Smart Structures and Systems, *Vol. 31*, *No. 6* (*2023*) *561-571*
https://doi.org/10.12989/sss.2023.31.6.561

BIM model-based structural damage localization using visual-inertial odometry

Junyeon Chung, Kiyoung Kim and Hoon Sohn*

Department of Civil Engineering, Korean Advanced Institute for Science and Technology, Daejeon 34141, South Korea

(*Received May 2, 2023, Revised May 17, 2023, Accepted May 26, 2023*)

Abstract. Ensuring the safety of a structure necessitates that repairs are carried out based on accurate inspections and records of damage information. Traditional methods of recording damage rely on individual paper-based documents, making it challenging for inspectors to accurately record damage locations and track chronological changes. Recent research has suggested the adoption of building information modeling (BIM) to record detailed damage information; however, localizing damages on a BIM model can be time-consuming. To overcome this limitation, this study proposes a method to automatically localize damages on a BIM model in real-time, utilizing consecutive images and measurements from an inertial measurement unit in close proximity to damages. The proposed method employs a visual-inertial odometry algorithm to estimate the camera pose, detect damages, and compute the damage location in the coordinate of a prebuilt BIM model. The feasibility and effectiveness of the proposed method were validated through an experiment conducted on a campus building. Results revealed that the proposed method successfully localized damages on the BIM model in real-time, with a root mean square error of 6.6 cm.

Keywords: building information modeling; damage detection; damage localization; structural maintenance, visual-inertial odometry

1. Introduction

The deterioration of buildings, bridges, and tunnels has emerged as a significant issue. For instance, in Japan and the United States, a considerable percentage of infrastructure was built over 50 years ago (American Society of Civil Engineers 2021, Japan Road Bureau 2018). Moreover, according to Canada's Infrastructure Report Card (Valinejadshoubi *et al*. 2019), 12% of buildings, bridges, and tunnels are either inadequate or extremely inadequate. The decay of structures may lead to structural collapse, resulting in economic losses and safety hazards (Biezma and Schanack 2017, Bazzucchi *et al*. 2018). Therefore, precise inspections and damage records are crucial to comprehending the structures' status and taking appropriate action, such as repairs, before a collapse occurs.

Identifying the precise damage location and recording the damage information accurately is essential to determine the need for repair or strengthening of a structure (Yamane *et al*. 2022). However, the current inspection reports are paper-based and two-dimensional data, which makes it challenging to determine the precise location of the damage and evaluate its chronological changes.

To address this limitation, recent studies have focused on building information modeling (BIM). By leveraging BIM, damage locations can be recorded precisely, and recording them in three dimensions eliminates the possibility of confusion regarding the damage's precise location, making it easier to assess its progression. Several studies have been conducted on the use of BIM in structural inspection and maintenance. Dang *et al*. (2018) proposed a framework linking structural damage to a BIM model, while Sack *et al*. developed a system that recorded bridge conditions and mapped them onto a BIM-based model (Hüthwohl *et al*. 2018). However, the need for an inspector to manually localize the damage in the BIM has limited its use in structural maintenance.

To overcome this limitation, researchers have developed methods for automatically localizing damage in BIM. Tan *et al*. (2022) utilized a global positioning system to localize the damage detected by a vision sensor on a BIM model. However, this system cannot be used indoors because satellite signals cannot be received inside buildings. In contrast, Structure from Motion (SfM) was used to obtain damage locations. SfM is a technique that can be used to build a 3D model by identifying the camera position and orientation by processing multiple overlapping images. Ribeiro *et al*. (2020) developed a system that created a 3D model of an entire structure with marked damage using SfM. Yamane *et al*. (2022) created 3D models of areas near the damage using SfM and aligned them with the BIM model using ICP. However, these studies cannot be performed in real-time because SfM and ICP must be conducted for each inspection or damage, which requires considerable computational time. Additionally, several researchers have employed augmented reality to map defects on a BIM model (Kwon *et al*. 2014, May *et al*. 2022). They overlaid the BIM model on a real structure by utilizing fiducial markers, found defects and mapped them onto the BIM model. However, the installation of fiducial markers is necessary before the inspection, and the precise positions of these markers in the BIM coordinate system

*Corresponding author, Ph.D., Professor,
E-mail: sohnhoon@kaist.ac.kr

ISSN: 1738-1584 (Print), 1738-1991 (Online)

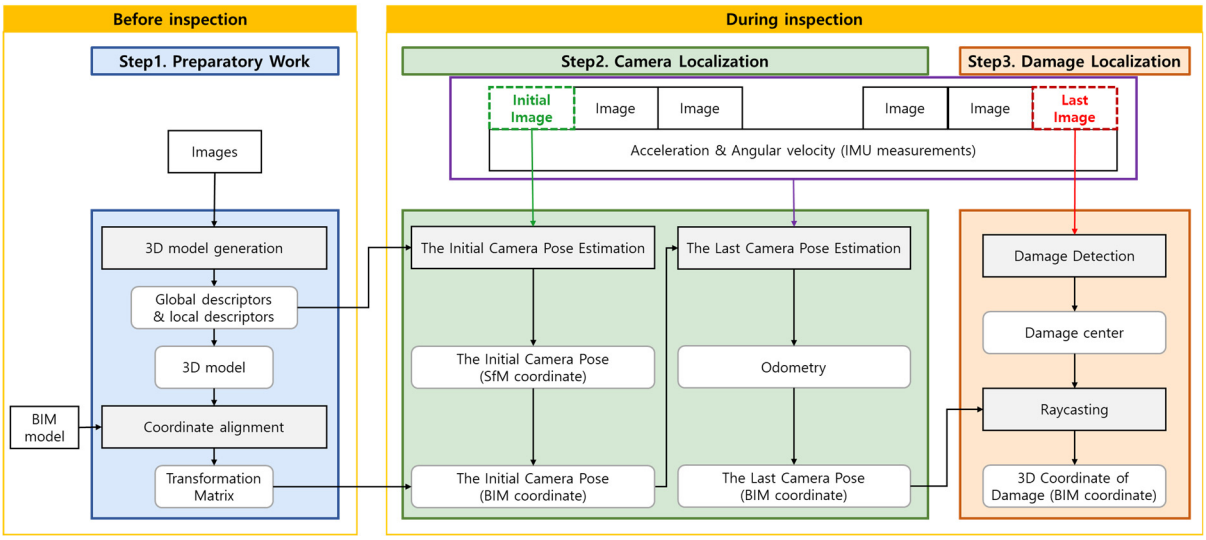


Fig. 1 Overall scheme of the proposed method

must be known in advance.

This study proposes a method for automatically localizing damage on a BIM model in real-time using a few seconds of consecutive images and IMU measurements near the damage, which is available both indoors and outdoors. The proposed method estimates the camera pose (3D position and orientation) using the visual-inertial odometry (VIO) algorithm based on point and line features and calculates the damage's location in BIM coordinates.

The contributions of this study are as follows:

(1) Unlike conventional methods that require the entire structure to be scanned or considerable computation time to compute the damage location, the proposed method can estimate the damage location in a BIM coordinate system in real-time using only a few seconds of images and IMU measurements.

(2) The proposed method, which combines line features with conventional visual-inertial odometry, enhances the accuracy of damage localization.

The remainder of this paper is organized as follows: Section 2 describes the working principles of the proposed system. Section 3 reports the performance validation of the proposed system using a case study. Finally, the conclusions and discussion are presented in Section 4.

2. Methods

Fig. 1 illustrates the process of the proposed method. This method involves three steps: (1) preparatory work, (2) camera localization, and (3) damage localization.

First, the preparatory work involves building a 3D model using SfM and determining a transformation matrix to convert the coordinate system from the SfM model to a prebuilt BIM model. This preparatory work must be completed before conducting a structural inspection.

(a) (b)

Fig. 2 Examples of captured images: (a) the location looking at an open space; (b) the location closing up the damage

During the inspection, an inspector locates the damage and starts recording time-synchronized images and IMU data (3-axes acceleration and 3-axes angular velocity) for each damage, from the location looking at an open space near the damage to the location close to the damage. The location looking at an open space, as shown in Fig. 2(a), represents the location wherein the camera view is open, and the location wherein the image was photographed can be identified by only using the image. This is called the initial camera pose. However, the location close to the damage, as shown in Fig. 2(b), enables damage detection at the pixel level; this position is denoted as the last camera pose. In the camera localization step, the last camera pose is identified by estimating the initial camera pose and odometry based on the initial camera pose to the last camera pose based on the VIO algorithm with the recorded data. Subsequently, in the damage localization step, the damage is detected at the pixel level in the image photographed at the last camera pose, and the 3D coordinates of the damage are computed by raycasting the center of the detected damage to the BIM model. Finally, the damage is automatically overlaid onto the BIM. Notably, the proposed system focuses on automatically localizing detected damages on a prebuilt BIM model instead of automatically detecting damage because several vision-based deep learning models for detecting structural damages have

already been studied.

2.1 Preparatory work

The preparatory work comprised two phases: (1) 3D model generation and (2) coordinate alignment. In the first phase, a 3D model of the structure is created based on SfM. In the second phase, a transformation matrix is computed to convert the coordinate system from the SfM model to a prebuilt BIM model.

2.1.1 3D model generation

A 3D model of the structure is created using SfM. The SfM consists of the modules shown in Fig. 3. Most of these processes are similar to conventional SFM (Schonberger and Frahm 2016). However, the repeatable and reliable detector and descriptor (R2D2) algorithm extracts keypoints from images, and the deep image retrieval method identifies image pairs captured at the same location (Revaud *et al*. 2019, Gordo *et al*. 2016). The overall SfM process is explained in detail as follows.

For each image fed into the SfM pipeline, feature extraction computes the keypoints and their local descriptors based on the R2D2 detector. The R2D2 detector exhibits a better performance for feature extraction in terms of reliability compared to conventional methods, such as the Harris corner detector or scale invariant feature transform(SIFT) (Revaud *et al*. 2019). In image matching, the global descriptor of an image is computed using a deep image retrieval algorithm, and similar images are selected based on the global descriptors (Gordo *et al*. 2016). The global descriptor computes the similarity between the images. If the global descriptors of the two images are similar, the chances of the images being captured in the same space are high, and vice versa. In feature matching, the correspondences of the keypoints among similar images are computed. If the images share a set of correspondences, they depict a common part of the scene. However, feature matching can only validate that the images share common points and do not ensure that the common points are real correspondences of 3D points. Therefore, geometric verification is employed to determine if the common points between the images correspond to the same 3D points. Finally, a scene graph is generated, where the nodes represent images and the edges represent pairs of images that have been geometrically verified.

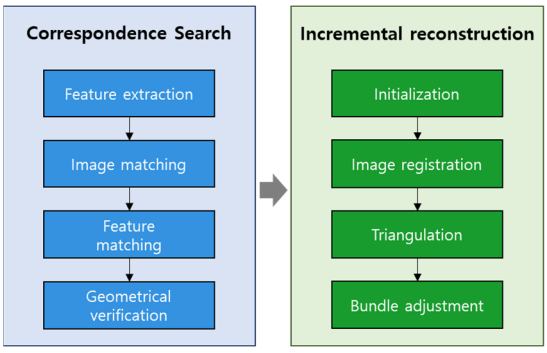


Fig. 3 Overall flowchart for Structure from motion

Once the scene graph is obtained, the 3D model is reconstructed using the following steps: During image registration, the camera poses corresponding to when the image is photographed are calculated. This is achieved by solving the perspective-n-point (PnP) problem. Next, a triangulation process is used to generate the 3D points. Finally, the optimal point cloud (PC_{sfm}) is computed using bundle adjustment. The bundle adjustment prevents inaccuracies in the estimation of the camera pose from propagating inaccuracies in the triangulation of 3D points, and vice versa; this is achieved by minimizing the sum of the reprojection error based on the Levenberg–Marquardt algorithm (Agarwal *et al*. 2010).

2.1.2 Coordinate alignment

The PC_{sfm} depicted in Fig. 4(a) has an undetermined scale and coordinate system that is inconsistent with the prebuilt BIM model. A transformation matrix for aligning the SfM and BIM models was computed using the following steps:

First, a point cloud, denoted as PC_{bim}, is created by sampling the BIM. The two-point clouds, PC_{sfm} and PC_{bim}, are then aligned along the axes, as depicted in Fig. 4(b). This alignment is performed using principal component analysis (PCA). The PCA identifies the primary axes from a point cloud because the architecture typically adheres to the Manhattan world assumption, which assumes three primary mutually perpendicular axes (Chen *et al*. 2018). The two point clouds are aligned along the axes using a transformation matrix (T_{rot}), which rotates the principal components of PC_{sfm} to the principal components of PC_{bim}.

After the two-point clouds are aligned along the axes, the scale of PC_{sfm} is adjusted to match that of PC_{bim}, as illustrated in Fig. 4(c). Scaling is performed to ensure that the story heights of the two point clouds are equal. Let h_{sfm} and h_{bim} respectively denote the story heights of PC_{sfm} and PC_{bim}. PC_{sfm} is then scaled using the transformation matrix T_{scale} defined in Eq. (2).

$$T_{scale} = \begin{bmatrix} \theta_s & 0 & 0 \\ 0 & \theta_s & 0 \\ 0 & 0 & \theta_s \end{bmatrix}, \qquad \theta_s = \frac{h_{sfm}}{h_{bim}} \tag{1}$$

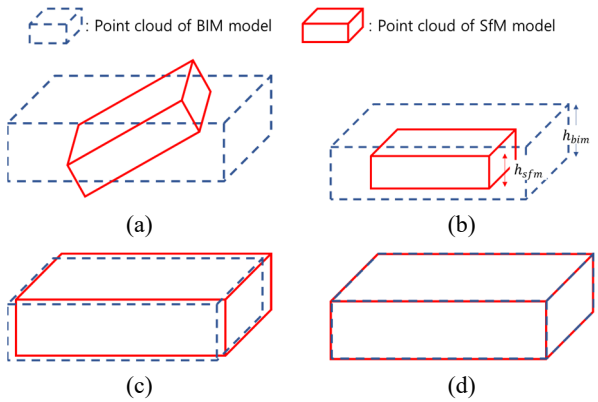


Fig. 4 Aligning SfM model with BIM model

Finally, the alignment is fine-tuned via a transformation matrix (T_{icp}) obtained through the iterative closest point (ICP) approach, as demonstrated in the process shown in Figs. 4(c) and (d). The ICP algorithm is an iterative optimization method aimed at minimizing the error between two sets of point clouds; this is achieved by attempting various transformations (Besl and McKay 1992). Suppose T_{icp}^{i} is an i^{th} arbitrary transformation matrix. Thus, the mathematical description of the ICP process is as follows

$$\begin{aligned} &PC_{icp}^{i} = \text{trans}\left(PC_{sfm}, T_{icp}^{i}\right) \\ &\text{s.t.} \min_{T_{icp}^{i}} (rmse(PC_{icp}^{i}, PC_{BIM})), \end{aligned} \tag{2}$$

where $\text{trans}(PC, T)$ represents the resulting point cloud obtained after applying the transformation matrix T to point cloud PC and $rmse(PC_1, PC_2)$ is the root mean square error (RMSE) between point clouds PC_1 and PC_2.

Finally, the transformation matrix T_{align}, obtained by multiplying T_{rot}, T_{scale}, and T_{icp}, aligns the SfM model with the BIM model. The transformation matrix T_{align} is utilized in the camera localization step to convert the initial camera pose from the coordinate system of the SfM model to that of the BIM model.

2.2 Camera localization

The objective of the camera localization step is to determine the final camera pose. The image captured in the last camera pose yields an enlarged view of the damage, which facilitates its segmentation at the pixel level. However, determining the location where the image was photographed using only the last image is impossible, because it only depicts the surface damage and yields no clues regarding the location. Therefore, the initial camera pose is estimated in advance, and the last camera pose is determined by estimating the odometry from the initial to the last camera pose. The details are as follows.

2.2.1 The initial camera pose estimation

Unlike the last image, the initial image contains several key points indicating the actual location. Therefore, the initial camera pose is computed in a manner similar to the SfM described in Section 2.1.1.

The initial image is considered as the newly added image in the SfM pipeline. Notably, key points with their local and global descriptors for all images used in SfM are available to us. The keypoints and their local descriptors in the initial image were extracted using an R2D2 detector (Revaud *et al*. 2019). Among the images used in SfM, those similar to the initial image were selected by comparing their global descriptors (Gordo *et al*. 2016). Subsequently, feature matching was used to determine the correspondences of keypoints between the initial and selected images, and geometric verification added a new node and edges to the previously constructed scene graph for the initial image. Finally, the PnP algorithm was used to estimate the initial camera pose. However, the estimated initial camera position was within the coordinate system of the SfM model. Therefore, the estimated initial camera pose was converted into the coordinate system of the BIM model by multiplying it with T_{align}, which was calculated in the preparatory work. Finally, the initial camera position ($p_{b_0}^{w}$) and orientation in the quaternion form ($q_{b_0}^{w}$) in BIM coordinates are obtained.

2.2.2 The last camera pose estimation

The last camera pose is calculated by estimating the odometry from the initial camera pose to the last camera pose. Odometry was estimated using both time-synchronized images and IMU measurements, employing the framework outlined in Fig. 5. This concept builds on VINS-MONO by introducing an ensemble pose-estimation strategy that integrates both point and line features (Qin et 2018).

Odometry estimation is initiated by preprocessing the two types of raw information measured by the camera and the IMU separately. Point and line features are detected and tracked concurrently for each incoming image captured by the camera. The proposed method adopts the Shi–Tomasi corner detector and the Kanade–Lucas–Tomasi tracker for the detection and tracking of point features, respectively (Lucas and Kanade 198, Zaccolo 2002). The line features are detected using a line segment detector and matched using a local binary descriptor (Von Gioi *et al*. 2010, Zhang and Koch 2013). For the raw gyroscope and accelerometer measurements obtained using the IMU, we followed the work of VINS-MONO to pre-integrate them between two consecutive frames. After preprocessing all the measurements, the state vector X, defined in Eq. (4), is optimized by minimizing the cost function illustrated in Eq. (5).

$$\begin{aligned} &\text{X} = \left[x_0, x_1, \cdots, x_{n_c}, P_0, P_1, \cdots, P_{n_p}, L_o, L_1, \cdots, L_{n_l}\right] \\ &x_k = \left[p_{b_k}^{w}, q_{b_k}^{w}, v_{b_k}^{w}, b_a^{b_k}, b_g^{b_k}\right], \end{aligned} \tag{3}$$

where x_k is the k^{th} IMU position $p_{b_k}^{w}$, the orientation in the quaternion form is $q_{b_k}^{w}$, the velocity is $v_{b_k}^{w}$ in the BIM

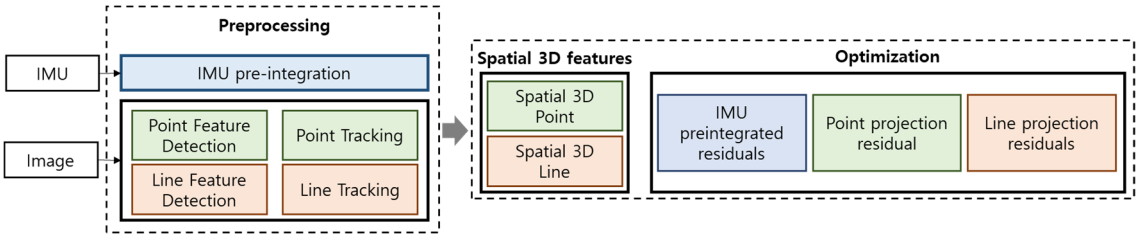


Fig. 5 Overall scheme of the last camera pose computation

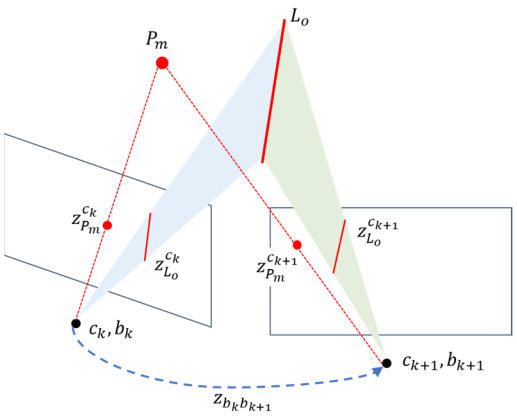


Fig. 6 Explanation of terms in the cost function

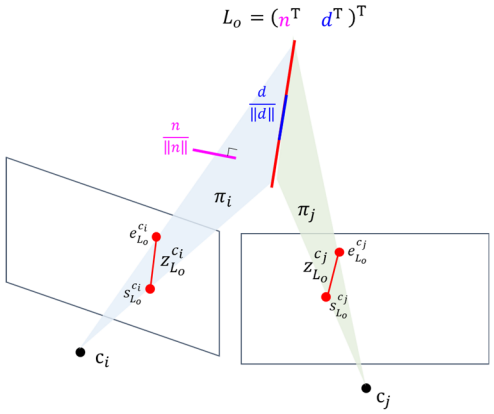


Fig. 7 Plücker coordinate of spatial 3D line

model coordinate system (world coordinate system), the acceleration bias is $b_a^{b_k}$, and the gyroscope bias is $b_g^{b_k}$ in the k^{th} IMU body frame. P_m and L_o represent the m^{th} spatial 3D point and o^{th} spatial 3D line feature, respectively. In addition, n_c, n_p, and n_l denote the total number of camera frames, spatial 3D points, and line features, respectively, during the recording.

$$\min_{x}\left(\sum_{k,o\in L}\left\|r_L\left(z_{L_o}^{c_k},X\right)\right\|+\sum_{k,m\in F}\left\|r_P\left(z_{P_m}^{c_k},X\right)\right\|+\sum_{k\in B}\left\|r_{imu}\left(z_{b_k b_{k+1}},X\right)\right\|\right) \tag{4}$$

In the cost function, $r_L(z_{L_o}^{c_k},X)$, $r_P(z_{P_m}^{c_k},X)$, and $r_{imu}(z_{b_k b_{k+1}},X)$ denote the line feature projection, point feature projection, and integrated IMU measurement residuals, respectively. $z_{L_o}^{c_k}$ and $z_{P_m}^{c_k}$ represent the line and point features in the k^{th} camera frame corresponding to the spatial 3D line L_o and point P_m, respectively, as illustrated in Fig. 6.

A 3D spatial line is represented in Plücker coordinate as $(n^T, d^T)^T \in \mathbb{R}^6$, where $n \in \mathbb{R}^3$ denotes the normal vector of the plane determined by the line and the coordinate origin and $d \in \mathbb{R}^3$ denotes the direction vector determined according to the two endpoints of the spatial 3D line (Fu *et al.* 2020). For instance, a spatial 3D line can be readily reconstructed in Plücker coordinates when observed in two frames. As shown in Fig. 7, a spatial 3D line L_O is captured at camera positions c_i and c_j and respectively projected onto $z_{L_o}^{c_i}$ and $z_{L_o}^{c_j}$. The line feature $z_{L_o}^{c_i}$ in the normalized image plane can be represented by two endpoints: $s_{L_o}^{c_i}$ and $e_{L_o}^{c_i}$. Three non-colinear points, including two endpoints of the line and the coordinate origin, determine a plane π_i. Regarding another camera, c_j, plane π_j is constructed in the same manner. Given the two plane π_i and π_j, the dual Plücker matrix L^* can be computed using Eq. (6), and the Plücker coordinates of the 3D spatial line L_o are readily extracted from the dual Plücker matrix L^*.

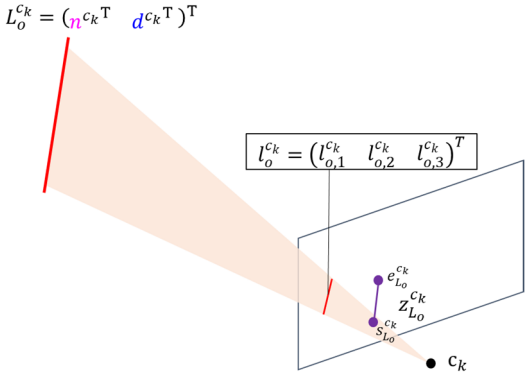


Fig. 8 Line projection residual

$$L^* = \begin{bmatrix}[d]_\times & n\\ -n^T & 0\end{bmatrix} = \pi_i\pi_j^T - \pi_i\pi_j^T \tag{5}$$

The line-feature projection residual is modeled in terms of the distance from the endpoints of the observed line feature in the normalized image plane to the corresponding projected line. The endpoints of $z_{L_o}^{c_k}$ are represented as $s_{L_o}^{c_k}$ and $e_{L_o}^{c_k}$, and the spatial 3D line L_O in the coordinate system of the k^{th} camera is expressed as $L_o^{c_k} = [n^{c_k T} \quad d^{c_k T}]^T$ as illustrated in Fig. 8. The 3D spatial line $L_o^{c_k}$ is projected onto the k^{th} camera frame as $l_o^{c_k} = [l_{o,1}^{c_k} \quad l_{o,2}^{c_k} \quad l_{o,3}^{c_k}]^T$, as explained in Eq. (7).

$$l_o^{c_k} = \begin{bmatrix} l_{o,1}^{c_k} \\ l_{o,2}^{c_k} \\ l_{o,3}^{c_k} \end{bmatrix} = \begin{bmatrix} f_y & 0 & 0 \\ 0 & f_x & 0 \\ -f_y c_x & -f_x c_y & f_x f_y \end{bmatrix} n^{c_k}, \tag{6}$$

where f_x and f_y are the focal lengths of the camera and c_x and c_y are the principal points of the camera, respectively. The line-feature projection residual of L_O in the k^{th} camera frame is defined in Eq. (8).

$$r_l(z_{L_o}^{c_k},X) = \begin{bmatrix} d(s_{L_o}^{c_k}, l_o^{c_k}) \\ d(e_{L_o}^{c_k}, l_o^{c_k}) \end{bmatrix} \tag{7}$$

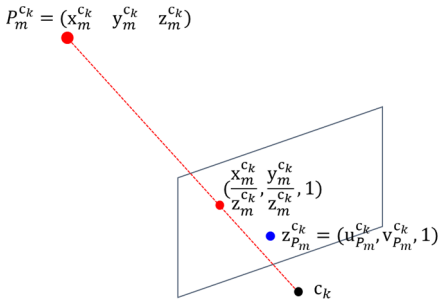


Fig. 9 Point projection residual

Herein, $d(s,l)$ indicates the distance function from endpoint s to the projection line $l = [l_1 \quad l_2 \quad l_3]^T$.

$$d(s,l) = \frac{s^T l}{\sqrt{{l_1}^2 + {l_2}^2}} \tag{8}$$

The point-feature projection residual is defined as the distance between the observed point feature in the normalized image plane and the corresponding projected point. Moreover, the spatial 3D point P_m in the coordinate system of the k^{th} camera is expressed as $P_m^{c_k} = [x_m^{c_k} \quad y_m^{c_k} \quad z_m^{c_k}]$, and the corresponding observed point feature is denoted as $z_{P_m}^{c_k} = [u_{P_m}^{c_k} \quad v_{P_m}^{c_k} \quad 1]^T$, as shown in Fig. 9. Subsequently, the point-feature projection residual of P_m in the k^{th} camera frame is defined using Eq. (10):

$$r_P(z_{P_m}^{c_k}, X) = \begin{bmatrix} \frac{x_m^{c_k}}{z_m^{c_k}} - u_{pm}^{c_k} \\ \frac{y_m^{c_k}}{z_m^{c_k}} - v_{pm}^{c_i} \end{bmatrix} \tag{9}$$

For the integrated IMU residual, the relation among the IMU position, velocity, and orientation during the duration $t \in [i,j]$ are firstly defined as below.

$$\begin{aligned} p_{b_j}^w &= p_{b_i}^w + v_{b_i}^w \Delta t - \frac{1}{2} g^w \Delta t^2 + R_{b_i}^w \alpha_{b_i b_j} \\ v_{b_j}^w &= v_{b_i}^w - g^w \Delta t + R_{b_i}^w \beta_{b_i b_j} \\ q_{b_j}^w &= q_{b_j}^w \otimes q_{b_i b_j}, \end{aligned} \tag{10}$$

Herein,

$$\begin{aligned} \alpha_{b_i b_j} &= \int \int_{t=i}^{t=j} R_{b_i}^{b_t} a_{b_t} \delta t^2 \\ \beta_{b_i b_j} &= \int_{t=i}^{t=j} R_{b_i}^{b_t} a_{b_t} \delta t \\ q_{b_i b_j} &= \int_{t=i}^{t=j} q_{b_i b_t} \otimes \begin{bmatrix} 0 \\ \frac{1}{2} w_{b_t} \end{bmatrix} \delta t, \end{aligned} \tag{11}$$

where a_{b_t} and w_{b_t} are the raw measurements of the 3-axis accelerometer and gyroscope at time t, respectively. g^w is the gravity vector in the world coordinate, and Δt is the time difference between i and j. $\otimes$ denotes the quaternion multiplication operation.

Accordingly, the integrated IMU measurement residual $r_{imu}(z_{b_k b_{k+1}}, X)$ can be defined as

$$r_{imu}(z_{b_k b_{k+1}}, X) = \begin{bmatrix} R_{b_k}^w \left(p_{b_{k+1}}^w - p_{b_k}^w - v_{b_k}^w \Delta t + \frac{1}{2} g^w \Delta t^2 \right) - \alpha_{b_k b_{k+1}} \\ 2\left[q_{b_{k+1} b_k} \otimes \left(q_{b_k}^w \otimes q_{b_{k+1}}^w \right) \right]_{real} \\ R_{b_k}^w \left(v_{b_{k+1}}^w - v_{b_k}^w + g^w \Delta t \right) - \beta_{b_k b_{k+1}} \\ b_a^{b_{k+1}} - b_a^{b_k} \\ b_g^{b_{k+1}} - b_g^{b_k} \end{bmatrix},$$

where $R_{b_k}^w$ represents the rotation matrix from the IMU body to the world frame and $[\cdot]_{real}$ extracts the real part.

To minimize the cost function, optimization was performed based on the Levenberg–Marquardt method (Ranganathan 2004). In the optimized state vector, the last camera position $p_{b_{n_c}}^w$ and orientation $q_{b_{n_c}}^w$ were acquired from the last state vector x_{n_c}.

2.3 Damage localization

The damage localization step consists of two phases: (1) damage detection and (2) raycasting. Damage detection automatically identifies damage at the pixel level and calculates its center in the image. In contrast, raycasting calculates the location of the damage by determining the intersection between the BIM model and the direction vector pointing towards the damage from the camera. The details are as follows.

2.3.1 Damage detection

To detect damage in the last image, this study utilized the SegNet model, which is a widely used deep learning architecture for vision-based applications (Li *et al.* 2019) [26]. The model comprises an encoder module and a decoder module, as illustrated in Fig. 10. In the encoder module, an input image is converted into feature maps four times using two 3 × 3 convolutions and 2 × 2 max pooling. Each time the number of feature channels doubles, the size of the feature map is halved by extracting higher-level

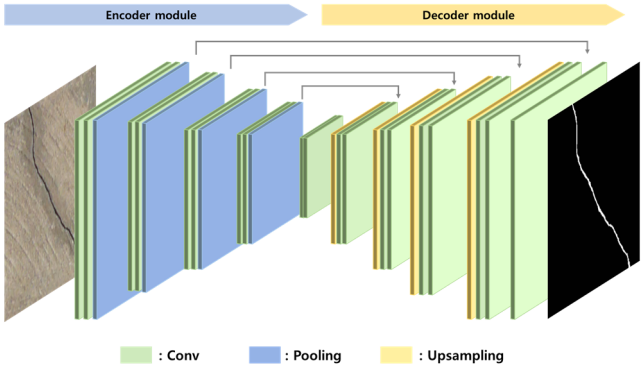


Fig. 10 Proposed model architecture

features. In the decoder module, the feature map from the encoder module is expanded four times using 2 × 2 upsampling and two 3 × 3 convolutions. During this process, skip operations between the encoder and decoder, indicated by the gray arrows in Fig. 8, enhance the recovery of details. Thereafter, the feature map from the decoder module is fed into a 1 × 1 convolution layer with a sigmoid function to output a probability map ranging from zero to one. Accordingly, pixels with a probability greater than 0.5 are considered as the damaged area.

The model was trained to detect cracks as an example of damage. The dataset used to train this model comprised 374 images of cracks and 115 images of non-cracked surfaces obtained from industrial partners and other academic institutions. The dataset was divided into 80% training, 10% validation, and 10% testing data. The batch size was set to two during training, and the training epoch was set to 100. The random weights in the convolutional layers were initialized using He normal initializers (He *et al*. 2015). The Adam optimizer was implemented with an initial learning rate of 0.0001 (Kingma and Ba 2015). After training, the precision of the model was 85.3%, and the recall value was 93.1%.

The last image, which features an enlarged view of the damage, is fed into the trained model and outputs a binary image that indicates the presence of a crack. The damage center is then calculated by averaging the pixel coordinates of the crack.

2.3.2 Raycasting

The three-dimensional location of the detected damage can be calculated by determining the intersection of the BIM model and the directional vector of the damage, as depicted in Fig. 11(a). To obtain the directional vector of the damage, the directional vector of the camera (camera orientation) was rotated by the angle of the horizontal and vertical difference between the directional vector of the camera and that of the damage around the x- and y-axes, respectively (Yamane *et al*. 2022). Eq. (14) is used to calculate the angle of the horizontal difference θ_{d_w} between the directional vectors of the camera and damage.

$$\theta_{d_w} = \tan^{-1}\frac{d_w l_w}{2\tan(\theta_h/2)}, \tag{12}$$

where θ_h is the horizontal angle of view of the camera, l_w is the width of the image in pixels, and d_w is the horizontal distance in pixels between the center of the damage and the center of the image, as illustrated in Figs. 11(b) and (c).

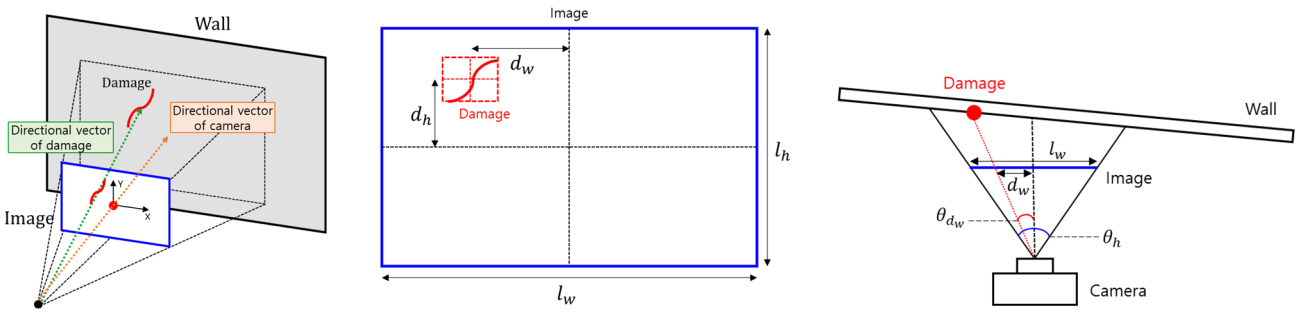
The angle of the vertical difference between the directional vector of the camera and the directional vector of the damage was calculated in the same manner. Accordingly, the 3D coordinates of the damage can be obtained by identifying the point of intersection between the direction vector of the damage and BIM. Subsequently, based on this information, the damage can be localized to the BIM.

3. Case study

A case study was conducted to validate the accuracy of damage localization on a BIM model and to measure the processing time. This study was conducted in the corridor of a campus building located at the Korea Advanced Institute of Science and Technology (KAIST). The corridor measures 67 m in length and 2 m in width with three prominent cracks on the floor.

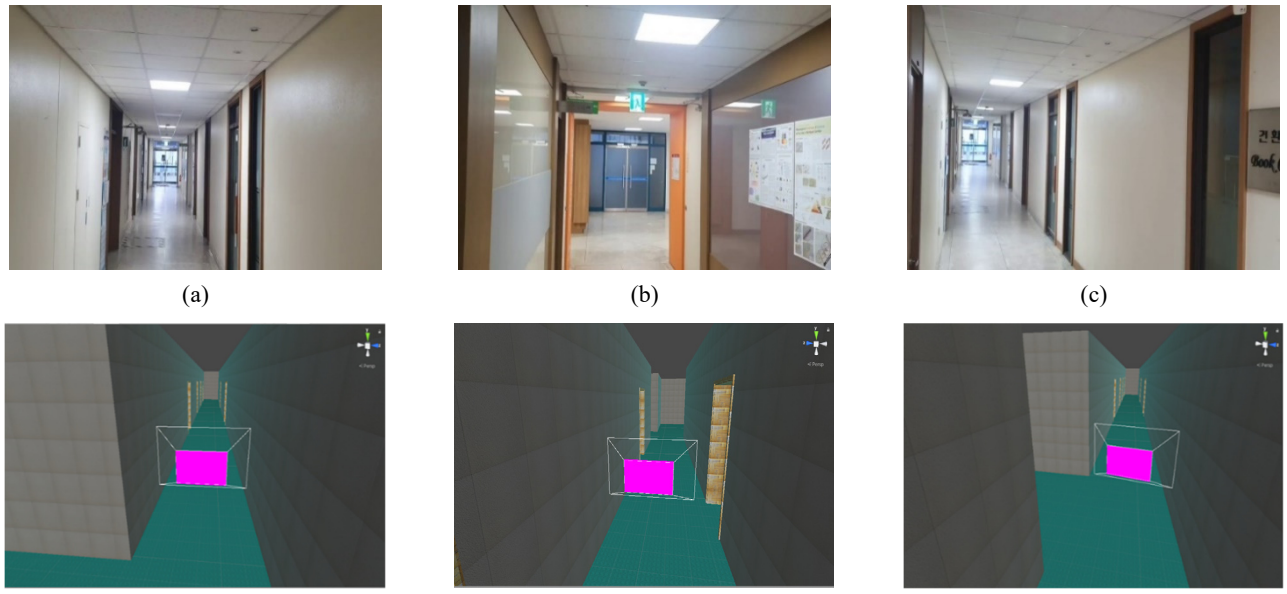
The cracks were overlaid on the prebuilt BIM model using the proposed method, and the accuracy of the crack locations was analyzed by comparing the estimated location with the ground truth. In addition, the accuracy of the proposed method was compared with that of VINS-MONO, which served as the baseline visual-inertial odometry using point features only. Finally, the computation time for each step was measured to determine the feasibility of the proposed method in real-time.

For the preparatory work, 638 images of the corridor were captured using a camera (See3CAM_CU55, manufactured by E-con Systems) with a resolution of 2304 × 1536 pixels, and a 3D point cloud model was created based on the proposed SfM. In the SfM pipeline, feature extraction, image matching, and feature matching were performed in Python. Geometric verification, triangulation, and bundle adjustment were performed using COLMAP, a popular SfM open-source program (Fisher *et al*. 2021). During computation, both the local and global descriptors of the images used in the SfM were recorded. In addition, the transformation matrix for aligning the SfM model using the BIM model was calculated in accordance with the



(a) (b) (c)

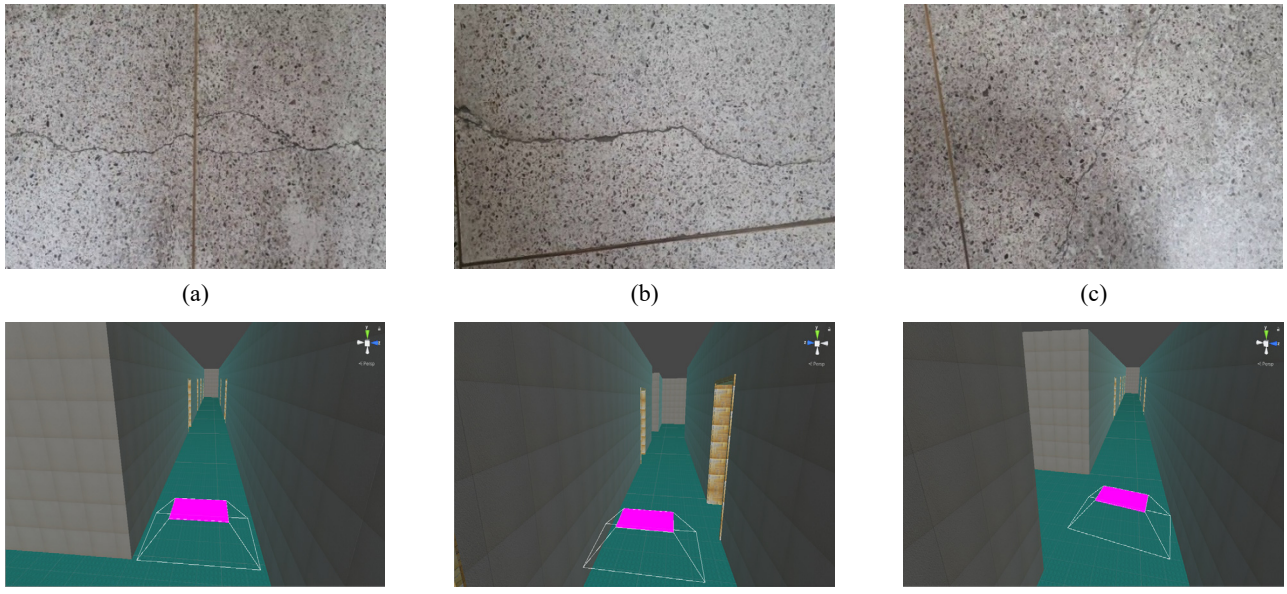
Fig. 11 Illustration of raycasting



(a) (b) (c)

(d) (e) (f)

Fig. 12 Results of the initial camera pose computation



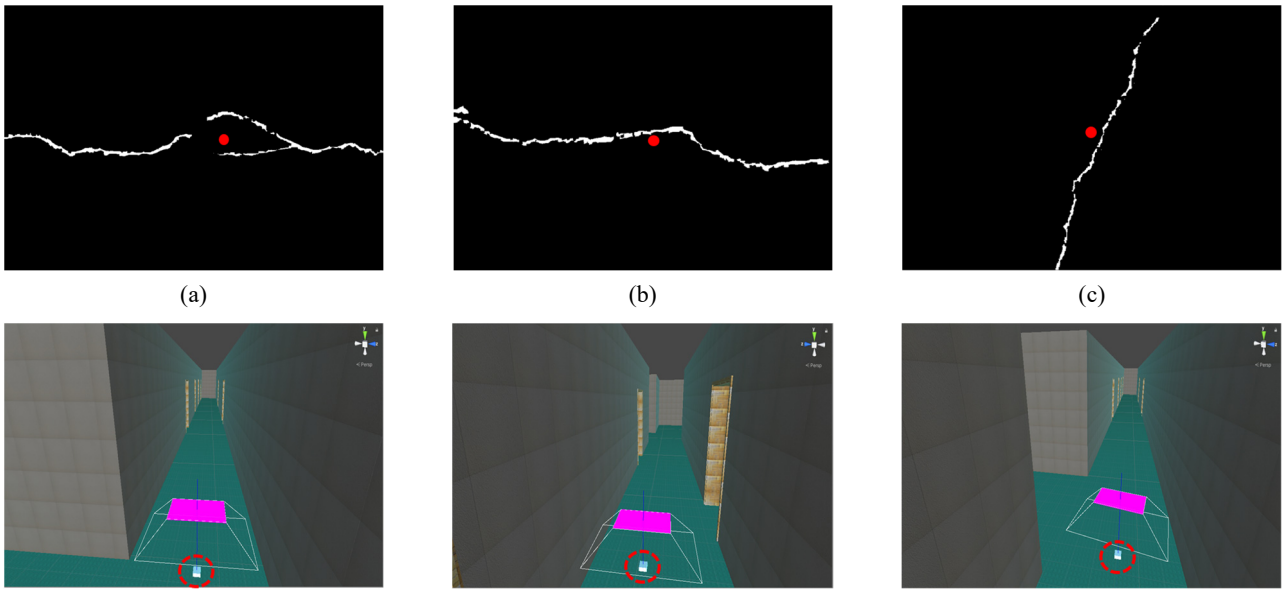
(a) (b) (c)

(d) (e) (f)

Fig. 13 Results of the last camera pose computation

specifications reported in Section 2.1.2.

After completing the preparatory work, the inspector walked around the corridor to search for cracks. The inspector discovered three cracks and recorded time-synchronized images and IMU measurements using a mobile phone (Galaxy Note 10). The recordings were made from a location looking at an open space near the crack to a location close to the crack for each of the three cracks. The device captured images with a resolution of 2288 × 1080 at a frequency of 5 Hz and measured 3-axes acceleration and angular velocity at 100 Hz. The recorded times for each crack were 4.6, 6.3, and 4.9 s, respectively. The initial images of each recording are illustrated in Figs. 12(a), (b), and (c). Because the initial image contained several key points that indicated the location where the image was captured, the initial camera pose was estimated by comparing its descriptors with those of the images used in SfM. The estimated initial pose was then expressed in the coordinates of the BIM model by multiplying it with the transformation matrix computed in the preparatory work, as shown in Figs. 12(d), (e), and (f). The last camera pose was computed by estimating the odometry from the initial camera pose to the last camera pose using the proposed visual-inertial odometry, which relied on both point and line features. Fig. 13 depicts the last camera image and the estimated last camera pose for each crack.



(a) (b) (c)

(d) (e) (f)

Fig. 14 Results of damage localization

The last images (Figs. 13(a), (b), and (c)) were fed into the trained deep learning model to detect cracks at the pixel level, and the center of the crack was computed. The crack detection results are presented in Figs. 14(a), (b), and (c), with a red circle marking the center of the crack. Finally, the location of the crack was calculated by raycasting from the center to the BIM, which was marked in the BIM. Figs. 14(d), (e), and (f) indicate that cracks are reflected in the BIM model. The results demonstrate that the proposed method can automatically integrate the damage location into the BIM.

The accuracy of the estimated crack location was analyzed in terms of the RMSE between the estimated crack location and ground truth by increasing the distance between the last camera position and the actual damage. As shown in Fig. 15, the RMSE increases with the distance. The RMSE values are 6.2, 5.8, and 7.8 cm at a distance of 10 cm; however, these values increased to 24.5, 21.8, and 22.0 at a distance of 40 cm. As the distance between the last camera position and the actual damage increases, even a small error in the last camera orientation can significantly affect the damage location estimated through raycasting.

In addition, the localization accuracy of the proposed method was compared with that of VINS-MONO, which used only point features for visual-inertial odometry. As shown in Fig. 16, the proposed method improved the accuracy of damage localization by 30% compared to VINS-MONO. The corridor contains many robust line features such as the boundaries of walls, doors, and floor patterns, which contributed to this improvement.

The computation time for both the camera localization and damage localization steps was analyzed. The computation was performed on hardware featuring an i5 core processor and 16 GB of RAM. The computation times for each step are depicted in Fig. 17. On average, the initial camera localization was executed in 4.3 s, whereas the last camera localization required the same amount of time as the

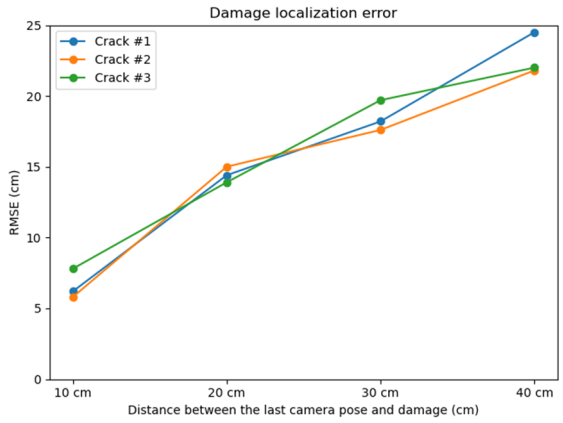


Fig. 15 Damage localization accuracy according to the distance between the last camera pose and damage

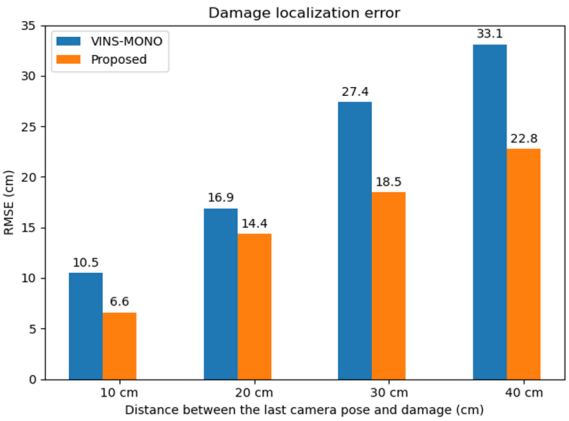


Fig. 16 Damage localization accuracy comparison of the proposed method and using VINS-MONO

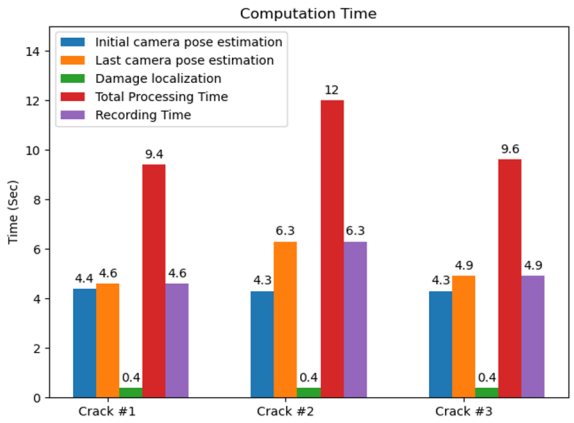


Fig. 17 Computation time measurement for each step

recording owing to its real-time processing. The damage-detection process and raycasting was performed in 0.4 s. In conclusion, the overall process took approximately 4.7 s longer than the recording time. This implies that the damage location can be marked on the BIM model in real-time given a 5-s time interval between each inspection.

4. Conclusions

In recent years, active research has been conducted in the field of damage localization in BIM because of their potential to improve structural maintenance efficiency. However, the applicability of the existing approaches is hindered by their protracted scanning and computation processes. To address this challenge, this study proposed a method for estimating BIM model-based damage locations using VIO, which can be computed in real-time. Unlike conventional methods, which require the entire target structure to be scanned to localize damage to the BIM, the proposed method can estimate the damage location in the BIM coordinate system in real-time using only a few seconds of image and IMU data proximate to the damage.

A case study was conducted at a campus building in KAIST, South Korea, to demonstrate the accuracy and effectiveness of the proposed method. The analysis showed that the system achieved a localization accuracy of 6.6 cm when closely inspected. Additionally, the distance between the last camera position and the actual damage exerted a considerable impact on the accuracy of the damage localization. Moreover, the damage location on the BIM model in real-time could be marked given that a 5-second interval was implemented between each inspection.

To further enhance the proposed method, future research should focus on addressing the following limitations. First, the damage-detection algorithm can be expanded to detect other types of damage, such as exposed rebar and corrosion, in addition to cracks. Second, additional information, such as inspection date, damage type, and damage size, can be integrated into the BIM model to develop a more comprehensive repair policy. By tracking the chronological changes in the damage status, operators can prioritize repairs based on their urgency.

Declaration of competing interest

The authors declare that they have no competing financial interests or personal relationships that may have influenced the work reported in this study.

Acknowledgments

This work was supported by a National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (grant number 2022R1C1C2008186).

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-00075, Artificial Intelligence Graduate School Program (KAIST)).

References

Agarwal, S., Snavely, N., Seitz, S.M. and Szeliski, R. (2010), "Bundle adjustment in the large", In: *Computer Vision–ECCV 2010*: *11th European Conference on Computer Vision*, *Heraklion*, Crete, Greece, September, Vol. 6312, pp. 29-42. https://doi.org/10.1007/978-3-642-15552-9_3

American Society of Civil Engineers (2021), A Comprehensive Assessment of America's Infrastructure, ASCE, p. 111.

Bazzucchi, F., Restuccia, L. and Ferro, G.A. (2018), "Considerations over the Italian road bridge infrastructure safety after the polcevera viaduct collapse: Past errors and future perspectives", *Frat. ed Integrita Strutt.*, **12**(46), 400-421. https://doi.org/10.3221/IGF-ESIS.46.37

Besl, P. and McKay, N.D. (1992), "A method for registration of 3-D shapes", *IEEE Trans. Pattern Anal. Mach. Intell.*, **14**(2), 239-256. https://doi.org/10.1109/34.121791

Biezma, M.V. and Schanack, F. (2007), "Collapse of steel bridges", *J. Perform. Constr. Facil.*, **21**(5), 398-405. https://doi.org/10.1061/(asce)0887-3828(2007)21:5(398)

Chen, J., Li, S. and Lu, W. (2022), "Align to locate: Registering photogrammetric point clouds to BIM for robust indoor localization", *Build. Environ.*, **209**, 1-29. https://doi.org/10.1016/j.buildenv.2021.108675

Dang, N., Kang, H., Lon, S. and Shim, C. (2018), "3D digital twin models for bridge maintenance", *Proceedings of the 10th International Conference on Short Medium Span Bridge*, Vol. 31, No. 73, pp. 1-9. [Online]. Available: https://www.researchgate.net/publication/331314334%0Ahttps://www.csce.ca/elf/apps/CONFERENCEVIEWER/conferences/SMSB/papers/FinalPaper_73_0508011616.doc

Fisher, A., Cannizzaro, R., Cochrane, M., Nagahawatte, C. and Palmer, J.L. (2021), "ColMap : A memory-efficient occupancy grid mapping framework", *Rob. Auton. Syst.*, **142**, p. 103755. https://doi.org/10.1016/j.robot.2021.103755

Fu, Q., Wang, J., Yu, H., Ali, I., Guo, F., He, Y. and Zhang, H. (2020), "PL-VINS: Real-Time Monocular Visual-Inertial SLAM with Point and Line Features", [Online]. Available: http://arxiv.org/abs/2009.07462

Gordo, A., Almazán, J., Revaud, J. and Larlus, D. (2016), "Deep image retrieval: Learning global representations for image search", In: *Computer Vision–ECCV 2016*: *14th European Conference*, *Amsterdam*, *The Netherlands*, *October 11-14*, *2016*, *Proceedings*, *Part VI 14* (pp. 241-257). https://doi.org/10.1007/978-3-319-46466-4_15

He, K., Zhang, X., Ren, S. and Sun, J. (2015), "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification", *Proceedings of the IEEE international conference on computer vision*, pp. 1026-1034.

Hüthwohl, P., Brilakis, I., Borrmann, A. and Sacks, R. (2018), "Integrating RC bridge defect information into BIM models", *J. Comput. Civil Eng.*, **32**(3), p. 04018013. https://doi.org/10.1061/(asce)cp.1943-5487.0000744

Japan Road Bureau (MLIT) (2018), "Roads in Japan 2018", *Minist. Land, Infastructure, Transp. Tour. Japan*, pp. 2-39. [Online]. Available: http://www.mlit.go.jp/road/road_e/index_e.html

Kingma, D.P. and Ba, J. (2015), "Adam: A method for stochastic optimization", *Proceedings of the 3rd International Conference on Learning Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1-15.

Kwon, O.S., Park, C.S. and Lim, C.R. (2014), "A defect management system for reinforced concrete work utilizing BIM, image-matching and augmented reality", *Autom. Constr.*, **46**, 74-81. https://doi.org/10.1016/j.autcon.2014.05.005

Li, S., Zhao, X. and Zhou, G. (2019), "Automatic pixel-level multiple damage detection of concrete structure using fully convolutional network", *Comput. Civil Infrastruct. Eng.*, **34**(7), 616-634. https://doi.org/10.1111/mice.12433

Lucas, B.D. and Kanade, T. (1981), "An iterative image registration technique with an application to stereo vision", Vol. 2, pp. 674-679.

May, K.W., KC, C., Ochoa, J.J., Gu, N., Walsh, J., Smith, R.T. and Thomas, B.H. (2022), "The Identification, Development, and Evaluation of BIM-ARDM: A BIM-Based AR Defect Management System for Construction Inspections", Buildings, **12**(2), p. 140.

Qin, T., Li, P. and Shen, S. (2018), "Vins-mono: A robust and versatile monocular visual-inertial state estimator", *IEEE Trans. Robot.*, **34**(4), 1004-1020. https://doi.org/10.1109/TRO.2018.2853729

Ranganathan, A. (2004), "The Levenberg-Marquardt Algorithm 3 LM as a blend of Gradient descent and Gauss-Newton itera", *Internet httpexcelsior cs ucsb educoursescs290ipdfL MA pdf*, Vol. 142, pp. 1-5. [Online]. Available: http://twiki.cis.rit.edu/twiki/pub/Main/AdvancedDipTeamB/the-levenberg-marquardt-algorithm.pdf

Revaud, J., Weinzaepfel, P., De Souza, C., Pion, N., Csurka, G., Cabon, Y. and Humenberger, M. (2019), "R2D2: Repeatable and reliable detector and descriptor", *Adv. Neural Inf. Process. Syst.*, vol. 32, no. NeurIPS.

Ribeiro, D., Santos, R., Shibasaki, A., Montenegro, P., Carvalho, H. and Calçada, R. (2020), "Remote inspection of RC structures using unmanned aerial vehicles and heuristic image processing", *Eng. Fail. Anal.*, **117**, p. 1048130. https://doi.org/10.1016/j.engfailanal.2020.104813

Schonberger, J.L. and Frahm, J.M. (2016), "Structure-from-motion revisited", *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4104-4113. https://doi.org/10.1109/CVPR.2016.445

Tan, Y., Li, G., Cai, R., Ma, J. and Wang, M. (2022), "Mapping and modelling defect data from UAV captured images to BIM for building external wall inspection", *Autom. Constr.*, **139**, p. 104284. https://doi.org/10.1016/j.autcon.2022.104284

Valinejadshoubi, M., Bagchi, A. and Moselhi, O. (2019), "Development of a BIM-based data management system for structural health monitoring with application to modular buildings: Case study", *J. Comput. Civ. Eng.*, **33**(3), p. 05019003. https://doi.org/10.1061/(asce)cp.1943-5487.0000826

Von Gioi, R.G., Jakubowicz, J., Morel, J.M. and Randall, G. (2010), "LSD: A fast line segment detector with a false detection control", **32**(4), 722-732. https://doi.org/10.1109/TPAMI.2008.300

Yamane, T., Chun, P.J. and Honda, R. (2022), "Detecting and localising damage based on image recognition and structure from motion, and reflecting it in a 3D bridge model", *Struct. Infrastruct. Eng.*, pp. 1-13. https://doi.org/10.1080/15732479.2022.2131845

Zaccolo, M. (2002), "Good features to track", *Methods Mol. Biol.*, **178**, pp. 255-258. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6755905%5Cnhttp://www.ncbi.nlm.nih.gov/pubmed/11968495

Zhang, L. and Koch, R. (2013), "An efficient and robust line segment matching approach based on LBD descriptor and pairwise geometric consistency", *J. Vis. Commun. Image Represent.*, **24**(7), 794-805. https://doi.org/10.1016/j.jvcir.2013.05.006

HJ