Gamma(j)_{k+1})$.

V. Experiments

In this section, we discuss the AdaNet training details and compare the experimental performance of the depth in-painting model and defect detection model. To verify the effectiveness of our system, we perform several field tests in a manual holding mode for the RGB-D camera and autonomous inspection mode using the CityFlyer.

A. Depth In-Painting Analysis

We first perform an ablation study on the depth in-painting performance from an accuracy and time performance perspective. Table II shows the results of InpaintNet compared to the raw output and in-painted result from a bilateral filter [31]. We performed four tests with each dataset containing RGB-D frames from planar concrete surfaces. The ground truth was manually obtained by measuring the distance of the camera to the surface plane. In Table II, the depth images of *Cracks 1* and *3* have large holes which are not removable through a bilateral filter. **InpaintNet**, however, is able to achieve a more accurate and complete depth in-

TABLE II
DEPTH ACCURACY COMPARISON TO RAW IMAGE AND [31] (MEAN ABSOLUTE ERROR (MAE) (mm))

Item	*Crack 1*	*Crack 2*	*Crack 3*	*Crack 4*
Raw	278.1070	25.0388	100.2492	16.1562
Bilateral [31]	60.4663	2.8168	39.2606	1.1824
InpaintNet	**17.4246**	**3.3215**	**27.9362**	**2.3362**

[1] https://github.com/ccny-ros-pkg/pytorch_Concrete_Inspection

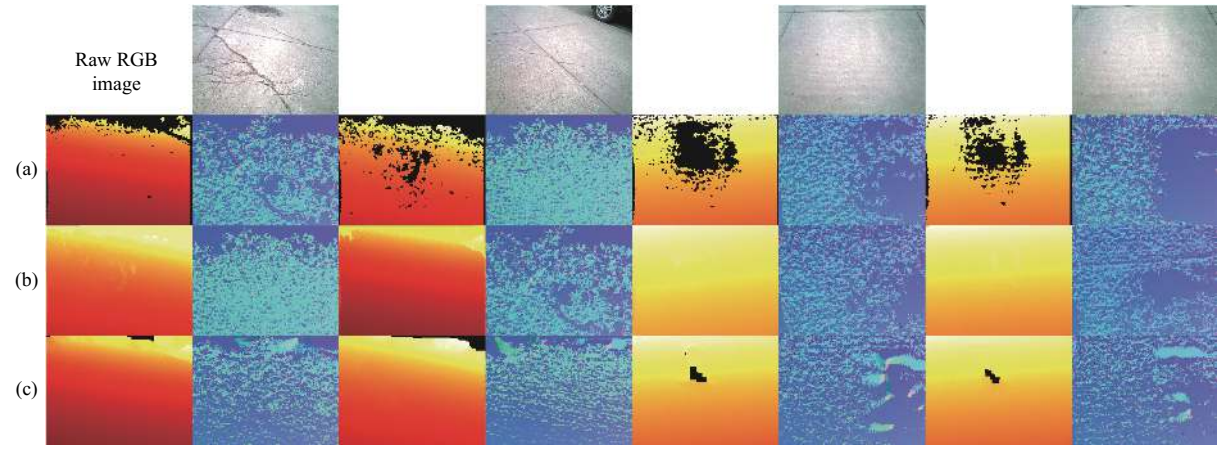


Fig. 8. Result of using bilateral filter and InpainNet to perform depth in-painting. (a) denotes the raw depth and the corresponding normal; (b) is the in-painted depth and normal using InpaintNet; (c) is the in-painted depth and normal using bilateral filter.

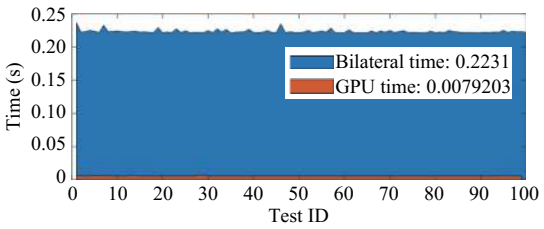


Fig. 9. Time performance comparison between bilateral filter and InpaintNet, where InpaintNet takes an average 0.008 seconds to in-paint per depth frame.

painting for *Cracks 1* and *3*. For the depth images of *Cracks 2* and *4*, have small holes and can be easily filtered through a bilateral filter.

A graphic comparison is given in Fig. 8, where we can see that **InpaintNet** is able to fill the big holes, even though it may not able to give precise prediction. Also, compared with bilateral filter, **InpaintNet** could resolve a smoother normal estimation. The time performance between the two algorithms were compared revealing **InpaintNet** to be 30 times faster compared with the bilateral approach (as illustrated in Fig. 9). The runtime of **InpaintNet** was 0.008 seconds on average with a GTX 1080 GPU for each depth frame.

B. Detection Model Comparative Analysis

As discussed in Section III-A, we cropped images to obtain training patches, and we made the cropped dataset[1] publicly available for the research community. The dataset has a total of 26 870 concrete *crack* image patches, 15 950 concrete *spalling* image patches, and 46 429 *back_ground* image patches. We label back ground as 0, concrete *crack* as 1, and concrete *spalling* as 2. Representative cropped images are presented in Fig. 5. All of the network training and testing are carried out on a GPU server with GTX 1080 GPU and implementated using Pytorch.

1) Does Multi-Resolution Help? We conducted various comparative experiments between our multi-resolution detection model and other models, especially F-VGG employed in [19]. Besides VGGs, we also made comparisons to current state-of-art models including ResNets [25] and AlexNet [27]. From the comparative results presented in Table III, we can conclude that our multi-resolution model does not achieve the highest learning accuracy, but does obtain the highest testing accuracy. We also conducted a comparative study to the model used in [19] and listed results in Table IV.

Inspection of the results in Table IV reveal that our multi-resolution model is able to achieve higher detection accuracy, with an average 8.405% higher detection accuracy. This is also illustrated in Fig. 10 where it shows that the multi-resolution model outputs better coverage predictions than that of F-ResNet-34.

2) Does Deeper Model Has Better Performance? Research has shown that increasing the depth of a neural network can improve the classification accuracy to a certain extent [26]. However, the model degradation problem occurs if the model is deeper than a suitable limit. Then, authors in [25] introduced a deep residual network to overcome the degradation problem, allowing the performance of networks to increase to a higher degree with deeper layer architectures. In this section, we focus on using a well-constructed model with a suitable depth and perform fine-tuning. We do not discuss the degradation problem.

Since our task is to classify three classes, the texture difference between *crack* and *spalling* are quite distinct. However, some possible challenges are the illumination variations and an insufficient dataset. We perform comparative testing on our multi-resolution model, F-ResNets [25], F-VGGs [26], and AlexNet [27]. For the comparison, we set the batch size, epoch, learning rate, and loss as the same for a fair comparison. The result is illustrated in Table III. From the table it is clear that the deeper a model, the higher the accuracy it can achieve. Table III shows that the highest accuracy 96.88% was achieved by F-ResNet-101. Another interesting finding is that F-ResNets have an average of 1.0% higher accuracy in performance compared to F-VGGs. However, deeper models cannot achieve the best detection performance if the best input cropping practice is not used.

3) Batch Normalization: In this paper, we also discuss the effect of batch normalization for neural network models.

TABLE III
ACCURACY COMPARISON BETWEEN F-RESNETS, F-VGGS, AND ALEXNET

Model	BathSize	Epoch	Learning rate	Loss	L-accuracy	T-accuracy	Loss
Ours	32	10	0.01	Cross entropy	0.9953	**0.9732**	0.1104
AlexNet	32	10	0.01	Cross entropy	0.9717	0.9543	0.1274
ResNet-18	32	10	0.01	Cross entropy	0.9967	0.9631	0.1405
ResNet-34	32	10	0.01	Cross entropy	**0.9977**	0.9639	0.1325
ResNet-50	32	10	0.01	Cross entropy	0.9931	0.9684	0.1205
ResNet-101	32	10	0.01	Cross entropy	0.9938	0.9688	0.117
VGG-11	32	10	0.01	Cross entropy	0.9773	0.9556	0.125
VGG-13	32	10	0.01	Cross entropy	0.9859	0.9589	0.1177
VGG-16	32	10	0.01	Cross entropy	0.9791	0.9595	0.1195
VGG-19	32	10	0.01	Cross entropy	0.9746	0.963	**0.1075**

TABLE IV
FIELD TEST DATA DETECTION COMPARISON

Method	Test No.	Average precision (%)	Blurred image (frames)	Average precision without blur (%)	Over estimated (%)	Total image
F-VGG	No.1	72.45	149	76.73	97.18	4998
F-VGG	No.2	67.65	55	71.19	24.3	2650
Ours	No.1	81.32	149	84.33	55.72	4998
Ours	No.2	75.59	55	79.03	16.5	2650
Further tuned with field data						
F-VGG	No. 1	83.69	149	87.97	93.34	4998
F-VGG	No. 2	81.38	55	84.92	33.57	2650
Ours	No. 1	88.72	149	91.7	19.45	4998
Ours	No. 2	85.12	55	88.66	13.3	2650

TABLE V
COMPARATIVE RESULTS OF USING BATCH NORMALIZATION (BN)

Model	Learning accuracy		Training accuracy	
	None	BN	None	BN
VGG-11	0.9773	0.9936	0.9556	0.9645
VGG-13	0.9859	0.9947	0.9589	0.9662
VGG-16	0.9791	0.9954	0.9595	0.9645
VGG-19	0.9746	0.9942	0.963	0.9660

Batch normalization is proposed to solve the **internal covariate shift** issue and can work on each neuron to allow scale normalization during training. This enables the model to converge even given larger learning rates and also removes the need for dropout. In this paper, we compare the performance of F-VGGs between given batch normalization and no batch normalization. The results are illustrated in Table V. The results in Table V reveals that batch normalization can improve the accuracy by 0.65% on average. This also proves that batch normalization can improve the model performance even with less diversity in the data. A quantitative comparison of detection accuracy illustrated in Tables III and IV, shows that VGG-Nets are not able to achieve comparable detection performance compared to our multi-resolution detection model.

C. Field Tests and Comparisons

We conducted field tests at 155 St Broadway, Upper Manhattan, on a concrete bridge. We performed the inspection under the bridge using an RGB-D camera mounted CityFlyer. The CityFlyer was also mounted with a MasterMind computer to perform on-board computation and image streaming to the ground station (a GPU computer for defect detection). Besides the field tests via the CityFlyer, we also manually scanned the concrete surface with the RGB-D camera.

1) Field Tests (Manual Field Test): In the first stage, we manually carried the RGB-D camera to scan the concrete surface and collect the RGB-D frames for inspection. It should be noted that we have to launch the VIO system to track the motion of the camera to perform a reconstruction of the target concrete surface.

We collected three sets of data for three different scenarios, which each RGB-D frame having a location tag. Then, we performed defect inspection using our deep inspector over each image. The results are illustrated in Fig. 11, where green rectangles denote *spalling* and cyan rectangles denote *cracks*. To perform detection, we deployed a sliding window to scan through the whole image with varying region sizes from 80×80 to 200×200.

We can see in the left-most image and the center image of Fig. 11 that our model is able to recognize the *spalling* region and *crack* region. Further demonstration of the performance of the model is shown in the center image where the *spalling* region is distinguished from the *crack* region. These results show how our model can cover the whole defect area in consecutive frames and how this method is able to help civil

Fig. 10. Two automative field tests, *Test* 1 and *Test* 2, are carried out using CityFlyer. In both tests, we illustrate the drone's trajectory, region detection results, and the 3D reconstructed mapping with defect highlighting.

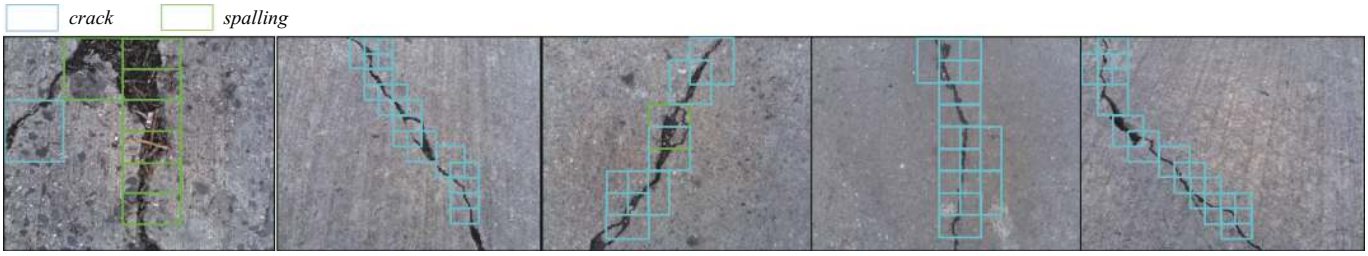


Fig. 11. We performed 3 field tests in a manual held RGB-D camera mode. The green rectangles indicate the spalling region, and cyan rectangles denote the crack region.

engineers be aware of the condition of the concrete structure.

2) Autonomous Field Test Using CityFlyer: We also performed two sets of field tests using our CityFlyer, and the results are illustrated in Fig. 10. In Fig. 10, *Test* 1 is carried out at the entrance of the area under a bridge, and *Test* 2 is carried out at the middle of the area under the bridge where the illumination is low.

For *Test* 1, the trajectory of the drone is illustrated in the left-most image, this illustrates how the CityFlyer was maneuvering to capture the target area. The defect inspection result is illustrated in the second to the left image, where cyan and green rectangles denote *crack* and *spalling*, respectively. The right-most image is the front view of the 3D map (point cloud) with color overlayed on the defects, and the second to the right image shows the same point cloud but with a different view from the back. We can see that the *spalling* and *cracks* are well highlighted.

The second test was carried under the bridge, which suffers from low illumination for inspection and localization. The trajectory of the drone is given in the left-most image of *Test* 2, and the inspection result at this location is given in the second to the left image and the second to the right image. The second to the left image indicates that our model is able to perform correct *spalling* detection even in a low-illumination environment. However, the second to the right image indicates that it missed detection of a *crack* region (indicated with a red dashed rectangle) due to low illumination.

3) Semantic 3D Fusion and Visualization: The semantic 3D highlighted results are illustrated in Fig. 10, where we performed back-projection using the predicted output and the corresponding depth image to the 3D world coordinate frame and the 3D spatial data is fused using consecutive frames. We use a voxel map to represent 3D structure information, where each voxel has to be updated through a back-projection manner. Since we deploy an image-based fusion approach, a global probabilistic map searching is not required, enabling non-GPU computation. The reconstructed 3D map with semantic highlighted areas is illustrated in the right-most images of Fig. 10. It can be seen in the figure that the regions of defect are well highlighted using green and cyan color. This helps civil engineers identify the defect categories as well as their location.

VI. Conclusion

In this paper, we introduced a new automatic concrete structure inspection system using the CityFlyer robot mounted with an RGB-D camera toward visual inspection. For visual concrete inspection, we introduced an AdaNet to perform a detection of defects within a sliding window approach. The AdaNet consists of two sub-models, which are, a depth inpainting model (**InpaintNet**) to fill holes in a depth image and multi-resolution defect detection model for concrete inspection. The depth adaptive multi-resolution detection model considers both distance and resolution effects, aiming to provide a robust concrete *crack* and *spalling* detection task in the field. Meanwhile, we pioneeringly propose using visual SLAM and deep neural network inspection to perform a 3D semantic reconstruction to highlight the defects in a 3D model. It can achieve an average 8.41% higher detection accuracy compared to F-VGG and F-ResNets. Furthermore, we introduce an RGB-D visual-inertial fusion with filtering and global bundle adjustment to perform pose estimation for the CityFlyer state control. The pose information is used to provide location tags defects predicted in images. Comparative experiments and field tests indicate that the system is able to perform high-quality detection and reconstruction. For future work, we will try optimal tuning of super parameters of the proposed models via intelligent optimization methods [39] and also work on pixel-level detection toward metric reconstruction.

Acknowledgment

The authors acknowledge the sponsorship support from NSF NRT: Resilient Infrastructure and Environmental Systems (RIES) Program at Clemson University. The views, opinions, findings and conclusions reflected in this publication are solely those of the authors and do not represent the official policy or position of the NSF, USDOTOST-R, or any State or other entity. J. Xiao has significant financial interest in InnovBot LLC, a company involved in R&D and commercialization of the technology.

References

[1] N. Gucunski and H. Parvardeh, "Condition assessment of bridge deck using various nondestructive evaluation (NDE) technologies," Center of Advanced Infrastructure and Transportation, Rutgers Univ., USA, Jun. 2015.

[2] U.S. Department of Transportation Federal Highway Administration, "Specification for the national bridge inventory bridge elements," U.S. Department of Transportation Federal Highway Administration, USA, 2014.

[3] N. Y. D. of Transportation, "Bridge inspection manual," Jan. 2016. [Online]. Available: https://www.dot.ny.gov/divisions/engineering/structures/manuals/bridge-inspection

[4] B. William and E. Steve, "Tunnel operations, maintenance, inspection, and evaluation (TOMIE) manual," Federal Highway Administration, Washington, DC, USA, Jul. 2015.

[5] R. S. Lim, H. M. La, and W. H. Sheng, "A robotic *crack* inspection and mapping system for bridge deck maintenance," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 2, pp. 367–378, Jan. 2014.

[6] P. Prasanna, K. J. Dana, N. Gucunski, B. B. Basily, H. M. La, R. S. Lim, and H. Parvardeh, "Automated *crack* detection on concrete bridges," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 591–599, Oct. 2016.

[7] H. M. La, N. Gucunski, K. Dana, and S. H. Kee, "Development of an autonomous bridge deck inspection robotic system," *J. Field Rob.*, vol. 34, no. 8, pp. 1489–1504, Dec. 2017.

[8] N. Hallermann and G. Morgenthal, "Visual inspection strategies for large bridges using unmanned aerial vehicles (UAV)," in *Proc. 7th IABMAS, Int. Conf. Bridge Maintenance, Safety and Management*, Washington, DC, USA, 2014, pp. 661–667.

[9] Z. Ren, K. Qian, Z. X. Zhang, V. Pandit, A. Baird, and B. Schuller, "Deep scalogram representations for acoustic scene classification," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 3, pp. 662–669, May 2018.

[10] D. Yu and J. Y. Li, "Recent progresses in deep learning based acoustic models," *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 3, pp. 396–409, Jul. 2017.

[11] Z. W. Wang, M. C. Zhou, G. G. Slabaugh, J. F. Zhai, and T. Fang, "Automatic detection of bridge deck condition from ground penetrating radar images," *IEEE Trans. Autom. Sci. Eng.*, vol. 8, no. 3, pp. 633–640, Dec. 2011.

[12] G. Li, S. H. He, Y. F. Ju, and K. Du, "Long-distance precision inspection method for bridge *cracks* with image processing," *Autom. Constr.*, vol. 41, pp. 83–95, May 2014.

[13] R. S. Adhikari, O. Moselhi, and A. Bagchi, "Image-based retrieval of concrete *crack* properties for bridge inspection," *Autom. Constr.*, vol. 39, pp. 180–194, Apr. 2014.

[14] M. R. Jahanshahi and S. F. Masri, "Adaptive vision-based *crack* detection using 3D scene reconstruction for condition assessment of structures," *Autom. Constr.*, vol. 22, pp. 567–576, Mar. 2012.

[15] S. K. Sinha and P. W. Fieguth, "Automated detection of *cracks* in buried concrete pipe images," *Autom. Constr.*, vol. 15, no. 1, pp. 58–72, Jan. 2006.

[16] T. H. Dinh, Q. P. Ha, and H. La, "Computer vision-based method for concrete *crack* detection," in *Proc. 14th Int. Conf. Control, Automation, Robotics and Vision*, Phuket, Thailand, 2016, pp. 1–6.

[17] L. L. Wu, S. Mokhtari, A. Nazef, B. Nam, and H. B. Yun, "Improvement of *crack*-detection accuracy using a novel *crack* defragmentation technique in image-based road assessment," *J. Comput. Civ. Eng.*, vol. 30, no. 1, pp. 04014118, Jan. 2016.

[18] L. Yang, B. Li, W. Li, Z. M. Liu, G. Y. Yang, and J. Z. Xiao, "A robotic system towards concrete structure *spalling* and *crack* database," in *Proc. IEEE Int. Conf. Robotics and Biomimetics*, Macau, China, 2017, pp. 1276–1281.

[19] L. Yang, B. Li, W. Li, Z. M. Liu, G. Y. Yang, and J. Z. Xiao, "Deep concrete inspection using unmanned aerial vehicle towards CSSC database," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, Vancouver, Canada, 2017, pp. 24–27.

[20] I. Dryanovski, R. G. Valenti, and J. Z. Xiao, "An open-source navigation system for micro aerial vehicles," *Auton. Rob.*, vol. 34, no. 3, pp. 177–188, Mar. 2013.

[21] R. G. Valenti, Y. D. Jian, K. Ni, and J. Z. Xiao, "An autonomous flyer photographer," in *Proc. IEEE Int. Conf. Cyber Technology in Automation, Control, and Intelligent Systems*, Chengdu, China, 2016, pp. 273–278.

[22] R. Mur-Artal and J. D. Tardòs, "ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Rob.*, vol. 33, no. 5, pp. 1255–1262, Jun. 2017.

[23] H. Z. Fang, N. Tian, Y. B. Wang, M. C. Zhou, and M. A. Haile, "Nonlinear Bayesian estimation: From Kalman filtering to a broader horizon," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 2, pp. 401–417, Feb. 2018.

[24] T. Y. Lin, P. Dollár, R. Girshick, K. M. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Honolulu, HI, USA, 2017, pp. 936–944.

[25] K. M. He, X. Y. Zhang, S. Q. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, 2016, pp. 770–778.

[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:* 1409.1556, Sept. 2014.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 25th Int. Conf. Neural Information Processing Systems*, Lake Tahoe, USA, 2012, pp. 1097–1105.

[28] H. Y. Xue, S. M. Zhang, and D. Cai, "Depth image inpainting: Improving low rank matrix completion with low gradient regularization," *IEEE Trans. Image Process.*, vol. 26, no. 9, pp. 4311–4320, Sept. 2017.

[29] Y. D. Zhang and T. Funkhouser, "Deep depth completion of a single RGB-D image," in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, pp. 175–185.

[30] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. 18th Int. Conf. Medical Image Computing and Computer-Assisted Intervention*, Munich, Germany, 2015, pp. 234–241.

[31] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," in *Proc. ACM SIGGRAPH*, California, USA, 2004, pp. 689–694.

[32] W. Q. Liu and S. E. Chen, "Reliability analysis of bridge evaluations based on 3D light detection and ranging data," *Struct. Control Health Monit.*, vol. 20, no. 12, pp. 1397–1409, Dec. 2013.

[33] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proc. IEEE Int. Conf. Robotics and Automation*, Roma, Italy, 2007, pp. 3565–3572.

[34] L. Armesto, J. Tornero, and M. Vincze, "Fast ego-motion estimation with multi-rate fusion of inertial and vision," *Int. J. Rob. Res.*, vol. 26, no. 6, pp. 577–589, Jun. 2007.

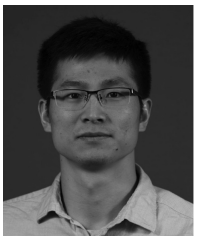
[35] R. ümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G.2o: A general framework for graph optimization," in *Proc. IEEE Int. Conf. Robotics and Automation*, Shanghai, China, 2011, pp. 3607–3613.

[36] L. Yang, B. Li, W. Li, B. Jiang, and J. Z. Xiao, "Semantic metric 3D reconstruction for concrete inspection," in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition Workshops*, Salt Lake City, UT, USA, 2018.

[37] B. Limketkai, D. Fox, and L. Liao, "CRF-filters: Discriminative particle filters for sequential state estimation," in *Proc. IEEE Int. Conf. Robotics and Automation*, Roma, Italy, 2007, pp. 3142–3147.

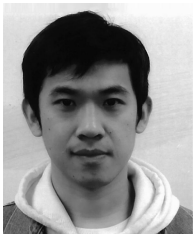
[38] A. Kundu, Y. Li, F. Dellaert, F. X. Li, and J. M. Rehg, "Joint semantic segmentation and 3D reconstruction from monocular video," in *Proc. 13th European Conf. Computer Vision*, Zurich, Switzerland, 2014, pp. 703–718.

[39] S. C. Gao, M. C. Zhou, Y. R. Wang, J. J. Cheng, H. Yachi, and J. H. Wang, "Dendritic neuron model with effective learning algorithms for classification, approximation, and prediction," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 30, no. 2, pp. 601–614, Feb. 2019.



Liang Yang received the B.S. degree from Shenyang Aerospace University in 2012, and the Ph.D. degree in electronics engineering from the City College of New York (CCNY/CUNY City College), USA, in 2019, and the Ph.D. degree in pattern recognition and intelligent system from University of Chinese Academy of Sciences in 2019. He joined APPLE as a Senior 3D computer vision Researcher in 2019, and currently working on 3D visual perception and understanding. His research interests include motion and path planning, 3D perception and understanding, visual SLAM and reconstruction, multi-sensor fusion, and robotic control.

Bing Li (M'17) received the B.S. and M.S. degrees from Beijing Forestry University and Beihang University, in 2006 and 2009, respectively, and the Ph.D. degree in electrical engineering from the City College of New York (CCNY/CUNY City College), USA, in 2018. He worked for the China Academy of Telecommunication Technology from 2009 to 2012, and for IBM and HERE North America LLC during the Ph.D. education. He joined the Department of Automotive Engineering in 2018 as an Assistant Professor in the Clemson University International Center for Automotive Research (CU-ICAR), USA. His current research interests include sensing and perception, computer vision, 3D visual SLAM, machine/deep learning and AI, HPC, robotics/vehicle autonomy, assistive technology, and robotic inspection.



Wei Li received the bachelor and master degrees from the Beijing University of Aeronanautics and Astronautics. He received the Ph.D. degree in the Department of Electrical Engineering, the City College of New York (CCNY/CUNY City College), USA. He was advised by Prof. Zhigang Zhu from CUNY. Currently he is an Applied Scientist in Amazon Computer Vision Science Team working on image classification and object tracking. His research interests include deep learning in facial analysis and facial related assistive technology, video processing, multiple object tracking, and motion learning.

Howard Brand received the bachelor and master degrees of science in mechanical engineering from Virginia Polytechnic Institute and State University, USA, in 2012 and 2016, respectively. He is currently a Ph.D. candidate at Clemson University International Center for Automotive Research (CU-ICAR), USA. His research interests include computer vision, machine learning, 3D mapping, and structural health monitoring.

Biao Jiang received the master degree in interdisciplinary studies in 2010, and the Ph.D. degree in electrical engineering in 2013 from the City College of New York (CCNY/CUNY City College), USA. He currently works at the CCNY Robotics Lab as a Research Scientist. His research interests include computer vision, SLAM, and wireless communication.

Jizhong Xiao (M'98–SM'06) is a Professor of electrical engineering at the City College of New York (CCNY/CUNY City College), USA, and a Doctoral Faculty Member of the Ph.D. program in computer science at CUNY Graduate Center. He received the Ph.D. degree from Michigan State University, USA, in 2002, the M.E. degree from Nanyang Technological University, Singapore, in 1999, the M.S. and B.S. degrees from the East China Institute of Technology, in 1993 and 1990, respectively. He started the robotics research program at CCNY in 2002 as the Founding Director of CCNY Robotics Lab. His current research interests include robotics and control, cyberphysical systems, autonomous navigation and 3D simultaneous localization and mapping (SLAM), real-time and embedded computing, assistive technology, multiagent systems, and swarm robotics. He has published more than 160 research articles in peer reviewed journal and conferences. He received the U.S. National Science Foundation CAREER Award in 2007, the CCNY Outstanding Mentor Award in 2011, and the Humboldt Research Fellowship for Experienced Researchers from the Alexander von Humboldt Foundation, Germany, from 2013 to 2015.