Digital Twin Approach for Drift-Free UAV Localization in GPS-Denied Environments

Thomas Matiki, S.M.ASCE[1]; Yasutaka Narazaki, M.ASCE[2]; Girish Chowdhary[3]; and Billie F. Spencer Jr., Dist.M.ASCE[4]

Abstract: Unmanned aerial vehicles (UAVs) hold promise for automating civil infrastructure inspections, but reliable pose estimation is crucial for autonomous navigation in GPS-denied environments. While simultaneous localization and mapping estimates UAV poses, drift often occurs and is typically corrected through loop closures. However, relying on loop closures limits UAV maneuverability to paths that enable loop formation. To overcome this, recent methods leverage digital twins by combining highly accurate three-dimensional (3D) models of the physical environment with identification of distinctive landmarks to enable drift-free localization. In earlier work, the authors proposed using a graphics-based digital twin (GBDT) for drift-free localization through landmark identification, but this approach fails in landmark-sparse environments. This paper eliminates landmark detection by matching visual patterns between the physical structure and its digital twin. UAV images are compared with views from virtual cameras positioned in the digital model. After identifying the virtual camera that best aligns with the UAV's viewpoint, the iterative closest point (ICP) algorithm is used to estimate their relative pose. To enhance accuracy, the method GBDTpose2 uses trustworthy regions, areas that are geometrically and visually similar in both the physical structure and digital twin. These regions guide the selection of 3D point cloud pairs for ICP. By computing the UAV pose relative to a virtual camera with a known pose, the approach naturally ensures drift-free localization. GBDTpose2 is validated in a laboratory environment using ROS (Robot Operating System) for real-time localization, achieving centimeter-level accuracy without drift, highlighting its potential for autonomous UAV navigation in GPS-denied environments. **DOI: 10.1061/JCCEE5.CPENG-7010.**

Author keywords: Unmanned aerial vehicles (UAVs); Localization; Pose estimation; Digital twin; Infrastructure inspection; Computer vision.

Introduction

Regular and periodic inspection of critical civil infrastructure, such as bridges, is crucial to ensure their safety and functionality, which is essential for seamless economic operations. To this end, visual inspection by trained inspectors is often required once every 24 months for highway bridges (FHWA 2004), or at least every 12 months for railroad bridges (FRA 2010) in the USA, to observe symptoms of structural distress and plan for more detailed nondestructive tests (Agdas et al. 2016). However, traditional visual inspections are slow, are expensive, pose risks to the inspectors, and are subjective (Malekloo et al. 2022; Dorafshan and Maguire 2018). To resolve these issues, many researchers have sought to employ unmanned aerial vehicles (UAVs) to automate the task of civil infrastructure inspection (e.g., Khaloo et al. 2017; Hallermann and Morgenthal 2014; Ali et al. 2021). While progress has been made toward the use of UAVs for inspections, manual control of UAVs for navigation is tedious and prone to collision with objects (Agnisarman et al. 2019). For autonomous UAV navigation to be achieved, a mission planner, a localization system, and flight controller are required (Drew et al. 2017; Kwak and Sung 2018). The mission planner generates the flight plan and the next waypoint (wherein the UAV would perform a predefined action such as taking a photo), and the localization system provides the current pose of the UAV, which the flight controller uses together with the waypoints to automatically navigate the UAV. Unfortunately, GPS is not always available around critical regions of civil infrastructure (for example, below a bridge deck), which poses significant localization challenges and limits autonomous UAV navigation and inspection (Vanegas et al. 2019; Szrek et al. 2021; Zhang et al. 2022). Even when available, GPS accuracy can vary up to several meters for commercial drones, which limits the ability of the drone to autonomously follow a preplanned path for inspection (Duncan et al. 2013). Therefore, development of techniques for localization in GPS-denied environments is necessary, to achieve the automation of regular and periodic inspection of critical civil infrastructure with UAVs.

To autonomously navigate to the next waypoint and perform a task (similar to visual inspections), a UAV needs to know where it currently is. Numerous researchers have developed strategies for localization in GPS-denied environments, which is still a challenging problem (Zhang and Hsu 2019; Lin and He 2021). One of the most popular approaches is simultaneous localization and mapping (SLAM), as found in these references (Davison et al. 2007;

[1]Ph.D. Student, Dept. of Civil and Environmental Engineering, Univ. of Illinois at Urbana-Champaign, Urbana, IL 61801. Email: tmatiki2@illinois.edu

[2]Assistant Professor, Zhejiang University–University of Illinois at Urbana-Champaign Institute, Zhejiang Univ., Haining 314400, China (corresponding author). ORCID: https://orcid.org/0000-0002-1680-5079. Email: narazaki@intl.zju.edu.cn

[3]Associate Professor, Coordinated Science Laboratory, Univ. of Illinois at Urbana-Champaign, Urbana, IL 61801. Email: girishc@illinois.edu

[4]Professor, Dept. of Civil and Environmental Engineering, Univ. of Illinois at Urbana-Champaign, Urbana, IL 61801. Email: bfs@illinois.edu

Note. This manuscript was submitted on March 18, 2025; approved on June 24, 2025; published online on September 29, 2025. Discussion period open until February 28, 2026; separate discussions must be submitted for individual papers. This paper is part of the ***Journal of Computing in Civil Engineering***, © ASCE, ISSN 0887-3801.

Newcombe et al. 2011; Mur-Artal et al. 2015; Chowdhary et al. 2013; Klein and Murray 2007; Engel et al. 2017; Bloesch et al. 2015; Campos et al. 2021; Doherty et al. 2019). SLAM seeks to localize a robot using computer vision to detect and track landmarks in the environment; however, SLAM requires loop closure (i.e., returning to a previously visited location) to minimize the accumulation of pose errors, implicitly constraining UAV trajectories to the ones that can close enough loops. Furthermore, even when loop closures are achieved, errors in the original position can corrupt the estimated poses (Guclu and Burak 2017; Tsintotas et al. 2022; Schubert et al. 2023). While artificial landmarks such as AprilTags (John and Edwin 2016) can be used to enhance recognition robustness, placing these tags at various locations on the structure can be challenging. Therefore, to enable autonomous navigation, techniques need to be developed that provide drift-free pose estimates.

Several studies have proposed methods to achieve drift-free pose estimates for mobile robots. For example, Schmitt et al. (1999) proposed using a virtual camera (VC) to project a set of lines and textures in a virtual environment to the physical world to localize a robot. To ensure viewpoint overlap between the VC and the robot for accurate localization results, they used the prior pose of the robot to reconfigure the VC. To maximize viewpoint overlap, Lin and He (2021) used a 360° virtual camera and projected predefined lines from a virtual space to the real world for localization. However, estimation of the robot's pose using a set of lines can result in ambiguities (Lin and He 2021), and correspondence of edges (or lines) in the physical world can also suffer misrepresentations and errors (Rosten and Drummond 2005; Zhou et al. 2021; Yue et al. 2023; Ulas and Hakan 2013). Although various techniques for feature matching have been developed (e.g., Lowe 2004; Sarlin et al. 2020; Chen et al. 2021; Lindenberger et al. 2023) feature matching errors can still occur in environments with inadequate distinctive features (Ma et al. 2020; Zhang et al. 2021a; Yao et al. 2021; Arshad and Kim 2023; Arshad and Park 2024). Moreover, in very large and complex environments, a single VC has a limited field-of-view, which may restrict comprehensive navigation. Real-time reconfiguration of the VC requires real-time communication between the VCs and a robot (or UAV), which is not straightforward. In another study, Conte and Doherty (2009) used geotagged images for drift-free UAV localization. Because the geotagged images already have reference position information, the authors used feature matching between geotagged images and images observed by an onboard camera on a UAV to estimate the UAV's relative pose. However, the need for preparing enough geotagged images can limit the applicability of the method (especially in GPS-denied environments). To overcome these challenges, high-fidelity digital twins can be leveraged to accurately represent and match features from the physical environment. By enabling communication between the digital and physical twins through an edge device mounted on the UAV, more descriptive features can be computed and retrieved in real time using distributed processing. Furthermore, by strategically placing multiple virtual cameras within the digital twin, the system can continuously provide accurate, drift-free pose estimates, making it well-suited for reliable UAV navigation in GPS-denied environments.

This paper presents a digital twin approach, termed herein as GBDTpose2 for drift-free UAV localization. GBDTpose2 extends the authors' previous work termed GBDTpose (Matiki et al. 2024) wherein a graphics-based digital twin (GBDT) of a structure was used to enable UAV localization in GPS-denied environments. The GBDT as presented in Wang et al. (2022) is a computer graphics model (CGM) containing structural information (e.g., finite-element model) and is intended to be a living photorealistic representation of the target structure. The CGM is designed to be continuously updated using UAV survey data to reflect structural damage or geometric changes, while corresponding adjustments are made to the finite-element model. Real-time, bidirectional communication between the GBDT and the physical structure, facilitated by an edge device mounted on the UAV, enables seamless updates to maintain the accuracy and relevance of the GBDT. The GBDTpose method localized UAVs by detecting distinctive patterns on both the physical structure and its digital twin, referred to as GBDTtags. These tags, similar to AprilTags (John and Edwin 2016), allow the UAV's pose to be estimated within the coordinate frame of the digital twin. Although GBDTpose demonstrated strong performance during field tests on a bridge, its effectiveness decreased in environments with sparse GBDTtags, which is often the case in civil structures with repetitive patterns. In addition, using a neural network for tag detection can limit real-time pose estimation on edge devices, as UAV flight controllers typically require pose updates at a rate of at least 2 Hz. To address these issues, GBDTpose2 processes the image stream from the target (physical) structure and that from the GBDT, without the intermediate step of developing GBDTtags. The image rendering in the GBDT environment runs in real time on a base-station computer. Using the Robot Operating System (ROS) messaging protocol (ROS 2018a), the VC poses and captured images are transmitted in real time to an edge device mounted on the UAV. On the edge device, GBDTpose2 uses the histogram of oriented gradient (HOG) to select the VC sustaining sufficient viewpoint overlap with the UAV's camera. After identifying the virtual camera that best aligns with the UAV's viewpoint, the iterative closest point (ICP) algorithm is used to estimate their relative pose. To enhance accuracy, GBDTpose2 uses trustworthy regions, which are areas that are geometrically and visually similar in both the physical structure and digital twin. These regions guide the selection of three-dimensional (3D) point cloud (PC) pairs for ICP. The proposed approach is validated in laboratory tests using a truss bridge as the physical twin, a UAV with mounted Jetson Nano (as the edge device), and an Intel RealSense D435 camera. The results demonstrate that localization accuracy within the centimeter to subcentimeter range is achieved for the entire duration of the navigation. Additionally, GBDTpose2 can run at 7 Hz on the Jetson Nano, indicating its potential for autonomous UAV navigation in GPS-denied environments.

Problem Formulation

This research aims to localize a UAV performing visual inspection in a GPS-denied environment (as shown in Fig. 1). The approach relies on matching textural and geometric patterns captured by the UAV's onboard camera and VCs. The UAV's onboard camera (denoted as $\mathbb{CA}$ in Fig. 1) streams RGB-D (colored and depth) images to an edge device mounted on the UAV. Meanwhile, the VCs, which are depth cameras hosted in a real-time simulator (e.g., Gazebo 2014a), run on a base-station computer. A single VC, denoted by $\mathbb{C}$, is considered in Fig. 1. Furthermore, the coordinate system of the VC is denoted by xyz, and its pose with respect to the fixed frame $\mathbb{G}$ (XYZ) is denoted by $\mathcal{P}_{\mathbb{C}}$ (where $\mathcal{P}_{\mathbb{C}} \equiv [\boldsymbol{R}_{\mathbb{C}}^{\mathbb{G}}(\mathrm{T}_{\mathbb{C}})_{XYZ}]$; $\boldsymbol{R}_{\mathbb{C}}^{\mathbb{G}}$ is the rotation matrix that transforms vectors from frame $\mathbb{C}$ to $\mathbb{G}$ and $(\mathrm{T}_{\mathbb{C}})_{XYZ}$ is the position vector of the VC with respect to the frame $\mathbb{G}$). In real time, the simulator uses ROS to wirelessly publish (i.e., transmit) pose-tagged images (i.e., images captured by the VC with associated pose, i.e., $\mathcal{P}_{\mathbb{C}}$) to the edge device. If the VC's pose is not updated, data from the GBDT can be preloaded to the edge device. The edge device processes the pose-tagged images along with the UAV's captured image to estimate the relative pose $\Delta\mathcal{P}$ of the UAV's camera with respect to the VC (where $\Delta\mathcal{P} \equiv [\Delta\boldsymbol{R}_{\mathbb{CA}}^{\mathbb{C}}(\mathrm{T}_{\mathbb{CA}})_{xyz}]$; $\Delta\boldsymbol{R}_{\mathbb{CA}}^{\mathbb{C}}$ is the rotation matrix that transforms vectors from frame $\mathbb{CA}$ to $\mathbb{C}$ and $(\Delta\mathrm{T}_{\mathbb{CA}})_{xyz}$ is the

Fig. 1. Pictorial illustration of problem formulation for a UAV navigating the environment of the physical structure and a single VC in the GBDT's environment.

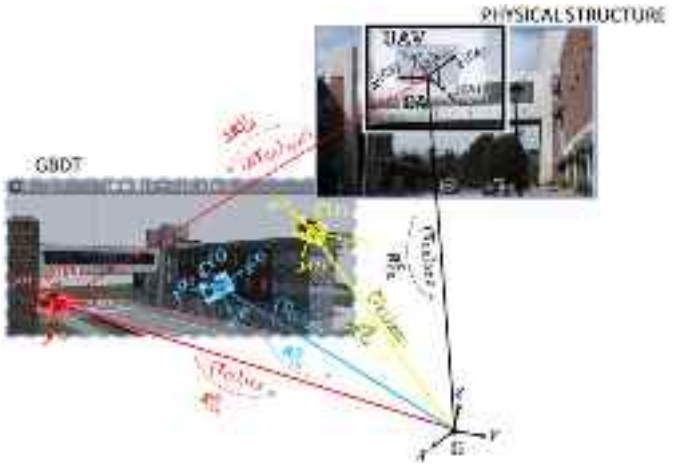

Fig. 2. Pictorial illustration of problem formulation for a UAV navigating the environment of the physical structure, with multiple VCs in the GBDT's environment.

position vector of the UAV's camera with respect to the frame $\mathbb{C}$). Because the GBDT and the real-world structure share the navigation frame $\mathbb{G}$, the UAV's camera pose with respect to $\mathbb{G}$ (i.e., $\mathcal{P}_{\mathbb{CA}} \equiv [\boldsymbol{R}^{\mathbb{G}}_{\mathbb{CA}}(\mathrm{T}_{\mathbb{CA}})_{XYZ}]$) can be recovered from $\Delta\mathcal{P}$ and the VC's pose. The detailed mathematical relationships between these quantities are provided in the Methods section, along with algorithms outlining the pose estimation process.

To increase the viewpoint overlap, multiple VCs denoted by $\mathbb{C}^{(i)}$ (where, $i = 1, 2 \ldots \mathcal{N}$, $\mathcal{N}$ being the total number of VCs used) can be placed at various locations in the simulator environment. Fig. 2, illustrates this setting, which uses the same notation as with the case of a single VC; however, the VC sustaining the highest viewpoint overlap with the UAV's camera should be selected first.

The proposed GBDTpose2 method makes the following assumptions:

1. Regions [i.e., trustworthy regions (TWRs)] exist in the GBDT and target structure that are geometrically and texturally similar.
2. At least one VC has a viewpoint overlap with the onboard UAV camera to allow for accurate pose estimation.
3. Wireless network service is available for communication between the base-station computer and edge device via ROS messages.

Regarding the first assumption, although an apparent domain shift is seen between the target structure and the GBDT, the GBDT is usually designed to be replete with regions (i.e., TWRs) that are texturally and geometrically similar to the physical structure. For example, references (Wang et al. 2022; Hoskere et al. 2022; Narazaki et al. 2021, 2022; Levine and Spencer 2022) have proposed techniques for developing a high-fidelity GBDT model, which can provide abundant TWRs required by GBDTpose2. Furthermore, previous research by the authors (Matiki et al. 2024) has shown that even with a GBDT designed using minimal textural manipulations, GBDTtags (similar to TWRs) could be detected by a neural network trained only with synthetic images from the GBDT. Nonetheless, for optimal performance, GBDTpose2 requires a feature-rich GBDT with strong visual similarity to the physical twin. Furthermore, the use of numerous VCs to capture images from various viewpoints and the option of equipping the UAV with multiple cameras can significantly increase the likelihood of observing TWRs.

For an accurate estimate of the UAV's pose to be achieved, the viewpoint of at least one VC should overlap with the UAV's camera. Ensuring sufficient viewpoint overlap with a single VC can be challenging; however, deploying multiple VCs at various locations within the GBDT significantly enhances the likelihood of success. Although this study did not consider dynamically updating the poses of the VCs, their positions could be adjusted based on the UAV's prior pose, transmitted from the edge device to the base station via ROS. Consequently, this research assumes the use of an adequately distributed set of VCs to maintain consistent viewpoint overlap, achievable using a simulation environment such as Gazebo.

Without a stable wireless connection, the transmission of data such as images, HOG features, and VCs poses to the edge device may be disrupted, potentially halting UAV localization. However, if the VCs are statically configured, their associated data can be preloaded onto the edge device, allowing the system to continue operating during temporary service interruptions. In cases where the VCs are dynamically updated, previously known VC poses and data can still support localization on the edge device until connectivity is restored. Therefore, GBDTpose2 relies on a wireless connection primarily for initialization and for ensuring the integrity of transmitted data. This communication framework aligns with the bidirectional communication strategies proposed by Molinar et al. (2023) and Hua and Masatoshi (2021) for linking digital and physical twins. The current approach, however, leverages ROS and Gazebo, which have proven to be robust for UAV localization.

Methods

This section outlines the methods used in GBDTpose2. The central scheme of the framework is to localize a UAV by aligning its viewpoint with that of a VC placed within a digital twin. This digital twin closely replicates the geometry and appearance of the real-world environment, even without GPS. Conceptually, the UAV navigates by using images from the VC as a reference map, leveraging them as visual and spatial guides to estimate its pose within the physical environment. Referring to Fig. 2, the configuration of any VC (i.e., $\mathbb{C}^{(i)}$) with respect to $\mathbb{G}$ and the configuration of the UAV's camera (i.e., $\mathbb{CA}$) with respect to $\mathbb{C}^{(i)}$ can respectively be written as

$$\boldsymbol{X}^{\mathbb{C}i}_{\mathbb{G}} \equiv \begin{pmatrix} \boldsymbol{R}^{\mathbb{G}}_{\mathbb{C}i} & (\boldsymbol{T}_{\mathbb{C}i})_{XYZ} \\ \boldsymbol{0}_{1\times 3} & \boldsymbol{1}_{1\times 1} \end{pmatrix} \in SE(3) \tag{1}$$

$$\Delta \boldsymbol{X}_{\mathrm{C}i}^{\mathbb{CA}} \equiv \begin{pmatrix} \Delta \boldsymbol{R}_{\mathbb{CA}}^{\mathrm{C}i} & (\Delta \boldsymbol{T}_{\mathbb{CA}})_{xyz^{(i)}} \\ \boldsymbol{0}_{1\times 3} & \boldsymbol{1}_{1\times 1} \end{pmatrix} \in SE(3) \tag{2}$$

Therefore, the configuration of $\mathbb{CA}$ with respect to $\mathbb{G}$ can be computed as

$$\boldsymbol{X}_{\mathbb{G}}^{\mathbb{CA}} = \boldsymbol{X}_{\mathbb{G}}^{\mathrm{C}i} \Delta \boldsymbol{X}_{\mathrm{C}i}^{\mathbb{CA}} \tag{3}$$

where

$$\boldsymbol{X}_{\mathbb{G}}^{\mathbb{CA}} \equiv \begin{pmatrix} \boldsymbol{R}_{\mathbb{CA}}^{\mathbb{G}} & (\boldsymbol{T}_{\mathbb{CA}})_{XYZ} \\ \boldsymbol{0}_{1\times 3} & \boldsymbol{1}_{1\times 1} \end{pmatrix} \in SE(3) \tag{4}$$

Given the configuration of $\mathbb{CA}$ with respect to the UAV's body reference frame denoted by $\boldsymbol{X}_{\mathbb{B}}^{\mathbb{CA}}$, the configuration of the UAV with respect to $\mathbb{G}$ can be computed as

$$\boldsymbol{X}_{\mathbb{G}}^{\mathbb{B}} = \boldsymbol{X}_{\mathbb{G}}^{\mathbb{CA}} (\boldsymbol{X}_{\mathbb{B}}^{\mathbb{CA}})^{-1} \tag{5}$$

However, the transformation matrix $\Delta \boldsymbol{X}_{\mathrm{C}i}^{\mathbb{CA}}$ is unknown and must be computed for the UAV to be localized using Eq. (5).

Fig. 3 presents the computational procedure for estimating the transformation $\Delta \boldsymbol{X}_{\mathrm{C}i}^{\mathbb{CA}}$ given a sequence of pose-tagged images published from the base-station to the edge-device. Each of the two main blocks (i.e., the edge device and base-station blocks) comprises one or more subblocks (i.e., the red boxes in Fig. 3), which are ROS nodes (i.e., programs running within ROS), that contribute to the computation of $\Delta \boldsymbol{X}_{\mathrm{C}i}^{\mathbb{CA}}$ on the edge device. Furthermore, these ROS nodes can communicate with each other either locally (i.e., within the same device) or wirelessly (i.e., between the base station and edge device) using the ROS messaging protocol. In Fig. 3, communication occurring within a device is denoted by blue solid lines, while wireless communication links are represented by black dashed lines. For wireless communication to be possible, both devices (i.e., the edge device and the base-station computer) must be on the same network (ROS 2018b). Moreover, with ROS, bidirectional communication (i.e., to and from transmission of data) is also possible, allowing the UAV's prior pose to be transmitted to the base-station computer as demonstrated in Fig. 3. To clearly describe the contribution of each of the blocks in Fig. 3 in estimating the transformation $\Delta \boldsymbol{X}_{\mathrm{C}i}^{\mathbb{CA}}$, the remainder of this sections employs the Smart Structures Technology Laboratory steel truss bridge (or SSTL bridge) as the physical twin.

Base-Station Block

Referring to the base-station block in Fig. 3, there are two ROS nodes (i.e., the *Gazebo node* and *Auxiliary node*) responsible for using the GBDT to generate the information needed for computation of $\Delta \boldsymbol{X}_{\mathrm{C}i}^{\mathbb{CA}}$ on the edge device. First, the *Gazebo node* publishes images of the GBDT captured by the VCs (together with the VCs poses) to the *Auxiliary node*. The *Auxiliary node* then transforms the RGB images of the GBDT to gradient images and also computes the HOG of the RGB images captured by all VCs. OpenCV library functions were leveraged for computation of gradient images and HOGs. These data (gradient images, depth images, VCs poses, and HOG) corresponding to all VCs are transmitted wirelessly by the *Auxiliary node* (on the base station) to the edge device. The *Main node* running on the edge device subscribes to the data published from the base station to perform further computations to estimate $\Delta \boldsymbol{X}_{\mathrm{C}i}^{\mathbb{CA}}$. Although Gazebo was used for this research to host the VCs and GBDT (because it has a ROS Interface for data transmission), other real-time simulators (e.g., Miller 2021; Morse 2009) can also be used, provided that RGB-D images and

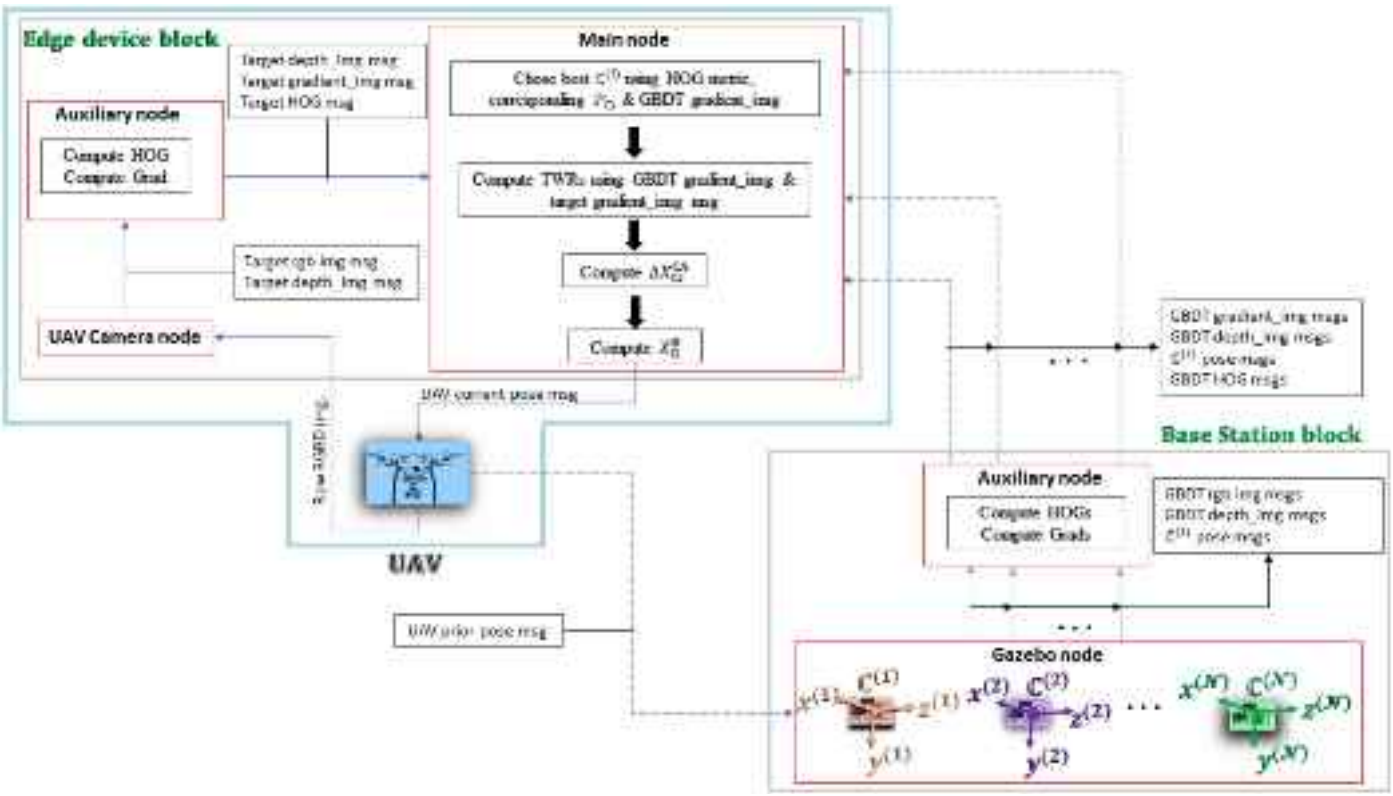

Fig. 3. Computational procedure for estimation of a UAV's relative pose with respect to VCs.

J. Comput. Civ. Eng., 2026, 40(1): 04025123

the VC poses can be obtained and transmitted to the edge device in real time.

Edge Device Block

To estimate the transformation $\Delta \boldsymbol{X}_{\mathbb{C}i}^{\mathbb{CA}}$, GBDTpose2 uses patterns of the GBDT transmitted from the base station and corresponding patterns of the physical twin. To this end, the edge device block first captures raw RGB-D images with the UAV's onboard camera, via the *UAV Camera node* (shown in Fig. 3). The captured images are transformed by the *Auxiliary node* (similar to the Auxiliary node in the base station) to compute gradient images and HOG, respectively. The *Main node* (subscribing to the Auxiliary nodes on the edge device and base station, respectively) uses the HOG similarity metric to select the $\mathbb{C}^{(i)}$ sustaining highest viewpoint overlap with $\mathbb{CA}$. Then, TWRs are detected and used with the ICP algorithm to compute $\Delta \boldsymbol{X}_{\mathbb{C}i}^{\mathbb{CA}}$. Given that the transformation matrix $\boldsymbol{X}_{\mathbb{B}}^{\mathbb{CA}}$ is known, the UAV's current pose can be calculated [using Eq. (5)] and sent to the UAV's flight controller (as shown in Fig. 3). While all computations on the edge device could be performed on the base station, with the poses then transmitted to the flight controller via the edge device, the architecture shown in Fig. 3 is designed to minimize overall latency in calculating the UAV's pose, ensuring it meets the real-time requirements of the flight controller.

Selecting the Best Virtual Camera Using Viewpoint Similarity Metric

Localization of the UAV using Eq. (5) requires that the UAV's onboard camera sustains a viewpoint overlap with the VCs (i.e., viewpoint similarity); however, determining the similarity between two images is a nontrivial task (Aljanabi et al. 2019; Chalom et al. 2013; Douze et al. 2021). Even further, measuring the similarity between two images from different domains (i.e., the GBDT and physical twin) is more challenging due to domain shift (Pinheiro 2018; Narazaki et al. 2023; Zhang and Gao 2022). While various techniques have been developed to resolve this issue, the HOG technique for image similarity analysis has been found to be invariant to moderate geometric and light intensity changes (Dalal and Triggs 2005; Lai and Teoh 2014). Thus, this research uses HOG to quantify the viewpoint similarity of each VC (i.e., $\mathbb{C}^{(i)}$) to $\mathbb{CA}$ given their RGB image pairs. To this end, the HOG of the image corresponding to $\mathbb{CA}$, i.e., $\text{HOG}^{\mathbb{CA}} \in \mathbb{R}^{1\times M}$, is compared against $\text{HOG}^{(i)} \in \mathbb{R}^{1\times M}$ (for each $\mathbb{C}^{(i)}$) and the L_2 norm is computed. Then, the $\mathbb{C}^{(i)}$ with minimum L_2 norm is returned for subsequent computations. The computations of $\text{HOG}^{(i)}$ take place on the auxiliary node within the base-station computer, so that the edge device is relieved of these computations.

TWRs for Relative Pose Estimation with ICP

The ICP algorithm (Besl and McKay 1992; Wang and Zhao 2017; Zhang et al. 2021b) is applied to align the PCs from the physical and digital twins to estimate the relative pose between the devices corresponding to those PCs. The standard ICP algorithm is known to converge to suboptimal estimates for very large PCs and/or for PCs with very dissimilar distributions (Li et al. 2020; Marchel et al. 2020). Furthermore, the standard ICP algorithm is blind to the contents of the scene, such that if the scenes are very different, the ICP algorithm still returns a relative pose which would be erroneous. Although direct feature matching between images corresponding to $\mathbb{CA}$ and $\mathbb{C}^{(i)}$ using SIFT (Lowe 2004) can sometimes yield better results, for scenes with repetitive patterns, SIFT features can suffer significant ambiguities. Therefore, this research employs TWRs to guide the ICP algorithm for accurate relative pose estimation between $\mathbb{CA}$ and $\mathbb{C}^{(i)}$.

While the standard ICP algorithm can be applied for relative pose estimation (i.e., $\Delta \boldsymbol{X}_{\mathbb{C}i}^{\mathbb{CA}}$) with GBDTpose2, unmodeled objects present in the physical environment may introduce significant alignment errors. TWRs are therefore introduced to ensure that the sampled point clouds originate from regions in the digital twin that are known to share geometric and textural similarity with the physical environment. Additionally, by restricting the sampled point cloud to these regions, the computational load is reduced, which can enable the ICP algorithm to converge more quickly, facilitating real-time use in UAV navigation. For detection and extraction of TWRs, GBDTpose2 uses the gradient image of the GBDT (i.e., $\nabla \boldsymbol{I}^g$) and the gradient image of the physical twin (i.e., $\nabla \boldsymbol{I}^p$). Figs. 4(c–d) visualize $\nabla \boldsymbol{I}^g$ and $\nabla \boldsymbol{I}^p$ taken for the SSTL bridge (Narazaki et al. 2020). Notice in Figs. 4(c–d) that the use of gradient images significantly reduces the textural domain shift between the GBDT and the physical twin and enhances their geometric similarity.

Having computed $\nabla \boldsymbol{I}^g$ and $\nabla \boldsymbol{I}^p$, they are stripped into patches (100×100 patches are used for this research), and for each patch in $\nabla \boldsymbol{I}^g$, its histogram is computed and compared against the histogram of all patches found in $\nabla \boldsymbol{I}^p$ using the cosine similarity score, as illustrated in Fig. 4(e). If the highest similarity score for a patch in $\nabla \boldsymbol{I}^g$ with any patch in $\nabla \boldsymbol{I}^p$ exceeds a set threshold, then the patch is classified to the same group as the patch in $\nabla \boldsymbol{I}^p$ that yielded that score. While smaller patches would be desirable (as it allows the cosine similarity metric to be more discriminative), too many patches can increase the computational burden on the edge device. Hence the patch size for this research was selected by limiting the total number of patches to 24. Furthermore, a check is made to ensure overlap between the corresponding patches in $\nabla \boldsymbol{I}^g$ and $\nabla \boldsymbol{I}^p$. This is achieved by projecting the patch coordinates from the image plane into their respective camera frames using the depth values of the patch image coordinates. The intersection over union (IoU) of the projected planar patches is computed to quantify the degree of overlap. Because the depth values are derived from the RGB-D images, this check inherently accounts for image scale. Notably, this check assumes that viewpoint overlap already exists between $\mathbb{CA}$ and the selected $\mathbb{C}^{(i)}$, as determined by the HOG similarity metric. However, this assumption becomes unreliable if the HOG similarity metric fails, as discussed in the validation section. Finally, note that some of the selected TWRs may correspond to the background of the environment. This does not pose a problem as long as the depth sensor provides valid depth values for the TWRs and the overlap check is satisfied. If undesirable, the background (and similar regions) can be isolated using pretrained object recognition models such as YOLO V8 (Ultralytics 2024, "https://docs.ultralytics.com/"). Indeed, pretrained models such as YOLO V8 can be fine-tuned to directly detect TWRs, albeit at the expense of increased computational cost.

Because computing TWRs for all patches can be expensive on the edge device, the NumPy library in Python is used for parallel computations. Thus, $\nabla \boldsymbol{I}^g$ is reshaped into $\mathbb{R}^{N_g\times p\times p}$, while $\nabla \boldsymbol{I}^p$ is also reshaped into $\mathbb{R}^{N_r\times p\times p}$ and the similarity score computed to obtain $\boldsymbol{S} \in \mathbb{R}^{N_g\times N_r}$ (where N_g and N_r represent the number of patches in $\nabla \boldsymbol{I}^g$ and $\nabla \boldsymbol{I}^p$, respectively). The maximum score for each patch in $\nabla \boldsymbol{I}^g$ is returned so that $\boldsymbol{s} \in \mathbb{R}^{N_g\times 1}$ becomes the class of each patch in $\nabla \boldsymbol{I}^g$. Moreover, the $\nabla \boldsymbol{I}^g$ corresponding to each VC is computed in the base-station computer and published wirelessly to the edge device, further relieving the edge device of the computational burdens. At the conclusion of this stage, one or more TWRs [shown in Figs. 4(f–g)] must have been detected for use.

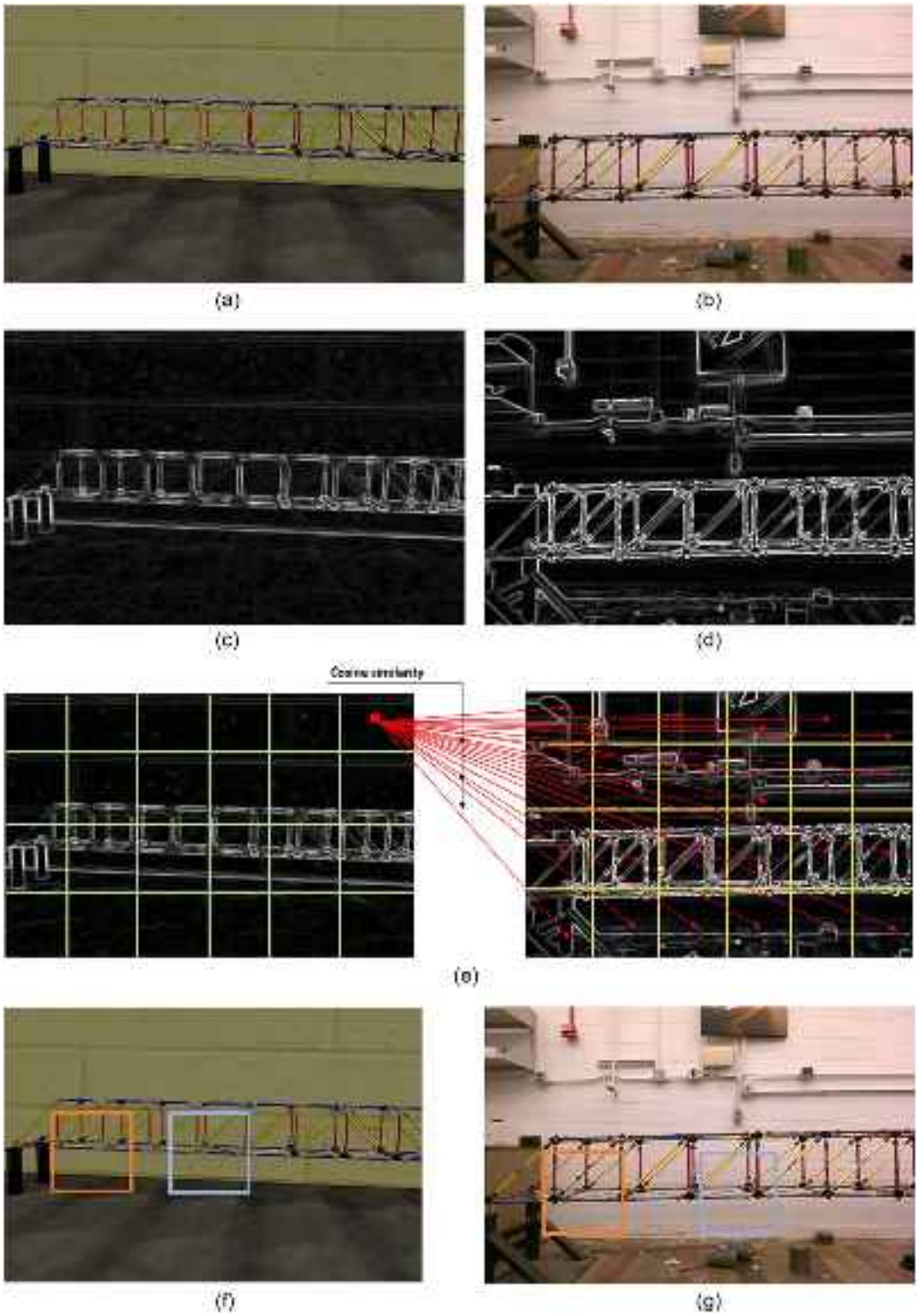

Fig. 4. (a) GBDT image; (b) physical twin image; (c) gradient image of GBDT ($\nabla \boldsymbol{I}^g$); (d) gradient image of physical twin ($\nabla \boldsymbol{I}^p$); (e) computation of similarity scores from each patch in $\nabla \boldsymbol{I}^g$ with respect to all patches in $\nabla \boldsymbol{I}^p$; (f) detected TWRs in GBDT image; and (g) detected TWRs in physical twin image.

Having detected the TWRs in the images corresponding to $\mathbb{CA}$ and $\mathbb{C}^{(i)}$, respectively, the next step is to estimate the relative pose using the ICP algorithm. Within these TWRs, random two-dimensional (2D) pixel coordinates are projected into 3D space using the corresponding depth values and intrinsic parameters of each camera to generate the 3D point clouds. This research initializes the ICP algorithm by setting $\Delta \boldsymbol{R}_{\mathbb{CA}}^{\mathbb{C}i}$ to the identity matrix (i.e. $\mathbf{I}_{3\times3}$) and $(\Delta \boldsymbol{T}_{\mathbb{CA}})_{xyz^{(i)}}$ to a vector of zeros (i.e., $0_{3\times1}$). Algorithm 1 presents the how the standard ICP algorithm can be modified using TWRs.

Algorithm 1. ICP with TWRs for Relative Pose Estimation

Load the TWRs of $\nabla \boldsymbol{I}^g$ and $\nabla \boldsymbol{I}^p$
Set number of ICP iterations, i.e., N_c
Set convergence threshold, i.e., ε
Set the number of TWRs to use, i.e., n_m
Define the number of 2D pixel coordinates to sample within each TWR, i.e., n_p
Create an empty list to store the sampled 3D points, i.e., $\boldsymbol{\Gamma}_{ci} \leftarrow []$ and $\boldsymbol{\Gamma}_{\mathbb{CA}} \leftarrow []$;
Extract first n_m TWRs in $\nabla \boldsymbol{I}^g$ and $\nabla \boldsymbol{I}^p$, respectively.
for each extracted TWR in $\nabla \boldsymbol{I}^g$ and $\nabla \boldsymbol{I}^p$ **do**:
  Sample n_p 2D pixel coordinates within the TWR in the image plane for $\mathbb{CA}$ and $\mathbb{C}^{(i)}$, respectively.
  Extract depth values of the sampled n_p 2D pixel coordinates.
  Convert the sampled 2D pixel coordinates from image coordinates to homogenous coordinates.
  Project the 2D pixel coordinates to the camera 3D frame using the corresponding intrinsic matrices.
  Store the 3D coordinates in $\boldsymbol{\Gamma}_{ci}$ and $\boldsymbol{\Gamma}_{\mathbb{CA}}$, respectively.
end for
if $\boldsymbol{\Gamma}_{ci}$ is not empty **do**:
  Initialize $(\Delta \boldsymbol{R}_{\mathbb{CA}}^{\mathbb{C}i})_0 \leftarrow \mathbf{I}_{3\times 3}$ and $(\Delta \boldsymbol{T}_{xyz^{(i)}}^{\mathbb{CA}})_0 \leftarrow \mathbf{0}_{3\times 1}$
  Given $\boldsymbol{\Gamma}_{ci}$, $\boldsymbol{\Gamma}_{\mathbb{CA}}$, $(\Delta \boldsymbol{R}_{\mathbb{CA}}^{\mathbb{C}i})_0$, $(\Delta \boldsymbol{T}_{xyz^{(i)}}^{\mathbb{CA}})_0$, N_c, and ε run the standard ICP algorithm.
  Return $\Delta \boldsymbol{R}_{\mathbb{CA}}^{\mathbb{C}i}$ and $(\Delta \boldsymbol{T}_{\mathbb{CA}})_{xyz^{(i)}}$ and compose $\Delta \boldsymbol{X}_{\mathbb{C}i}^{\mathbb{CA}}$
end if

Algorithm 2. Localization of $\mathbb{CA}$ Using Pose-Tagged Images Published by Base-Station Computer

Launch the Gazebo world file that hosts GBDT and VCs in the base-station computer.
Set confidence threshold for TWR detection, i.e., τ
Set the overlap threshold for TWRs, i.e., γ
Set the patch size p
Set the number of bins for histogram similarity computation, i.e., n_b
While UAV camera streams images, in the main node **do**:
  Subscribe to all topics published by base-station computer.
  Subscribe to $\text{HOG}^{\mathbb{CA}}$ array topic published by auxiliary node within edge device.
  Instantiate and store all the subscribed gradient images $\nabla \boldsymbol{I}^g$ for each $\mathbb{C}^{(i)}$ into $\mathbf{G}_{ci}$
  Instantiate and store all the $\text{HOG}^{(i)}$ for each $\mathbb{C}^{(i)}$ into $\mathbf{H}_{ci}$
  Instantiate and store all pose messages $\boldsymbol{X}_{\mathbb{G}}^{\mathbb{C}i}$ for each $\mathbb{C}^{(i)}$ into $\mathbf{O}_{ci}$
  Instantiate and store into $\mathbf{K}_{ci}$ all the intrinsic camera matrices K_{ci} for each VC.
  Set $\mathbf{H}_{\mathbb{CA}} \leftarrow \text{HOG}^{\mathbb{CA}}$, i.e., $\mathbf{H}_{\mathbb{CA}} \in \mathbb{R}^{1\times M}$
  Reshape $\mathbf{H}_{ci}$ into $\mathbb{R}^{\mathcal{N}\times M}$; where $\mathcal{N}$ is the total number of $\mathbb{C}^{(i)}$
  Compute $\underset{ci}{\text{argmin}} \|\mathbf{H}_{ci} - \mathbf{H}_{\mathbb{CA}}\|_2$
  Return the index k of $\mathbf{H}_{ci}$ with minimum L_2 norm.
  Given k extract the corresponding $\boldsymbol{X}_{\mathbb{G}}^{\mathbb{C}k}$ from $\mathbf{O}_{ci}$
  Given k extract the corresponding $\nabla \boldsymbol{I}_{(k)}^g \in \mathbb{R}^{H_g\times W_g}$ from $\mathbf{G}_{ci}$
  Given k extract the corresponding camera intrinsic matrix, i.e., K_k from $\mathbf{K}_{ci}$
  Extract N_g patches of size $p \times p$ from $\nabla \boldsymbol{I}_{(k)}^g \in \mathbb{R}^{H_g\times W_g}$ to compose $\mathbb{P}_{(k)}^g \in \mathbb{R}^{N_g\times p\times p}$
  Extract N_r patches of size $p \times p$ from $\nabla \boldsymbol{I}^p \in \mathbb{R}^{H_r\times W_r}$ to compose $\mathbb{P}_R \in \mathbb{R}^{N_r\times p\times p}$
  Set $TWR_k \leftarrow []$ to store the TWRs in $\nabla \boldsymbol{I}_{(k)}^g$
  Set $TWR_{\mathbb{CA}} \leftarrow []$ to store the TWRs in $\nabla \boldsymbol{I}^p$
  Instantiate $\boldsymbol{S} \leftarrow []$ to store similarity scores.
  Compute the histogram of $\mathbb{P}_R$ along the first dimension using n_b bins.
  for each patch j in $\mathbb{P}_{(k)}^g$ **do**:
    Compute the histogram of patch j using n_b bins.
    Compute the cosine similarity for patch j against all patches in $\mathbb{P}_R$
    Store the scores in $\boldsymbol{S}$
  end for
  Using $\boldsymbol{S}$, extract the patches in $\mathbb{P}_{(k)}^g$ and $\mathbb{P}_R$, respectively, with similarity scores exceeding τ.
  Extract the depth values of vertices of the patches and project the patches to the respective camera 3D frames.
  Calculate the IoU of the corresponding projected planar patches.
  Store the corresponding patches with IoU greater than γ into TWR_k and $TWR_{\mathbb{CA}}$, respectively.
  Given TWR_k, $\text{TWR}_{\mathbb{CA}}$, K_k, and $K_{\mathbb{CA}}$ run Algorithm 1 to compute $\Delta \boldsymbol{X}_{\mathbb{C}i}^{\mathbb{CA}}$
  Compute $\boldsymbol{X}_{\mathbb{G}}^{\mathbb{CA}} \leftarrow \boldsymbol{X}_{\mathbb{G}}^{\mathbb{C}k}(\Delta \boldsymbol{X}_{\mathbb{C}k}^{\mathbb{CA}})$
  Compute $\boldsymbol{X}_{\mathbb{G}}^{\mathbb{B}} \leftarrow \boldsymbol{X}_{\mathbb{G}}^{\mathbb{CA}}(\boldsymbol{X}_{\mathbb{B}}^{\mathbb{CA}})^{-1}$
end while

In Algorithm 1, the number of points n_p to be used for ICP iteration can be set to vary across the TWRs. Furthermore, NumPy can be used to allow for parallelization of the projection of TWRs from the image plane to the camera's 3D reference frame to allow for the computation of IoU (or overlap between the patches). This research used $n_m = 3$, $n_p = 10$, $N_c = 20$, and $\varepsilon = 1e-5$ for Algorithm 1. For Algorithm 2, $\tau = 0.5$, $\gamma = 0.1$, and $N_g = N_r = 24$ were used. Another key hyperparameter for detecting TWRs is the patch size p. In this research, p was set to 100, which was determined to be satisfactory based on the total number of computationally feasible patches (i.e., $N_g = N_r = 24$). These hyperparameters are not universally optimal and can be fine-tuned through trial and error in a simulation environment, such as Gazebo, prior to deployment on the UAV. Lastly, Algorithms 1 and 2 can operate without TWRs by sampling 3D point cloud pairs over the entire image.

GBDTpose2 Validation

GBDTpose2 is first validated numerically using Gazebo and ROS and subsequently experimentally in the laboratory. For both the numerical and laboratory tests, a steel truss bridge is used as the physical twin because it exhibits repetitive patterns that are characteristic of civil infrastructure. In the numerical validation, the performance of GBDTpose2 is compared with ORB-SLAM3 (Campos et al. 2021) and RTAB-Map (Labbé and Michaud 2024), both of which are feature-based SLAM algorithms that use RGB-D sequences. In the experimental validation, AprilTags (John and Edwin 2016) were installed, and the PnP algorithm (Lu 2009) is used to provide a reference solution. For the laboratory tests, GBDTpose2 is also compared against ORB-SLAM3 and RTAB-Map to observe the robustness against the drift errors.

Numerical Validation of GBDTpose2 Using Gazebo and ROS

In Gazebo, 10 Openni Kinect (Gazebo 2014b) depth cameras were set at predefined locations to function as VCs [shown in Fig. 5(a)]. The SSTL bridge structure measures approximately 5 m in length and 1.5 m in height. Positioning the VCs at a distance of 2 m allowed each camera to capture a substantial portion of the structure.

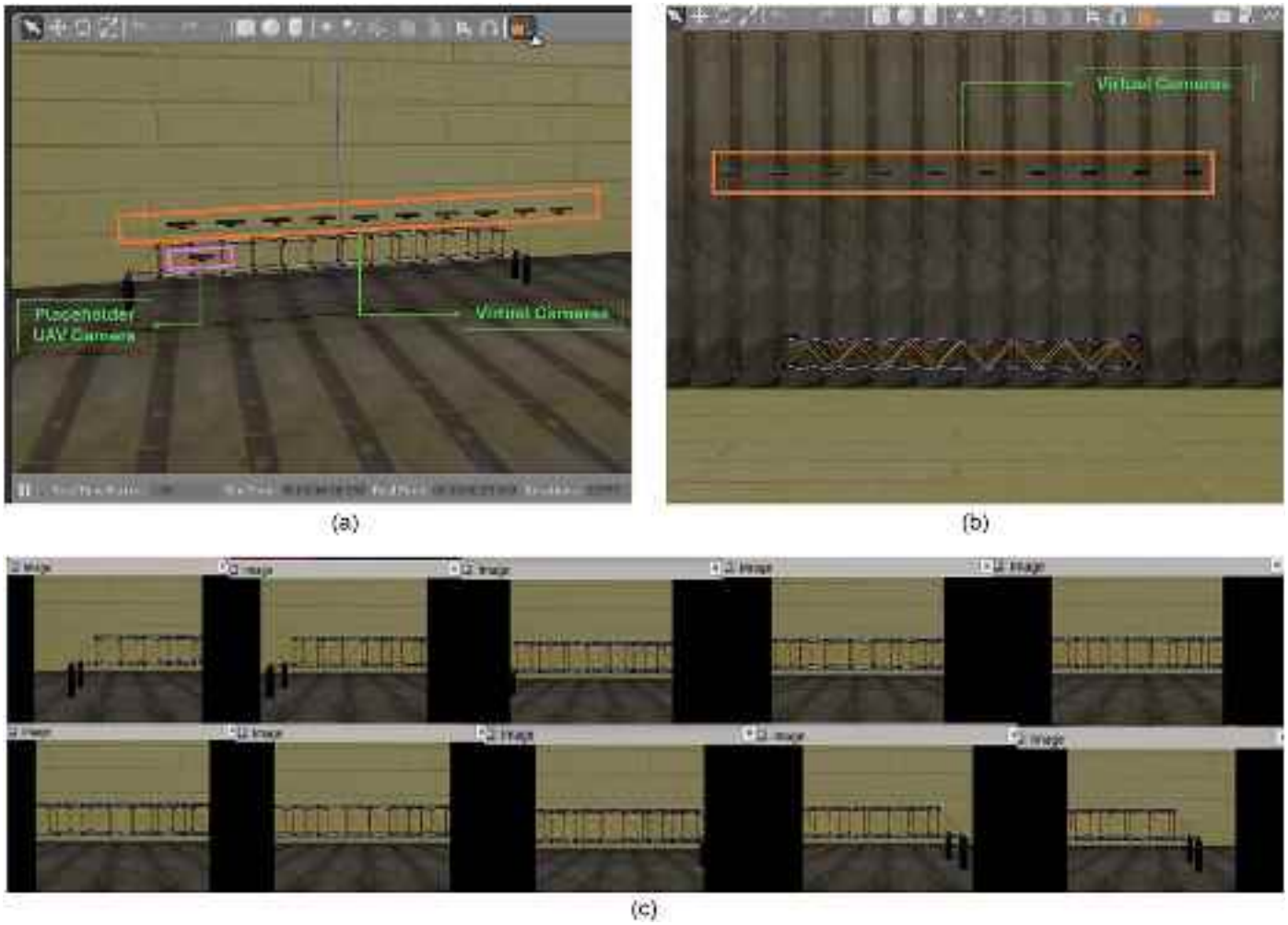

Fig. 5. (a) VCs and $\mathbb{CA}$ in Gazebo (3D view); (b) plan view: VCs are spaced 0.6 m apart, and 2 m from the bridge; and (c) RGB images captured by the 10 VCs.

For larger-scale structures, VCs may need to be placed farther away and in greater numbers to ensure adequate coverage. Although up to 1,000 VCs can be used on a 32.0 GB RAM Windows base-station computer, only 10 VCs were used because the Jetson Nano (which is the edge device used for laboratory tests) is overburdened in calculating the HOG similarity for too many VCs. In addition to the VCs in Gazebo, the UAV's onboard camera, $\mathbb{CA}$ [also shown in Fig. 5(a)], is assumed also be an Openni Kinect camera with the same parameters as the VCs. Both the $\mathbb{CA}$ and the VCs stream images of the GBDT from Gazebo to the *Main node* where the relative pose is calculated. Notice that because the pose of $\mathbb{CA}$ is known, it was used to evaluate GBDTpose2 in the numerical simulation. Figs. 5(a–c) demonstrate this setup.

Aside from comparing GBDTpose2 predictions against the ground truth, other important considerations for the numerical tests include (i) assessment of the efficacy of the HOG similarity metric for VC selection; and (ii) assessment of the pose update rate of GBDTpose2 for UAV navigation. The first numerical experiment intended to assess the performance of GBDTpose2 under pure lateral translation is shown in Fig. 6(a). The circular error probable metric (or CEP), which measures the median error, together with the root-mean-squared error (RMSE) metric, was used for evaluation. Because the estimated poses align closely with the ground truths, a zoomed-in visualization of the pose errors is provided in Fig. 6(b). As illustrated in Fig. 6(a), a very high update rate for the poses is achieved, which is attributed to the convergence speed of the ICP algorithm when TWRs are used to sample a smaller 3D point cloud. Note that the standard ICP algorithm can also converge quickly when properly initialized and may produce accurate relative poses when the GBDT closely matches the physical environment. On the other hand, direct assessment of the efficacy of GBDTpose2 in selecting the VC with highest viewpoint overlap is challenging; nonetheless, the effect of a poor VC selection can be seen from the estimated poses. For example, in between the pairs of red dashed lines in Fig. 6(a), the returned poses are less accurate due to ambiguity of the HOG similarity metric when $\mathbb{CA}$ is around the central region of the GBDT [shown in Fig. 6(d)]. These ambiguities arise due to the repetitive pattern of the bridge around the central region. Figs. 6(c and e) show some of the corresponding VC and $\mathbb{CA}$ images when $\mathbb{CA}$ is around the left and right portions of the bridge, respectively, and for these regions the corresponding patterns are very distinctive, allowing for the HOG similarity metric to work well. Moreover, the authors have observed that detailed planning of the VCs pose together with $\mathbb{CA}$'s target locations can reduce errors, which will be a direction for future studies.

The second numerical experiment is to assess the performance of GBDTpose2 under lateral translation combined with rotational perturbations of $\mathbb{CA}$ ranging from $-3°$ to $+3°$ along the roll, pitch, and yaw axes. As illustrated in Fig. 7(a), the errors for this second

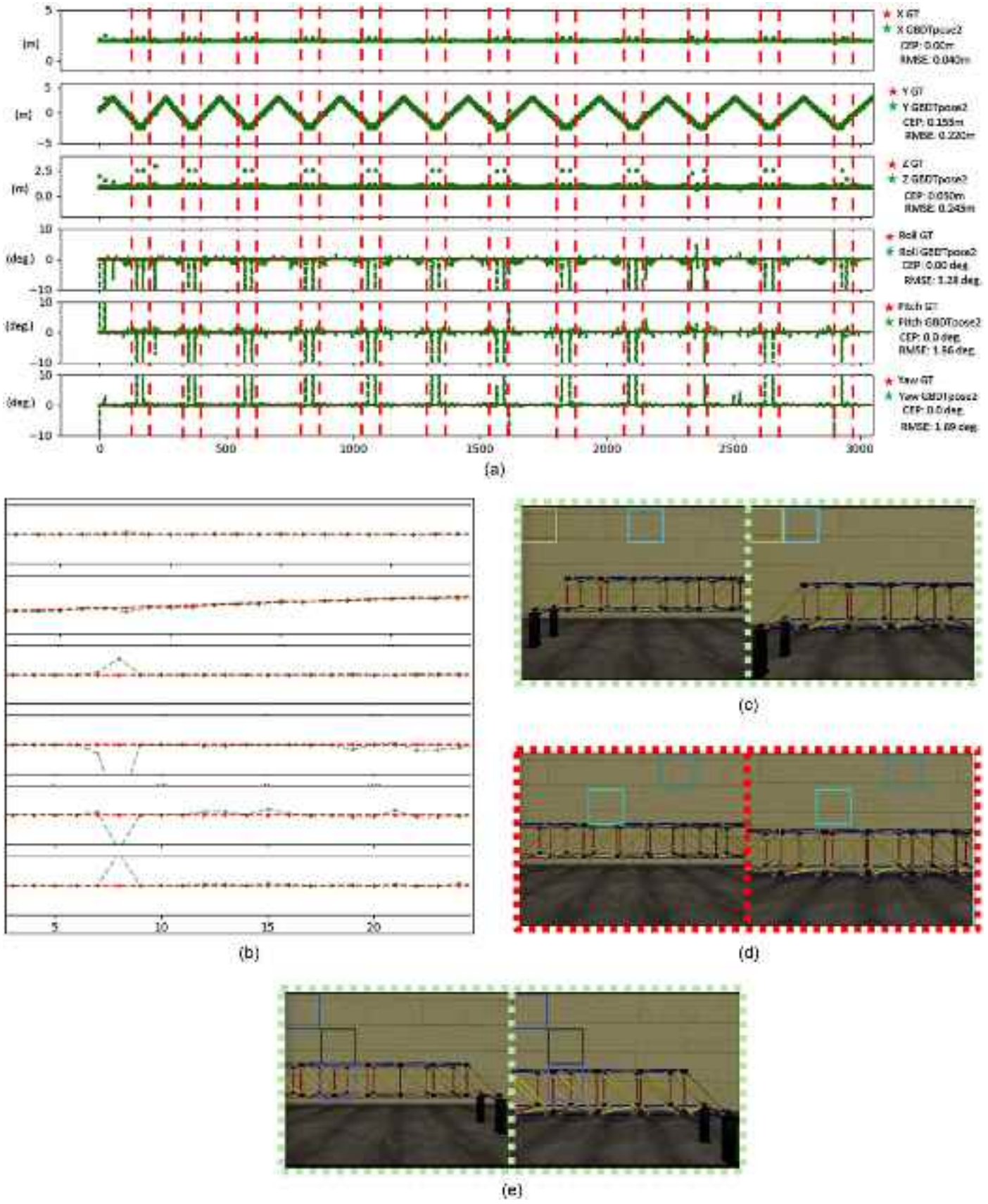

Fig. 6. (a) Pose estimation results under pure lateral translation; (b) zoomed-in visualization of pose errors in (a); (c) correct viewpoint association; (d) incorrect viewpoint association; and (e) correct viewpoint association.

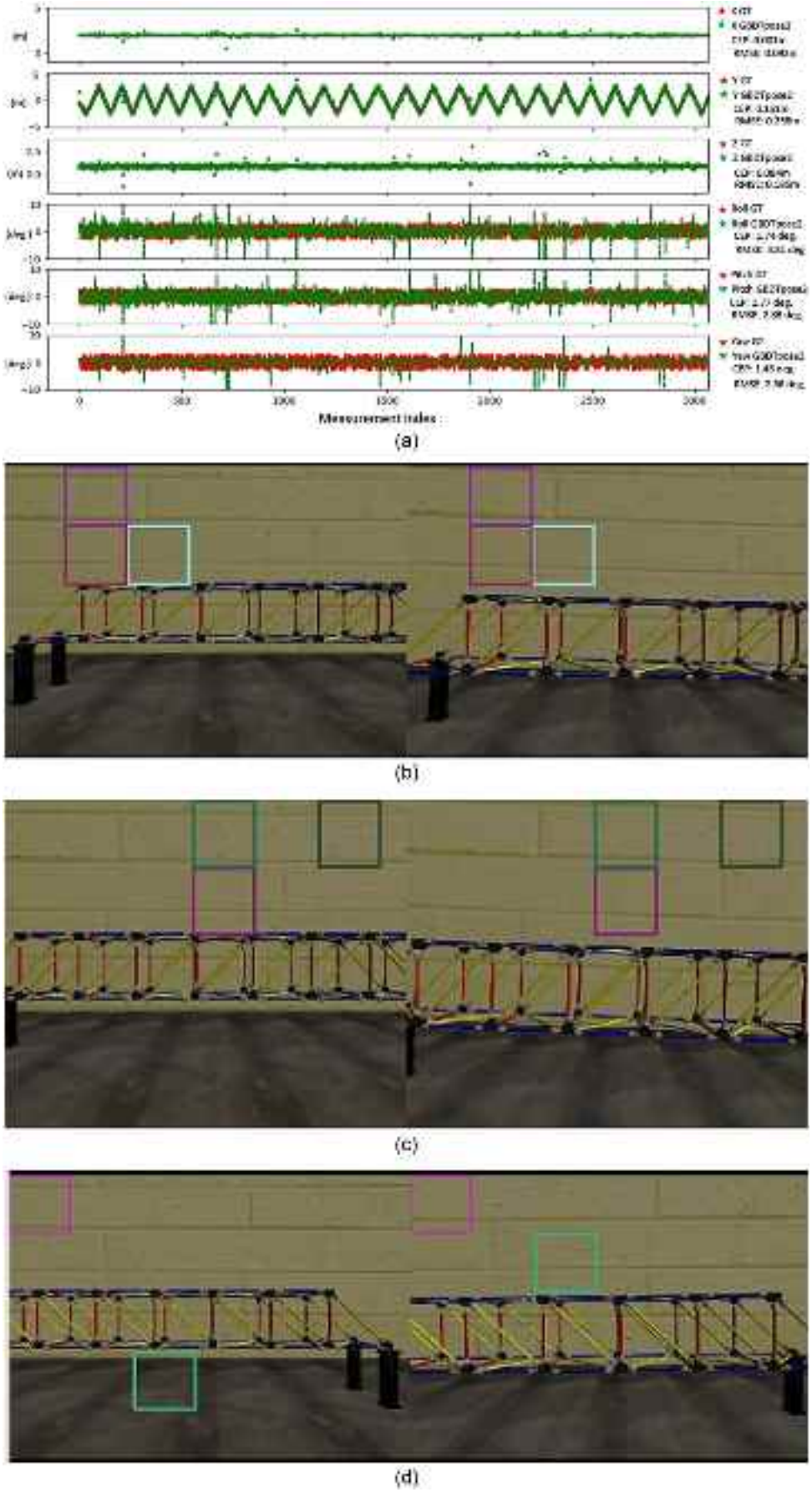

Fig. 7. (a) Pose results under lateral translation with rotations; (b) correct viewpoint association; (c) correct viewpoint association; and (d) correct viewpoint association.

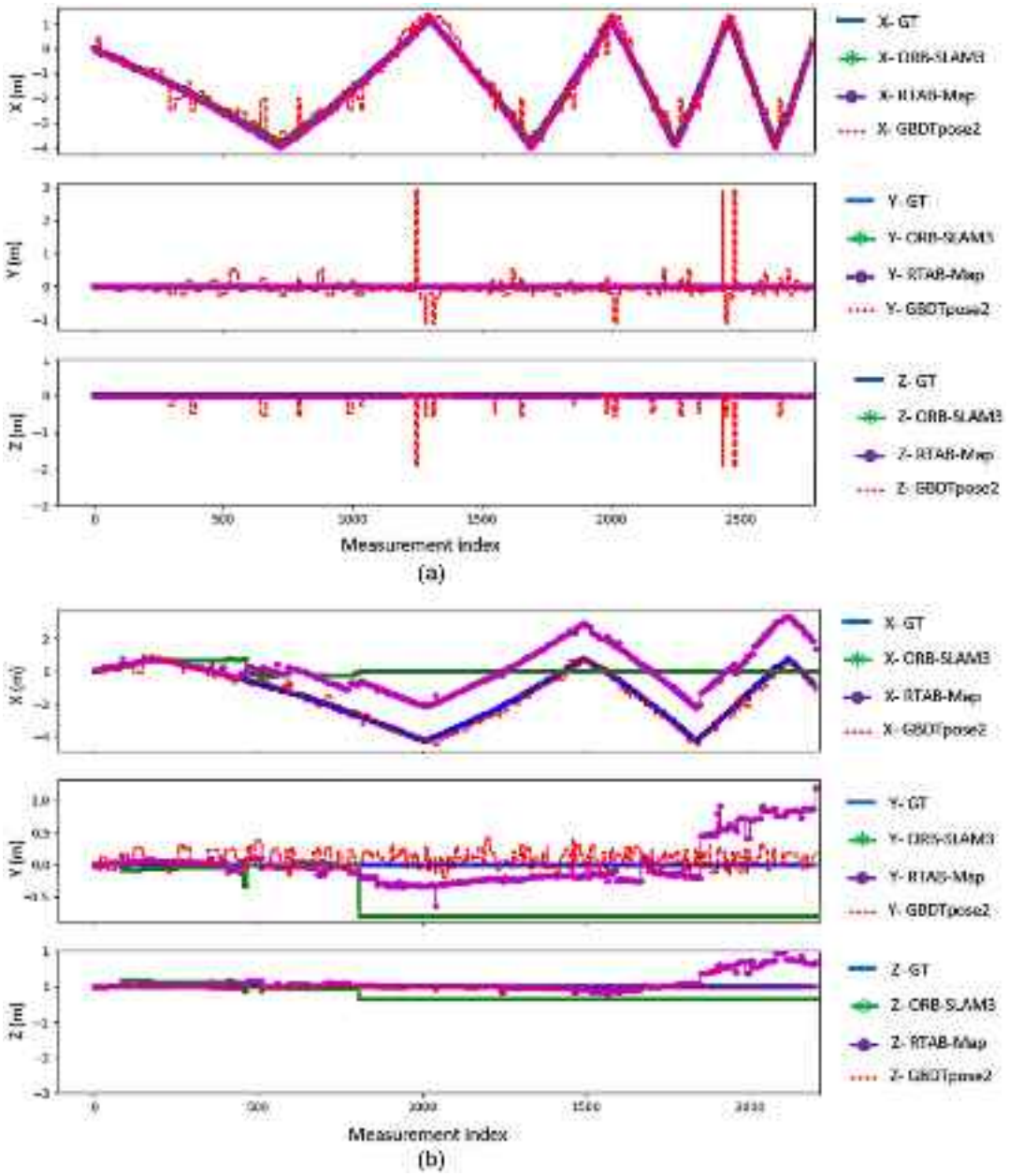

Fig. 8. (a) Position estimates for pure lateral translation at 2.5 m away from the GBDT; and (b) position estimates for lateral translation with rotations ranging $-3°$ to $+3°$ at 2.5 m away from the GBDT.

test are more uniformly spread than for the first. This result occurs because the rotation of the $\mathbb{CA}$ provides variations that allow the HOG similarity metric to work better even around the central regions of the bridge. Moreover, Figs. 6(a) and 7(a) demonstrate that the poses do not drift. This result can be explained by referring to Eqs. (3)–(5), where the estimated UAV's camera relative pose (i.e., $\Delta \boldsymbol{X}_{Ci}^{\mathbb{CA}}$) does not depend on any prior information. Furthermore, if the right VC is selected, then an accurate estimate of $\Delta \boldsymbol{X}_{Ci}^{\mathbb{CA}}$ (within centimeter error range) can be achieved, while being drift-free.

The last series of numerical tests is designed to compare GBDTpose2 against ORB-SLAM3 and RTAB-Map. These tests consist of (i) pure lateral translation at a distance of 2.5 m from the GBDT; and (ii) lateral translations combined with rotations ranging from $-3°$ to $+3°$ along the roll, pitch, and yaw axes. The code to execute ORB-SLAM3 was obtained from its GitHub repository (Tardos 2021), while RTAB-Map (which is available as a ROS package) was directly run with ROS for these tests. In Figs. 8(a and b) the position estimates corresponding to the ground truth, ORB-SLAM3, RTAB-Map, and GBDTpose2 are shown in blue, green, purple, and red, respectively. [In Fig. 8(a), the position estimates align closely with the ground truth, making discrepancies barely noticeable.] Because ORB-SLAM3 and RTAB-Map set their origins to the initial location of the camera, all position estimates shown in Figs. 8(a and b) are with respect to the initial location of $\mathbb{CA}$ during simulation.

Both ORB-SLAM3 and RTAB-Map provide smooth and drift-free pose estimates as seen in the plot of Fig. 8(a) and Table 1. However, when the camera is rotated, ORB-SLAM3 fails to keep track of the camera, while RTAB-Map drifts, as seen in Fig. 8(b)

Table 1. Simulation results (Scenario 1): pure lateral translations without rotations

	CEP			RMSE		
Algorithm	X (m)	Y (m)	Z (m)	X (m)	Y (m)	Z (m)
ORB-SLAM3	**0.038**	**0.002**	**0.008**	0.058	0.012	**0.013**
RTAB-Map	0.040	0.005	0.013	**0.054**	**0.007**	**0.013**
GBDTpose2	0.204	0.106	0.095	0.376	0.336	0.219

Note: Results for comparison of GBDTpose2 against ORB-SLAM3 and RTAB-Map in simulation under pure translations. Bold values signifies the lowest error for each column ("X", "Y", "Z") of the competing localization algorithms. Smaller error is better.

Table 2. Simulation results (Scenario 2): lateral translations with rotations ranging $-3°$ to $+3°$

	CEP			RMSE		
Algorithm	X (m)	Y (m)	Z (m)	X (m)	Y (m)	Z (m)
ORB-SLAM3	1.274	0.794	0.341	2.096	0.633	0.278
RTAB-Map	2.064	0.173	0.039	1.863	0.337	0.272
GBDTpose2	**0.156**	**0.105**	**0.008**	**0.246**	**0.158**	**0.010**

Note: Results for comparison of GBDTpose2 against ORB-SLAM3 and RTAB-Map in simulation under combined lateral translations and rotations. Bold values signifies the lowest error for each column ("X", "Y", "Z") of the competing localization algorithms. Smaller error is better.

and Table 2 (which shows the CEP and RMSE errors). On the other hand, the plot in Fig. 8(b) demonstrates that GBDTpose2 remains robust to rotational perturbations, maintaining accuracy similar to the case of pure lateral motions shown in Fig. 8(a). The failure of ORB-SLAM3 results primarily from the repetitive patterns in the simulation environment. Likewise, RTAB-Map drifts because current poses are composed from the prior poses, which get easily corrupted with repetitive patterns in the environment. Note that although GBDTpose2 does not drift, some spurious pose estimates can occur [i.e., the spikes in Fig. 8(a)] when the wrong VC is selected, which would require additional postprocessing before transferring to the flight controller of a UAV for navigation.

Laboratory Validation

A laboratory specimen of a steel truss bridge located in the Smart Structures Technology Laboratory (SSTL) at the University of Illinois Urbana-Champaign was used for validation of GBDTpose2 [Fig. 9(a)]. A Holybro X500 drone equipped with a Jetson Nano edge device and an Intel RealSense D455 camera [Fig. 9(b)] was used to conduct these tests. The base-station computer used for these tests is also located within 7 m of the UAV. Both the base-station computer and Jetson Nano communicate wirelessly using the Wi-Fi through ROS messages, and Jetson Nano runs GBDTpose2 to

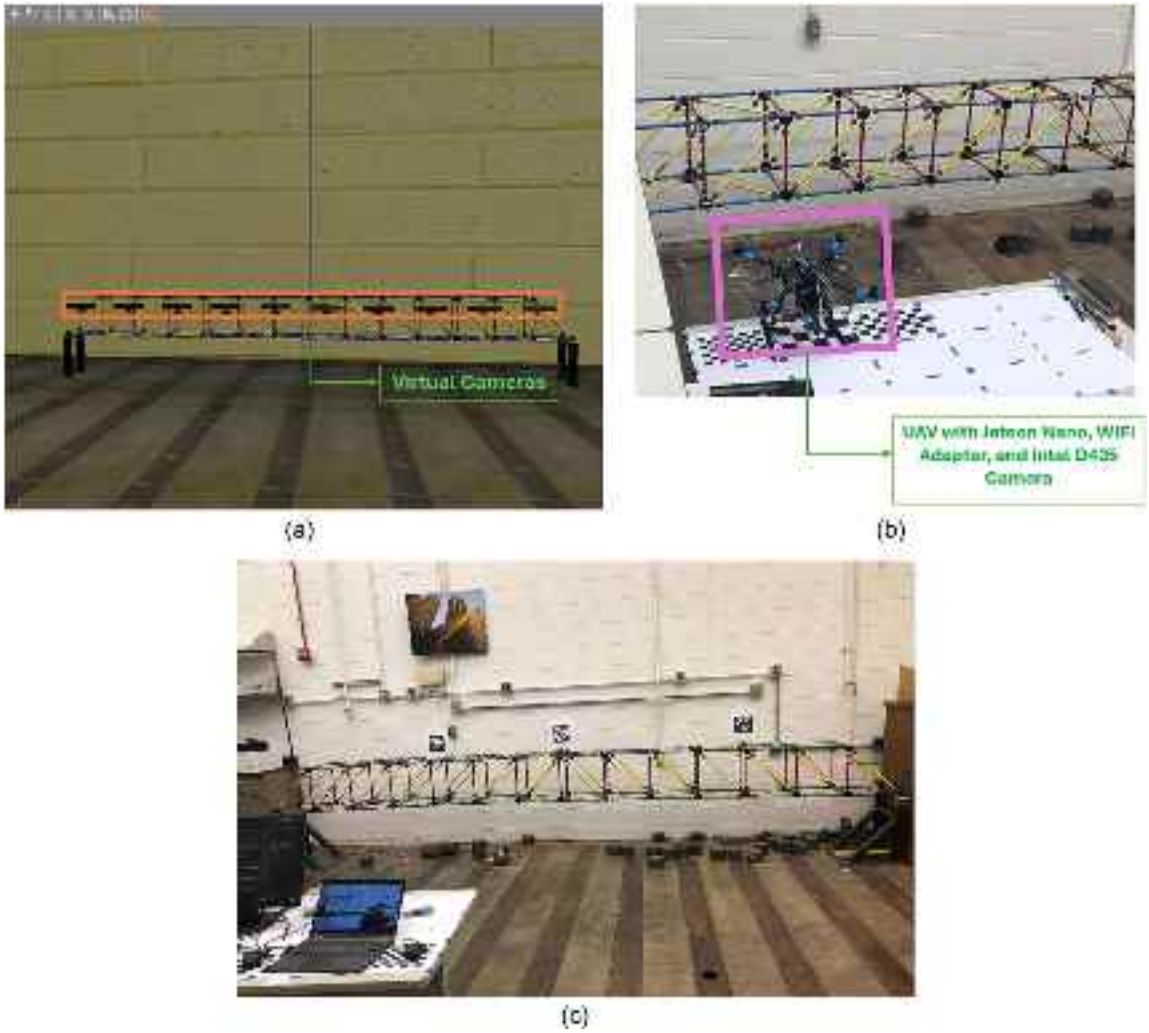

Fig. 9. (a) GBDT and VCs in Gazebo running in the base-station computer in SSTL; (b) UAV with mounted camera, Jetson Nano, and WIFI adapter in SSTL; and (c) AprilTags used for ground truth pose estimates, mounted on SSTL bridge.

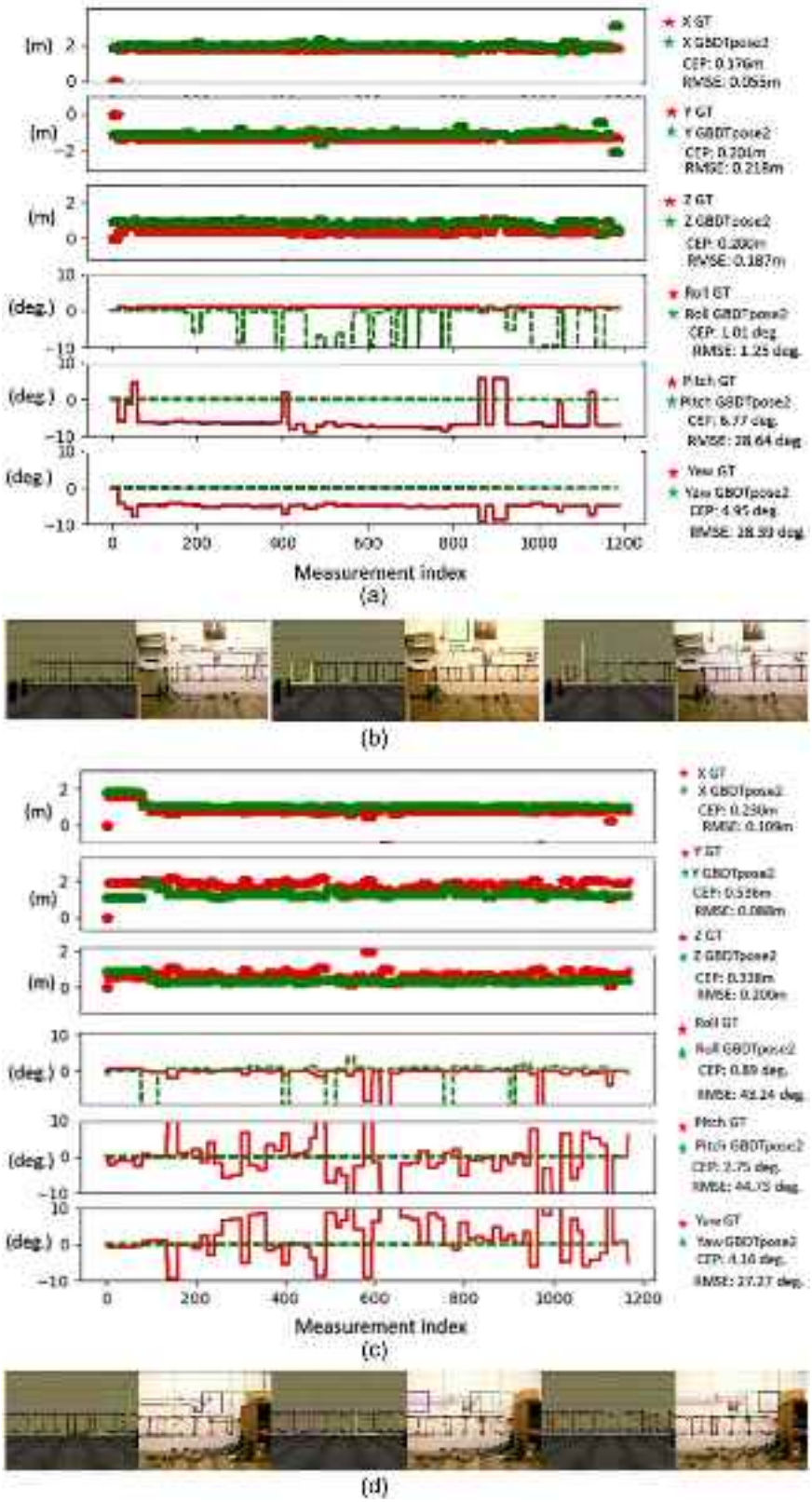

Fig. 10. (a) Pose results for laboratory static test No. 1. The UAV was positioned at the left end of the bridge; (b) Examples of matching image pairs: the left image is from the VC, and the right is from the UAV's camera. GBDTpose2 selects the VC with viewpoint overlap and detects TWRs (colored boxes) for pose estimation; (c) Pose results for laboratory static test No. 2. The UAV was positioned at the right end of the bridge; (d) Corresponding image pairs with the UAV at the right end of the bridge. GBDTpose2 selects VCs sustaining viewpoint overlap with the UAV's camera; (e) Pose results for laboratory lateral motion test with random rotational perturbations; (f) Corresponding image pairs with UAV under lateral motion with small rotational perturbations. For most of the sequences, a VC with viewpoint overlap is selected; (g) Pose results for laboratory aggressive test, with very large random rotational perturbations; and (h) Corresponding image pairs, with UAV camera under aggressive random motion. GBDTpose2 tries to select the right VC but fails in some sequences under very large viewpoint discrepancies.

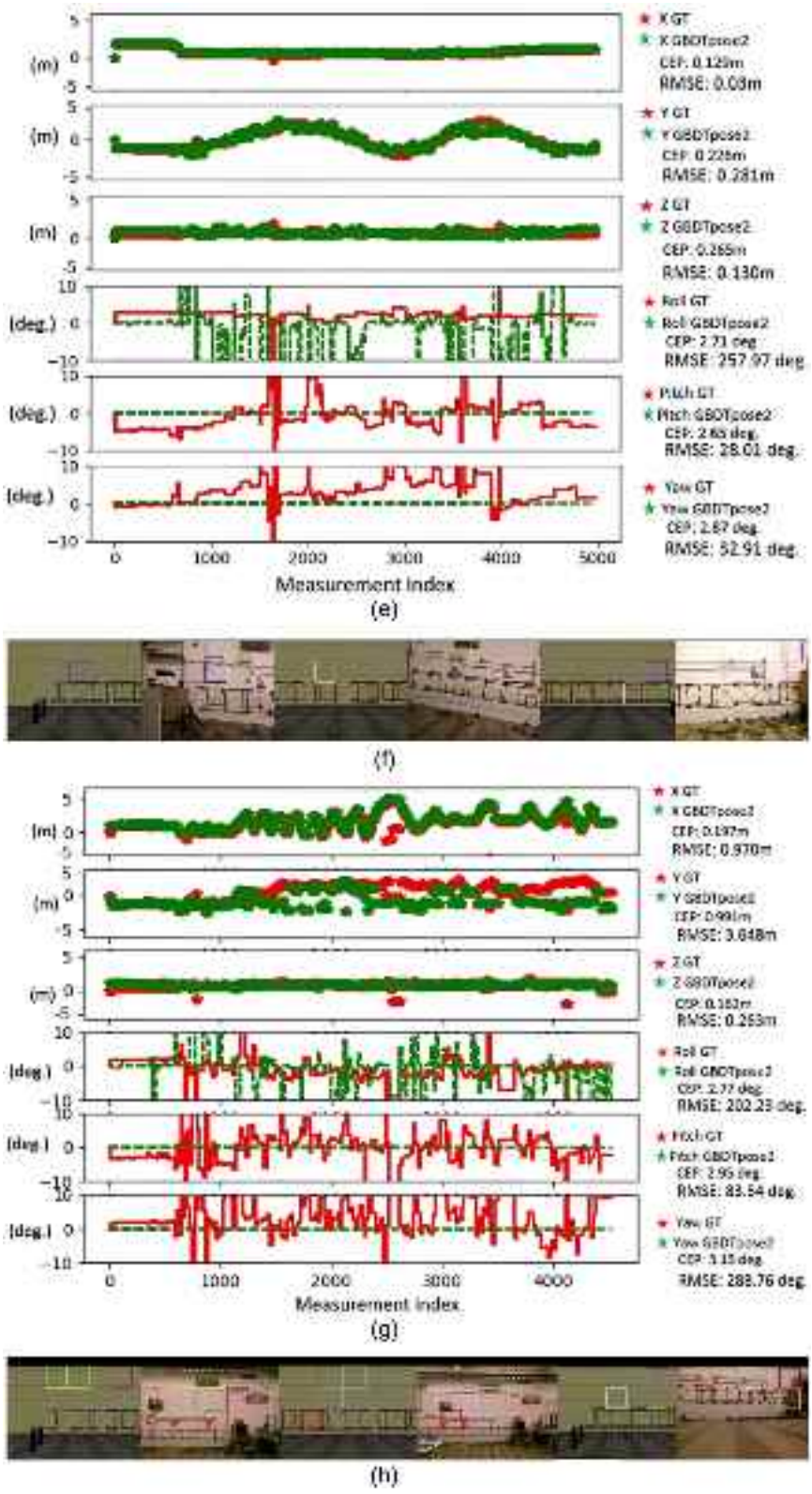

Fig. 10. (*Continued.*)

localize the UAV, based on pose-tagged images published from the base-station computer. Note that for field inspections, a reliable wireless network is required to initialize and maintain the integrity of transmitted data. Alternatively, the VCs can be statically positioned and their data preloaded from the base station onto the edge device. To obtain reference ("ground truth") data for pose estimation, AprilTags were placed on the bridge [as shown in Fig. 9(c)], and the coordinates of their corners were obtained with respect to $\mathbb{G}$ for use with the PnP algorithm.

The UAV was set at locations within 2.5 m from the bridge for the first two tests. These tests consist of a static test [shown in Figs. 10(a–d)] whereby the UAV was fixed at some locations (to mimic hovering motion of a drone) and a lateral motion test [shown in Figs. 10(e–f)], which is similar to that performed for the numerical

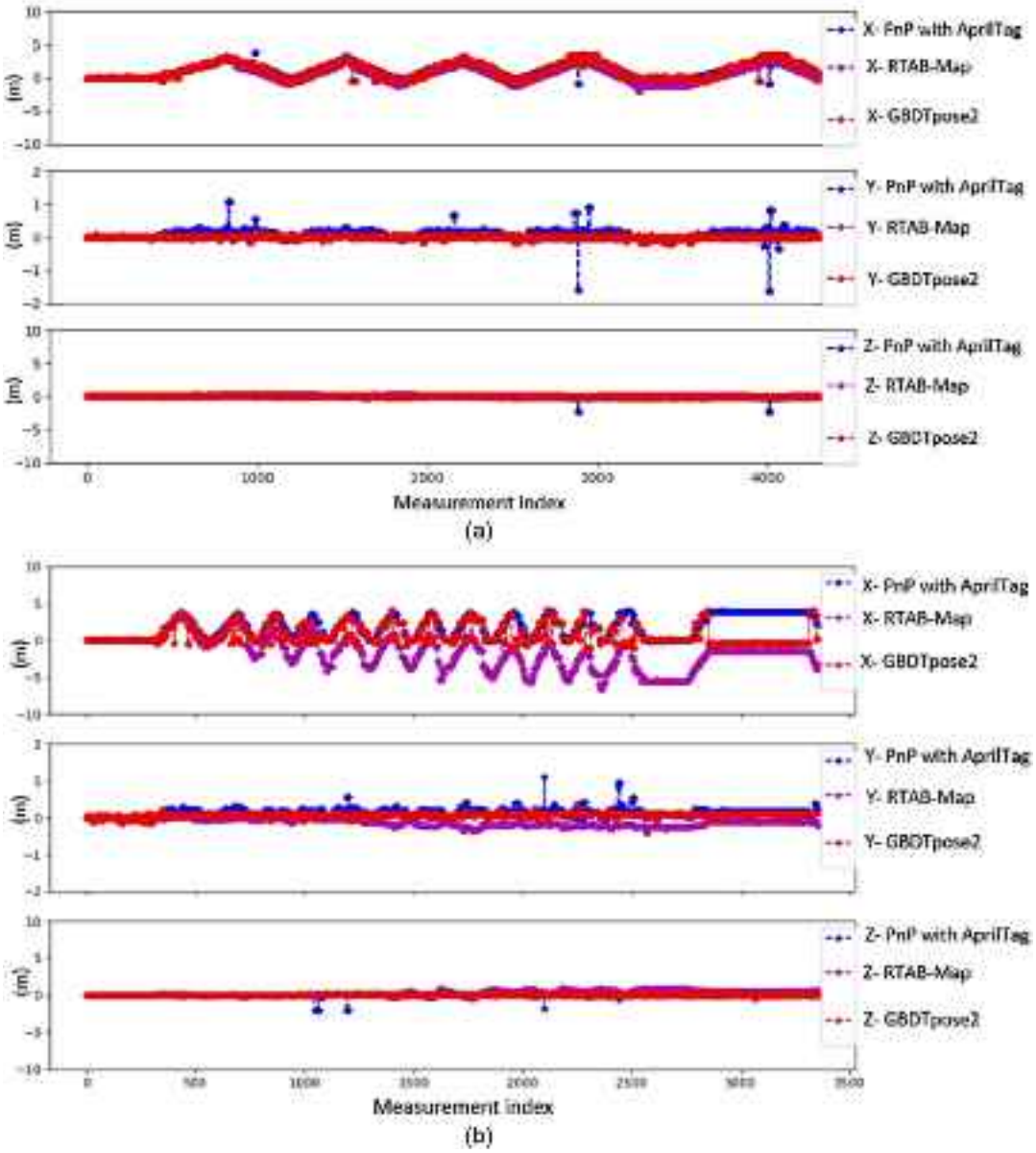

Fig. 11. (a) Position estimates for lateral motion of UAV with very minimal rotations; and (b) position estimates for sideways motion of UAV with rotations ranging −5° to +5° along the yaw axis.

validation of GBDTpose2. Finally, an aggressive test [shown in Figs. 10(g–h)] was conducted whereby the UAV was moved very rapidly away from bridge and under large rotations.

For all the tests reported in Figs. 10(a–h), the position estimates of the UAV show subcentimeter-level accuracy and do not drift; i.e., errors do not accumulate over time. Even for the aggressive test case [shown in Fig. 10(g–h)], GBDTpose2 can still provide the pose of the UAV, once a VC with an overlapping viewpoint to the UAV is chosen. However, the attitude estimates (roll, pitch, and yaw angles) contain many outliers, which is partly attributed to the errors in the ground truth measurements and wrong VC selection by GBDTpose2.

GBDTpose2 was also compared against RTAB-Map in the laboratory environment. ORB-SLAM3 was not used for evaluation because it was unstable and failed to keep track of the UAV's position. Fig. 11(a) and Table 3 shows the results of the position estimates when the UAV was moved laterally and with minimal rotations, while Fig. 11(b) and Table 4 show the case whereby random rotations (within −5° to +5° along the yaw axis) were introduced. These results are similar to those obtained in simulation [i.e. Figs. 8(a and b)], although RTAB-Map performed better for these tests than those shown in Figs. 8(a and b), due to the diverse patterns in the laboratory environment.

While RTAB-Map could provide accurate position estimates, it tends to drift after extended periods of operation [as seen in Fig. 11(b)], or when errors due to feature matching and tracking

Table 3. Results (Scenario 1): lateral translations with small rotations

	CEP			RMSE		
Algorithm	*X* (m)	*Y* (m)	*Z* (m)	*X* (m)	*Y* (m)	*Z* (m)
RTAB-Map	0.760	0.131	0.076	0.749	**0.184**	0.140
GBDTpose2	**0.169**	**0.129**	**0.021**	**0.493**	0.186	**0.114**

Note: Results for comparison of GBDTpose2 against RTAB-Map under minimal rotations. Bold values signifies the lowest error for each column ("X", "Y", "Z") of the competing localization algorithms. Smaller error is better.

Table 4. Results (Scenario 1): lateral translations with rotations ranging −5° to +5°

	CEP			RMSE		
Algorithm	*X* (m)	*Y* (m)	*Z* (m)	*X* (m)	*Y* (m)	*Z* (m)
RTAB-Map	3.657	0.284	0.314	3.940	0.304	0.460
GBDTpose2	**0.306**	**0.085**	**0.048**	**2.045**	**0.131**	**0.230**

Note: Results for comparison of GBDTpose2 against RTAB-Map under minimal rotations. Bold values signifies the lowest error for each column ("X", "Y", "Z") of the competing localization algorithms. Smaller error is better.

occur. On the other hand, GBDTpose2 is drift-free, but its position estimates are not smooth owing to selection of the wrong VC. Occasionally, the impact of suboptimal VC selection by GBDTpose2 (its most significant limitation) can persist over an extended duration [as shown at the end of the X-GBDTpose2 plot in Fig. 11(b)]. This phenomenon (of spurious pose estimates) can affect localization performance should the UAV hover for a prolonged period at a location whereby the HOG similarity metric fails. A similar situation can be found in the plots for Figs. 10(c–d) whereby GBDTpose2 struggled to select the VC with sufficient viewpoint overlap. Note that the HOG similarity metric fails in these scenarios because the feature representations obtained for the VC images differ substantially from the features observed by the UAV's camera, resulting in ambiguities that lead to selecting the wrong VC. In outdoor environments with moving objects and frequent occlusions, both HOG-based matching and ICP may become less reliable due to inconsistencies between the digital and physical twins. Moreover, for larger environments, more TWRs may be needed to obtain enough point cloud data for accurate pose estimation, which increases computational load and can slow down GBDTpose2 on edge devices. As an alternative, ICP can be run on a base-station computer, with the calculated poses sent to the UAV through the edge device. However, this setup may introduce latency, particularly in unstable network conditions. Future studies will focus on optimizing image spatial resolution alongside the HOG feature representation to improve performance. Additionally, the optimization of VC poses will be explored using the prior pose of the UAV, transmitted from the edge device. Nevertheless, GBDTpose2 does automatically correct for these errors once the UAV changes its location or rotates, as demonstrated in Figs. 6(a), 7(a), 8(b), 10(e), and 11(a and b).

Conclusion

Autonomous navigation of UAVs is crucial for automation of regular inspection of civil infrastructure to ensure safety and functionality. For autonomous navigation to be achieved, a drift-free and accurate pose estimate of the UAV is required. GPS can provide for drift-free localization of a UAV, but in typical civil infrastructure environments, GPS suffers interference and multipath effects, which impairs availability and accuracy. Previous studies have attempted to use virtual environments to localize a robot (also applicable for UAVs) in a drift-free fashion. However, ambiguities result from feature correspondences of local patterns between the real and virtual image pairs, and the lack of full coverage of the navigation field has not yet been resolved. This paper presented GBDTpose2, whose goal is to provide a drift-free and accurate pose estimate of a UAV using a digital twin approach. To this end, GBDTpose2 employs a GBDT alongside multiple VCs to localize a UAV by matching global textural and geometric patterns observed by the UAV's onboard camera and the VCs. GBDTpose2 also enabled a bidirectional communication between the UAV and the VCs that can allow for reconfiguration of the VCs to correct for errors in the pose estimates.

Numerical and laboratory experiments have been conducted to validate GBDTpose2. These tests include lateral translation with and without rotational perturbations (to mimic an actual UAV in motion) and static tests (whereby the drone was held stationary, to mimic hovering motion of a UAV). Results show that GBDTpose2 can not only enable drift-free self-localization of a UAV, but also provide centimeter-level position estimates of the UAV, together with an accurate attitude estimate, if a VC sustaining viewpoint overlap with the UAV is chosen for relative pose estimation. If the wrong VC is selected, spurious pose estimates results, but GBDTpose2 can automatically correct for these errors when a VC with adequate viewpoint overlap is chosen. In comparison to ORB-SLAM3 and RTAB-Map, GBDTpose2 does not drift, but the frame-to-frame pose estimates are not as smooth, which would require additional postprocessing before use for autonomous UAV navigation. Future studies will develop sophisticated techniques for replanning the poses of all VCs and selecting the VC with highest viewpoint overlap with the UAV to reduce the pose errors. Additionally, the processing speed of GBDTpose2 will be enhanced to support the use of more VCs in the base station, leading to more accurate pose estimates for seamless autonomous UAV navigation in GPS-denied environments. These improvements will facilitate the automation of civil infrastructure inspections using UAVs.

Data Availability Statement

Some or all data, models, or code generated or used during the study are proprietary or confidential in nature and may only be provided with restrictions.

Acknowledgments

The authors would like to acknowledge the financial support by the U.S. Army Corps of Engineers (Contract/Purchase Order No. W9132rT-22-2-0014).

Author Contributions

Thomas Matiki: Conceptualization; Formal analysis; Methodology; Validation; Writing – original draft; Writing – review & editing. Yasutaka Narazaki: Investigation; Methodology; Writing – original draft; Writing – review and editing. Girish Chowdhary: Supervision; Writing – review and editing. Billie F. Spencer: Funding acquisition; Investigation; Methodology; Project administration; Resources; Software; Supervision; Writing – original draft; Writing – review and editing.

References

Agdas, D., J. A. Rice, J. R. Martinez, and I. R. Lasa. 2016. "Comparison of visual inspection and structural-health monitoring as bridge condition assessment methods." *J. Perform. Constr. Facil.* 30 (3): 04015049. https://doi.org/10.1061/(ASCE)CF.1943-5509.0000802.

Agnisarman, S., S. Lopes, K. C. Madathil, K. Piratla, and A. Gramopadhye. 2019. "A survey of automation-enabled human-in-the-loop systems for infrastructure visual inspection." *Autom. Constr.* 97 (Jan): 52–76. https://doi.org/10.1016/j.autcon.2018.10.019.

Ali, R., D. Kang, G. Suh, and Y. J. Cha. 2021. "Real-time multiple damage mapping using autonomous UAV and deep faster region-based neural

networks for GPS-denied structures." *Autom. Constr.* 130 (Oct): 103831. https://doi.org/10.1016/j.autcon.2021.103831.
Aljanabi, M. A., Z. M. Hussain, N. A. A. Shnain, and S. F. Lu. 2019. "Design of a hybrid measure for image similarity: A statistical, algebraic, and information-theoretic approach." Supplement, *Eur. J. Remote Sens.* 52 (S4): 2–15. https://doi.org/10.1080/22797254.2019.1628617.
Arshad, S., and G.-W. Kim. 2023. "A robust feature matching strategy for fast and effective visual place recognition in challenging environmental conditions." *Int. J. Control Autom. Syst.* 21 (3): 948–962. https://doi.org/10.1007/s12555-021-0927-x.
Arshad, S., and T.-H. Park. 2024. "SVS-VPR: A semantic visual and spatial information-based hierarchical visual place recognition for autonomous navigation in challenging environmental conditions." *Sensors* 24 (3): 906. https://doi.org/10.3390/s24030906.
Besl, P. J., and N. D. McKay. 1992. "Method for registration of 3-D shapes." In *Sensor fusion IV: Control paradigms and data structures*. Bellingham, WA: International Society for Optics and Photonics.
Bloesch, M., S. Omari, M. Hutter, and R. Siegwart. 2015. "ROVIO: Robust visual inertial odometry using a direct EKF-based approach." In *Proc., IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. New York: IEEE.
Campos, C., R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós. 2021. "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM." *IEEE Trans. Rob.* 37 (6): 1874–1890. https://doi.org/10.1109/TRO.2021.3075644.
Chalom, E., E. Asa, and E. Biton. 2013. "Measuring image similarity: An overview of some useful applications." *IEEE Instrum. Meas. Mag.* 16 (1): 24–28. https://doi.org/10.1109/MIM.2013.6417053.
Chen, H., Z. Luo, J. Zhang, L. Zhou, X. Bai, Z. Hu, C. L. Tai, and L. Quan. 2021. "Learning to match features with seeded graph matching network." In *Proc., IEEE/CVF Int. Conf. on Computer Vision (ICCV)*. New York: IEEE.
Chowdhary, G., E. N. Johnson, D. Magree, A. Wu, and A. Shein. 2013. "GPS-denied indoor and outdoor monocular vision aided navigation and control of unmanned aircraft." *J. Field Rob.* 30 (3): 415–438. https://doi.org/10.1002/rob.21454.
Conte, G., and P. Doherty. 2009. "Vision-based unmanned aerial vehicle navigation using geo-referenced information." *J. Adv. Signal Process.* 2009 (1): 387308. https://doi.org/10.1155/2009/387308.
Dalal, N., and B. Triggs. 2005. "Histograms of oriented gradients for human detection." In *Proc., IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*. New York: IEEE.
Davison, A. J., I. D. Reid, N. D. Molton, and O. Stasse. 2007. "MonoSLAM: Real-time single camera SLAM." *IEEE Trans. Pattern Anal. Mach. Intell.* 29 (6): 1052–1067. https://doi.org/10.1109/TPAMI.2007.1049.
Doherty, K., D. Fourie, and J. Leonard. 2019. "Multimodal semantic SLAM with probabilistic data association." In *Proc., 2019 Int. Conf. on Robotics and Automation (ICRA)*. New York: IEEE.
Dorafshan, S., and M. Maguire. 2018. "Bridge inspection: Human performance, unmanned aerial systems and automation." *J. Civ. Struct. Health Monit.* 8 (3): 443–476. https://doi.org/10.1007/s13349-018-0285-4.
Douze, M., et al. 2021. "The 2021 image similarity dataset and challenge." Preprint, submitted June 17, 2021. https://arxiv.org/abs/2106.09672.
Drew, S., H. Andersen, X. Du, X. Shen, M. Meghjani, Y. H. Eng, D. Rus, and M. H. Ang. 2017. "Perception, planning, control, and coordination for autonomous vehicles." *Machines* 5 (1): 6. https://doi.org/10.3390/machines5010006.
Duncan, S., T. I. Stewart, M. Oliver, S. Mavoa, D. MacRae, H. M. Badland, and M. J. Duncan. 2013. "Portable global positioning system receivers: Static validity and environmental conditions." *Am. J. Prev. Med.* 44 (2): e19–e29. https://doi.org/10.1016/j.amepre.2012.10.013.
Engel, J., V. Koltun, and D. Cremers. 2017. "Direct sparse odometry." *IEEE Trans. Pattern Anal. Mach. Intell.* 40 (3): 611–625. https://doi.org/10.1109/TPAMI.2017.2658577.
FHWA. 2004. *Federal Highway Administration: National bridge inspection standards*. Washington, DC: FHWA.
FRA (Federal Railroad Administration). 2010. *Federal railroad administration: Bridge safety standards*. Washington, DC: FRA.
Gazebo. 2014a. "Gazebo documentation." Accessed June 20, 2023. https://staging.gazebosim.org/home.
Gazebo. 2014b. "Gazebo plugins in ROS." Accessed June 20, 2023. https://classic.gazebosim.org/tutorials?tut=ros_gzplugins#OpenniKinect.
Guclu, O., and A. Burak. 2017. "Fast and effective loop closure detection to improve SLAM performance." *J. Intell. Rob. Syst.* 93 (3): 495–517.
Hallermann, N., and G. Morgenthal. 2014. "Visual inspection strategies for large bridges using Unmanned Aerial (UAV)." In *Proc., 7th IABMAS, Int. Conf. on Bridge Maintenance, Safety and Management*, 661–667. Boca Raton, FL: CRC Press.
Hoskere, V., Y. Narazaki, and B. F. Spencer Jr. 2022. "Physics-based graphics models in 3D synthetic environments as autonomous vision-based inspection testbeds." *Sensors* 22 (2): 532. https://doi.org/10.3390/s22020532.
Hua, Z., and H. Masatoshi. 2021. "A decision support system for urban agriculture using digital twin: A case study with aquaponics." *IEEE Access* 9 (Feb): 35691–35708.
John, W., and O. Edwin. 2016. "AprilTag 2: Efficient and robust fiducial detection." In *Proc., IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. New York: IEEE.
Khaloo, A., D. Lattanzi, K. Cunningham, R. Dell'Andrea, and M. Riley. 2017. "Unmanned aerial vehicle inspection of the Placer River trail bridge through image-based 3D modelling." *Struct. Infrastruct. Eng.* 14 (1): 124–136. https://doi.org/10.1080/15732479.2017.1330891.
Klein, G., and D. Murray. 2007. "Parallel tracking and mapping for small AR workspaces." *IEEE Trans. Pattern Anal. Mach. Intell.* 29 (6): 1052–1067.
Kwak, J., and Y. Sung. 2018. "Autonomous UAV flight control for GPS-based navigation." *IEEE Access* 6 (Jul): 37947–37955. https://doi.org/10.1109/ACCESS.2018.2854712.
Labbé, M., and F. Michaud. 2024. "RTAB-map as an open-source lidar and visual SLAM library for large-scale and long-term online operation." *J. Field Rob.* 24 (2): 416–446.
Lai, C. Q., and S. S. Teoh. 2014. "A review on pedestrian detection techniques based on Histogram of Oriented gradient feature." In *Proc., IEEE Student Conf. on Research and Development*. New York: IEEE.
Levine, N. M., and B. F. Spencer Jr. 2022. "Post-earthquake building evaluation using UAVs: A BIM-based digital twin framework." *Sensors* 22 (3): 873. https://doi.org/10.3390/s22030873.
Li, P., R. Wang, Y. Wang, and W. Tao. 2020. "Evaluation of the ICP algorithm in 3D point cloud registration." *IEEE Access* 8 (Apr): 68030–68048. https://doi.org/10.1109/ACCESS.2020.2986470.
Lin, H.-Y., and C.-H. He. 2021. "Mobile robot self-localization using omnidirectional vision with feature matching from real and virtual spaces." *Appl. Sci.* 11 (8): 3360. https://doi.org/10.3390/app11083360.
Lindenberger, P., P.-E. Sarlin, and M. Pollefeys. 2023. "LightGlue: Local feature matching at light speed." In *Proc., IEEE/CVF Int. Conf. on Computer Vision (ICCV)*. New York: IEEE.
Lowe, D. G. 2004. "Distinctive image features from scale-invariant keypoints." *Int. J. Comput. Vis.* 60 (2): 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94.
Lu, X. X. 2009. "A review of solutions for perspective-n-point problem in camera pose estimation." *J. Phys.* 1087 (5): 052009. https://doi.org/10.1088/1742-6596/1087/5/052009.
Ma, J., X. Jiang, A. Fan, J. Jiang, and J. Yan. 2020. "Image matching from handcrafted to deep features: A survey." *Int. J. Comput. Vis.* 129 (1): 23–79. https://doi.org/10.1007/s11263-020-01359-2.
Malekloo, A., E. Ozer, M. AlHamaydeh, and M. Girolami. 2022. "Machine learning and structural health monitoring overview with emerging technology and high-dimensional data source highlights." *Struct. Health Monit.* 21 (4): 1906–1955. https://doi.org/10.1177/14759217211036880.
Marchel, Ł., C. Spech, and M. Specht. 2020. "Testing the accuracy of the modified ICP algorithm with multimodal weighting factors." *Energies* 13 (22): 5939. https://doi.org/10.3390/en13225939.
Matiki, T., Y. Narazaki, G. Chowdhary, and B. F. Spencer Jr. 2024. "A graphics-based digital twin framework for accurate localization of UAVs in GPS-denied environments." *Adv. Struct. Eng.* 27 (16): 2879–2900. https://doi.org/10.1177/13694332241291256.
Miller, D. 2021. "Advance your robot autonomy with ROS 2 and unity." Accessed July 20, 2023. https://unity.com/.

Molinar, J., J. Garcia, R. Robles, C. Mendoza, J. Reyes, A. Choudhuri, and A. Flores-Abad. 2023. "Digitally assisted development of a space robotic arm for satellite applications." In *Proc., ASCEND 2023*. Reston, VA: American Institute of Aeronautics and Astronautics.

Morse. 2009. "ROS and MORSE tutorial." Accessed July 20, 2023. https://www.openrobots.org/morse/doc/stable/user/beginner_tutorials/ros_tutorial.html.

Mur-Artal, R., J. M. M. Montiel, and J. D. Tardos. 2015. "ORB-SLAM: A versatile and accurate monocular SLAM system." *IEEE Trans. Rob.* 31 (5): 1147–1163. https://doi.org/10.1109/TRO.2015.2463671.

Narazaki, Y., F. Gomez, V. Hoskere, M. D. Smith, and B. F. Spencer. 2020. "Efficient development of vision-based dense three-dimensional displacement measurement algorithms using physics-based graphics models." *Struct. Health Monit.* 20 (4): 1841–1863. https://doi.org/10.1177/1475921720939522.

Narazaki, Y., V. Hoskere, G. Chowdhary, and B. F. Spencer Jr. 2022. "Vision-based navigation planning for autonomous post-earthquake inspection of reinforced concrete railway viaducts using unmanned aerial vehicles." *Autom. Constr.* 137 (May): 104214. https://doi.org/10.1016/j.autcon.2022.104214.

Narazaki, Y., V. Hoskere, K. Yoshida, B. F. Spencer, and Y. Fujino. 2021. "Synthetic environments for vision-based structural condition assessment of Japanese high-speed railway viaducts." *Mech. Syst. Sig. Process.* 160 (Nov): 107850. https://doi.org/10.1016/j.ymssp.2021.107850.

Narazaki, Y., W. Pang, G. Wang, and W. Chai. 2023. "Unsupervised domain adaptation approach for vision-based semantic understanding of bridge inspection scenes without manual annotations." *J. Bridge Eng.* 29 (2): 04023118. https://doi.org/10.1061/JBENF2.BEENG-6490.

Newcombe, R., S. Lovegrove, and A. Davison. 2011. "DTAM: Dense tracking and mapping in real-time." In *Proc., Int. Conf. on Computer Vision*, 2320–2327. New York: IEEE.

Pinheiro, P. O. 2018. "Unsupervised domain adaptation with similarity learning." In *Proc., IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. New York: IEEE.

ROS (Robot Operating System). 2018a. "Robot operating system: ROS documentation." Accessed July 10, 2023. https://docs.ros.org/.

ROS (Robot Operating System). 2018b. "Running ROS across multiple machines." https://wiki.ros.org/ROS/Tutorials/MultipleMachines.

Rosten, E., and T. Drummond. 2005. "Fusing points and lines for high performance tracking." In *Proc., 10th IEEE Int. Conf. on Computer Vision*. New York: IEEE.

Sarlin, P.-E., D. DeTone, T. Malisiewicz, and A. Rabinovich. 2020. "SuperGlue: Learning feature matching with graph neural networks." In *Proc., IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*. New York: IEEE.

Schmitt, M., M. Rous, A. Matsikis, and K.-F. Kraiss. 1999. "Vision-based self-localization of a mobile robot using a virtual environment." In *Proc., 1999 IEEE Int. Conf. on Robotics & Automation*. New York: IEEE.

Schubert, S., P. Neubert, S. Garg, M. Milford, and T. Fischer. 2023. "Visual place recognition." *IEEE Robot. Autom. Mag.* 2–16.

Szrek, J., P. Trybała, M. Goralczyk, A. Michalak, B. Ziętek, and R. Zimroz. 2021. "Accuracy evaluation of selected mobile inspection robot localization techniques in a GNSS-denied environment." *Sensors* 21 (1): 141. https://doi.org/10.3390/s21010141.

Tardos, J. D. 2021. "ORB-SLAM3." Accessed September 23, 2023. https://github.com/UZ-SLAMLab/ORB_SLAM3.

Tsintotas, K. A., L. Bampis, and A. Gasteratos. 2022. "The revisiting problem in simultaneous localization and mapping: A survey on visual loop closure detection." *IEEE Trans. Intell. Transp. Syst.* 23 (11): 19929–19953. https://doi.org/10.1109/TITS.2022.3175656.

Ulas, C., and T. Hakan. 2013. "A fast and robust feature-based scan-matching method in 3D SLAM and the effect of sampling strategies." *Int. J. Adv. Rob. Syst.* 10 (11): 396. https://doi.org/10.5772/56964.

Ultralytics. 2024. "Ultralytics YOLOv8 docs." Accessed August 20, 2024. https://docs.ultralytics.com/.

Vanegas, F., K. J. Gaston, J. Roberts, and F. Gonzalez. 2019. "A framework for UAV navigation and exploration in GPS-denied environments." In *Proc., IEEE Aerospace Conf.* New York: IEEE.

Wang, F., and Z. Zhao. 2017. "A survey of iterative closest point algorithm." In *Proc., 2017 Chinese Automation Congress (CAC)*. New York: IEEE.

Wang, S., C. Rodgers, G. Zhai, T. N. Matiki, B. Welsh, A. Najafi, J. Wang, Y. Narazaki, V. Hoskere, and B. F. Spencer Jr. 2022. "A graphics-based digital twin framework for computer vision-based post-earthquake structural inspection and evaluation using unmanned aerial vehicles." *J. Infrastruct. Intell. Resilience* 1 (1): 100003. https://doi.org/10.1016/j.iintel.2022.100003.

Yao, G., A. Yilmaz, F. Meng, and L. Zhang. 2021. "Review of wide-baseline stereo image matching based on deep learning." *Remote Sens.* 13 (16): 3247. https://doi.org/10.3390/rs13163247.

Yue, Y., X. Wang, and Z. Zhan. 2023. "Single-point least square matching embedded method for improving visual SLAM." *IEEE Sens. J.* 23 (14): 16176–16188. https://doi.org/10.1109/JSEN.2023.3277487.

Zhang, G., and L.-T. Hsu. 2019. "A new path planning algorithm using a GNSS localization error map for UAVs in an urban area." *J. Intell. Rob. Syst.* 94 (1): 219–235. https://doi.org/10.1007/s10846-018-0894-5.

Zhang, H., H. Xu, X. Tian, J. Jiang, and J. Ma. 2021a. "Image fusion meets deep learning: A survey and perspective." *Inf. Fusion* 76 (Dec): 323–336. https://doi.org/10.1016/j.inffus.2021.06.008.

Zhang, J., Y. Yao, and B. Deng. 2021b. "Fast and robust iterative closest point." *IEEE Trans. Pattern Anal. Mach. Intell.* 44 (7): 3450–3466.

Zhang, L., and X. Gao. 2022. "Transfer adaptation learning: A decade survey." *IEEE Trans. Neural Networks Learn. Syst.* 35 (1): 23–44. https://doi.org/10.1109/TNNLS.2022.3183326.

Zhang, R., A. Ouyang, and Z. Li. 2022. "Automatic UAV inspection of tunnel infrastructure in GPS-denied underground environment." In *Proc., European Workshop on Structural Health Monitoring*. Cham, Switzerland: Springer.

Zhou, F., L. Zhang, C. Deng, and X. Fan. 2021. "Improved point-line feature based visual SLAM method for complex environments." *Sensors* 21 (13): 4604. https://doi.org/10.3390/s21134604.