Automation in Construction 127 (2021) 103723

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

A novel methodology for the path alignment of visual SLAM in indoor construction inspection

Tao Lu [a,*], Sonja Tervola [b], Xiaoshu Lü [a,c,d,e], Charles J. Kibert [f], Qunli Zhang [d], Tong Li [g], Zhitong Yao [h]

[a] Department of Electrical Engineering and Energy Technology, University of Vaasa, P.O.Box 700, FIN-65101 Vaasa, Finland
[b] Department of Computer Science, Aalto University, P.O.Box 15400, FIN-00076 Aalto, Finland
[c] College of Construction Engineering, Jilin University, Changchun 130026, China
[d] Beijing Key Lab of Heating, Gas Supply, Ventilating and Air Conditioning Engineering, Beijing University of Civil Engineering and Architecture, Beijing 100044, China
[e] Department of Civil Engineering, Aalto University, P.O.Box. 11000, FIN-02130 Aalto, Finland
[f] Powell Center for Construction & Environment, University of Florida, P.O. Box 115703, Gainesville, FL 32611-5703, USA
[g] Changchun Tiancheng Technology Development Co., Ltd., Changchun 130061, China
[h] College of Materials and Environmental Engineering, Hangzhou Dianzi University, Hangzhou 310018, China

ARTICLE INFO

Keywords:
Principal component analysis
Simultaneous localization and mapping
Path alignment
Affine transformation
Path distortion
2-Point Scheme

ABSTRACT

Path alignment is the process of mapping an indoor construction inspection path reconstructed by a visual SLAM system onto a 2D map with user interaction required to pinpoint at least two common tie points. In practice, more points are often needed due to path distortions and linear transformations, potentially resulting in reduced productivity. This paper proposes a methodology that combines two novel algorithms for the path alignment: (1) PCA_STAN_ALGO applies principal component analysis to remove path distortions caused by the xz plane of a camera coordinate system not being parallel to the floor plane; and (2) GRPX_TRANS utilizes a graphical user interface to facilitate the path alignment. The proposed methodology enables the users to utilize just two tie points for successful path alignment. An experimental study showed that applying both PCA_STAN_ALGO and GRPX_TRANS saved about 50% in time compared to using only GRPX_TRANS, a result of needing minimal moving points.

1. Introduction

Video technology is widely used on construction sites for labor and equipment tracking [1–5], schedule and progress monitoring [6,7], and safety management applications [8–11].

A more innovative and inexpensive method is to combine video, AI (Artificial Intelligence), and visual SLAM (simultaneous localization and mapping) [12–25] techniques to inspect building construction processes in terms of safety, quality, regulation, and progress. In general, some digital IT companies provide services where inspectors visit buildings on a weekly basis with helmet cameras. The extensive use of 360-degree cameras can improve the inspection outcome and vSLAM (visual SLAM) performance. A typical inspection video usually covers 3–6 flats and public sections (corridors and stairs) from the same floor and the same unit. Video processing typically includes: 1) a vSLAM processing to get the location information for each frame; 2) an inspection process using machine learning and interaction by inspectors. All issues concerning safety, quality, and progress are annotated in the relevant frames in the video stream. Fig. 1 shows such a frame, where on the left a red bounding box indicates that the ladder does not meet Finnish construction safety standards. On the right, the location of the camera is depicted on the floor (plan) map.

Since an issue concerning safety, quality, regulation, or progress could occur at any moment in a video, sufficient and accurate location information must be provided preferably for every frame. Thus, vSLAM plays a very important role in the video processing due to its low cost and good accuracy [13].

1.1. Visual SLAM systems and indoor construction inspection

SLAM is a computational technique applied in mobile robot for constructing or updating a map of an unknown environment while

* Corresponding author.
E-mail address: tao.lu@uwasa.fi (T. Lu).

https://doi.org/10.1016/j.autcon.2021.103723
Received 14 June 2020; Received in revised form 10 April 2021; Accepted 15 April 2021
0926-5805/© 2021 Elsevier B.V. All rights reserved.

simultaneously keeping track of the agent's pose within it [14]. It includes two features: mapping (environment representation) and localization (pose estimation). vSLAM systems fall into two categories: indirect and direct. The main difference is that indirect methods implement SLAM in two consecutive steps, namely, extracting a sparse set of key points from the current frame and matching them across multiple frames, and subsequently solving the problems of camera poses and feature depths by minimizing the geometric reprojection errors between feature pairs. In contrast, direct methods tackle the SLAM problem in a single step by minimizing the photometric error of imposing color or brightness consistency across all images [15]. In practice, the brightness of a 3D (three-dimensional) point observed in multiple images is not constant. Therefore, indirect methods are more robust than direct methods in addressing brightness inconsistencies among consecutive frames because of the improvement in modern feature descriptors [15]. However, in a low-texture environment, direct methods are more robust since they can utilize much more image information. Rolling shutter cameras scan images sequentially, from one side of the sensor to the other, line by line. This leads to image distortions when the camera is moving [16], i.e., rolling shutter effect. As demonstrated in [17], direct methods are very sensitive to rolling shutter effects. Nowadays, ORB-SLAM [18] and DSO (Direct Sparse Odometry) [17] constitute the state-of-the-art of vSLAM systems.

ORB-SLAM is an indirect and sparse SLAM system that uses ORB features [19] for tracking, mapping, relocalization, and loop closing. It includes three parallel threads: tracking, local mapping, and loop closing. For tracking, the initial camera pose is estimated by the motion BA (Bundle Adjustment) and further optimized using all matches between the current frame and the local map retrieved using the covisibility graph of keyframes [15]. If the tracking is lost, a global relocalization is performed. This thread also decides if a new keyframe needs to be inserted based on some criteria. In the local mapping, whenever a new keyframe is inserted, the covisibility graph is updated. The local BA optimizes the currently processed keyframe, all connected in the covisibility graph, and all the map points seen by those keyframes [15]. The loop closing searches for loops with every new keyframe. If a loop is detected, both sides of the loop are aligned and duplicate points are fused [18].

DSO is a direct and sparse SLAM system that performs a novel sparse point sampling across image area with sufficient intensity gradient [15]. A novel photometric calibration pipeline is proposed to account for brightness inconsistencies. In the front end of the system, each new image is tracked by direct image alignment in a coarse-to-fine manner [17]. A new keyframe is created, if the current frame meets some conditions. The system always keeps some keyframes in the active window and old keyframes are culled out by marginalization [17]. When a new keyframe is inserted to the active window, all model parameters are optimized by minimizing the photometric error in the back end of the system [15]. DSO is not a full SLAM algorithm because it lacks the functions of relocalization, loop detection, and map reuse [17].

Indoor construction inspection has the following characteristics:

- It is conducted inside a building that has typically poor lighting conditions.
- The majority of cameras used for inspection have rolling shutters.
- Inspectors often complete inspection routes in a rush because they have to visit many sites in a day. Sometimes this leads to strong rolling shutter effects in a video. Furthermore, in order to capture more areas to shorten the inspection time, 360-degree cameras dominate. Because 360-degree cameras are capable of capturing the entire surroundings, tracking (if it is lost) can be restored if the SLAM system has ability for relocalization.
- Every indoor inspection route includes many small loops (e.g., in most cases, the inspection route in a room forms a loop).
- A building will be visited for many times, normally, once a week over an extended period of time.

Therefore, a vSLAM system, which assists indoor construction inspection, should be capable of 1) performing relocalization when tracking is lost; 2) supporting 360-degree cameras; 3) detecting loops and closing them; 4) storing a map and reusing it when a building is revisited. In addition, it should also be robust to be able to address brightness inconsistencies and not be overly sensitive to the rolling shutter effects. The ORB-SLAM system seems to be the best choice because it satisfies almost all the demands. However, the current ORB-SLAM open-source system does not support 360-degree cameras. On the other hand, OpenVSLAM released in 2019 by National Institute of Advanced Industrial Science and Technology, Japan is a vSLAM framework with good usability and extensibility [22]. It is an indirect and sparse SLAM system based largely on ORB-SLAM. It also incorporates some well-known SLAM approaches (e.g., ProSLAM [23], UcoSLAM [24]). Compared with ORB-SLAM, OpenVSLAM performs better in accuracy and tracking time [22]. Moreover, OpenVSLAM has two very unique and attractive features: 1) OpenVSLAM supports multiple camera models: perspective, fisheye, and equirectangular. Particularly, it is probably the first open-source vSLAM framework that can accept equirectangular imagery [22]; 2) OpenVSLAM can store and load a priori map, and then localize new images using that map [22]. Most of the vSLAM frameworks cannot store and load maps [22]. Thus, this study concentrates on the paths reconstructed by OpenVSLAM using an equirectangular camera model. However, the methodology discussed in this study is independent of the type of vSLAM system and camera model used.

As pointed out in [25], most SLAM research concentrates on the

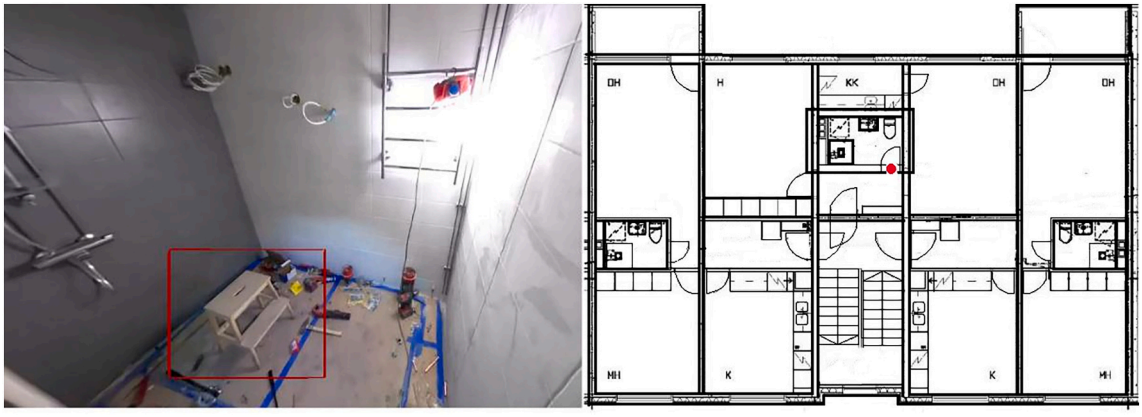


Fig. 1. Left: A processed video frame with a red bounding box annotation pointing out that an unsafe ladder is used. Right: The red dot indicates the current location of the camera on the floor (plan) map.

issues of robust performance, high-level understanding, resource awareness, and task-driven inference. Little attention was paid to the SLAM path alignment with an existing map, particularly for the case of indoor construction inspection (see Fig. 2).

A camera coordinate system is a type of Cartesian coordinate system that uses the optical center of the camera as its origin and the optical axis as the z-axis (see Fig. 2). For a helmet camera, the y-axis of its camera coordinate system points to the ground (Fig. 2). The xz plane of a camera coordinate system is the plane that contains both the x- and z-axis.

1.2. The path alignment of SLAM reconstruction in indoor construction inspection

In this paper, the term "SLAM path" refers to a path of a moving video camera used for construction inspection indoors that is reconstructed by a vSLAM system in 3D space. An indoor construction inspection path is 2D (two-dimensional). The path alignment of a SLAM path is the final and decisive stage, involving a coordinate system transformation from a camera (three-dimensional) to a 2D floor map (Fig. 2). In order to simplify the path alignment, the y coordinates of a SLAM path are ignored and the transformation is simplified as a 2D affine transformation accounting for reflection, rotation, scaling, and translation. In geometry, a reflection is known as a flip. In this study, the reflections over the x- and y-axis are considered, for example, when a point is reflected across the x-axis, the x coordinate remains the same. But the y coordinate is transformed into its opposite sign. This 2D affine transformation can also be regarded as sending the orthographic projection of a 3D path on the xz plane of the coordinate system onto a 2D map. Because rotation and scaling always exist between the two coordinate systems, at least two points and their corresponding locations on a floor map need to be known to determine the affine transformation matrix [26]. Finding out the corresponding location of a point on a floor map is often done manually. This indicates that, if all location points of a SLAM path (2D without y coordinates) appear on a floor map, the user needs to relocate at least two points to finish the affine transformation. In this work, this strategy is called the 2-point scheme. The aim of this study is to validate the 2-point scheme.

1.3. The 2-point scheme and path distortion on the xz plane of a camera coordinate system

Because of the minimum number of points used, the 2-point scheme greatly reduces workload, saves time, and has low demand for skill and experience. In addition, it also keeps the shape of a path unchanged since only rotation, scaling, and translation are taken into account. However, there are several theoretical and practical issues that make the 2-point scheme quite difficult:

- In practice, most of the time there are reflections in the two coordinate systems (Fig. 2). If reflections are not treated separately and properly, one more point is needed to solve the affine transformation, which in turn increases risk for path distortion.
- Without a proper graphical user interface (GUI), it is difficult to precisely determine the target locations of two points on a floor map by just watching the video. Furthermore, watching a video to search locations is cumbersome and time-consuming.
- An aligned path on a floor map is the orthographic projection of a SLAM path on the xz plane of the camera coordinate system (Fig. 2) while the indoor construction inspection path is the orthographic projection of the same SLAM path on the floor plane. In practice, the xz plane of a camera coordinate system is not truly parallel to the floor plane, leading to differences in the two projections. This phenomenon is termed as the 3D-to-2D scale drift in this paper, but it should not be confused with the scale drift occurred in monocular SLAM that is the accumulation of small errors in scale estimate caused by the depth scaling ambiguity [27,28]. Therefore, the 3D-to-2D scale drift cannot be overcome using IMU (Inertial Measurement Unit), LiDAR (Light Detection and Ranging), stereo or depth cameras, and it becomes significant when the vertical movement of a camera varies greatly, for instance, climbing the stairs in a building. Chapter 2 will give more information about the 3D-to-2D scale drift. Fig. 3 shows the orthographic projection of a path reconstructed by OpenVSLAM on the xz plane. The video was shot on a building construction site using a Gopro Fusion 360 camera. The true starting point was on the second floor, but the one on the xz plane from OpenVSLAM appears on the stairs. Fixing such 3D-to-2D scale drift requires more (tie) points to move the path for proper alignment. A false (aligned) path could result due to this 3D-to-2D scale drift.

The accuracy requirement of the paths reconstructed by SLAM in indoor construction inspection is generally not as high as for other SLAM applications, such as mobile robots. For the path alignment, it is expected that, on a floor map, 1) the branches of the indoor construction inspection route should be properly located in the target rooms and (public) areas (e.g., stairs), and 2) there is no apparent 3D-to-2D scale drift occurred in the area of the stairs. These will provide sufficient

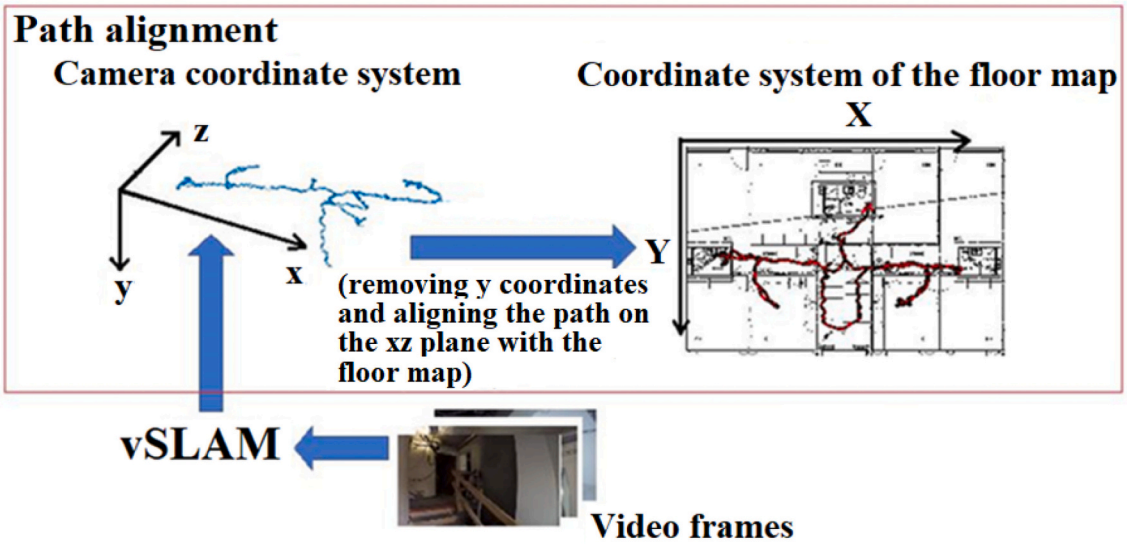


Fig. 2. Illustration of the procedures of the path alignment for a path reconstructed by a vSLAM system.

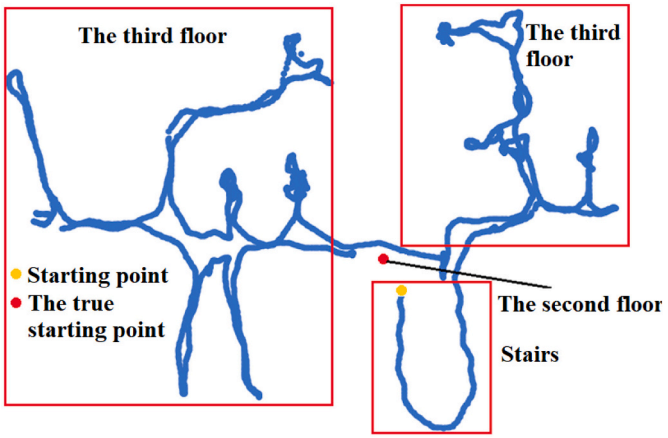


Fig. 3. The orthographic projection (xz plane) of an indoor construction inspection path reconstructed by OpenVSLAM, which experiences the 3D-to-2D scale drift.

accuracy for the customers to observe indoor construction processes when 360-degree cameras are used.

The objective of this study is to address the aforementioned issues and accuracy requirement for the purpose of facilitating the affine transformation so that the users can implement the 2-point scheme even without watching the original video. A GUI based methodology was developed for this purpose. The methodology is comprised of a camera coordinate system standardization algorithm and a GUI algorithm. A SLAM path first goes through a camera coordinate system standardization algorithm, where the camera coordinate system is automatically adjusted to make the xz plane parallel with the floor plane aiming to remove the 3D-to-2D scale drift. Then, by deleting the y coordinates, the path is projected onto the xz plane of the adjusted camera coordinate system. The actual affine transformation that maps the projected path (2D) onto the floor map (2D) is done by a GUI algorithm called Graphical Transformation. In this work, we deal with only the 3D-to-2D scale drift for a SLAM path and ignore local distortions caused by OpenVSLAM.

Graphical Transformation contains a couple of 2D linear transformation algorithms. In Graphical Transformation, reflections are dealt independently by a "single click" so that the users do not need to move extra points. Rotations, scalings and translations are handled by a "Pivots and Moving Point" (PAMP) algorithm. The PAMP algorithm will be discussed later in detail. In short, after performing reflections in the two coordinate systems, based on the size of the floor map, PAMP first estimates the transformation matrix by considering only scaling and translation in order to visualize all points (Fig. 4).

As shown in Fig. 4, the user can easily determine which room a branch of the path belongs to, and rough locations of some feature points (e.g., the points close to the doors). By selecting a point as a pivot (the blue one in Fig. 4) and moving another point continuously, the transformation matrix is simultaneously updated, which moves the rest of the points accordingly (see Fig. 5). All actions were completed without watching a single video.

The camera coordinate system standardization algorithm employs PCA (principal component analysis) for rotating the camera coordinate system of a path to make its xz plane and the floor plane parallel. It is much simpler than the traditional method. The traditional method tries to determine the vector description of the floor in a camera coordinate system first, and then calculates the rotation between the floor plane and the xz plane of the camera coordinate system. Because the vertical movement of the camera is small on a given floor, the vector of the floor can be obtained using the least squares method, namely, the sum of the squares of the distances from path points to the floor plane should be minimized. A 3D rotation is described by an angle rotating around a unit axis. The rotation angle and unit axis between the xz plane and the floor plane can be calculated using the cross product and inner product. Finally, the rotation matrix is estimated by the obtained rotation angle and unit axis. The whole process is complicated, and in 3D space, the orientation of the unit axis and the direction of the rotation angle can be quite confusing (e.g., the cross product is not commutative). On the contrary, PCA is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components [29]. The vertical movement of the camera is small and independent of the horizontal movement (i.e., the path on the floor). If we consider the x, y, and z coordinates of a path as three

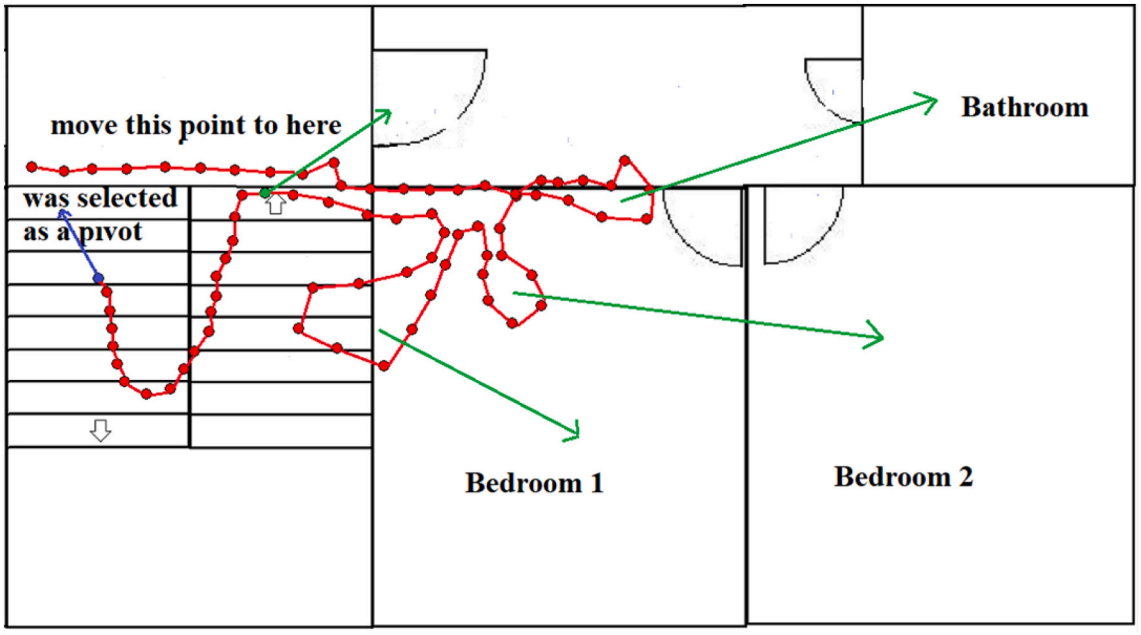


Fig. 4. Illustration of Graphical Transformation: the initial visualization of a path on a floor map, aiming at displaying all location points on the map. Note: The blue point is the first point that was selected and moved to the target location. This also brought the rest of the points to their current locations. The green point is the second point that is going to be moved to the desired location to align the path with the flat.

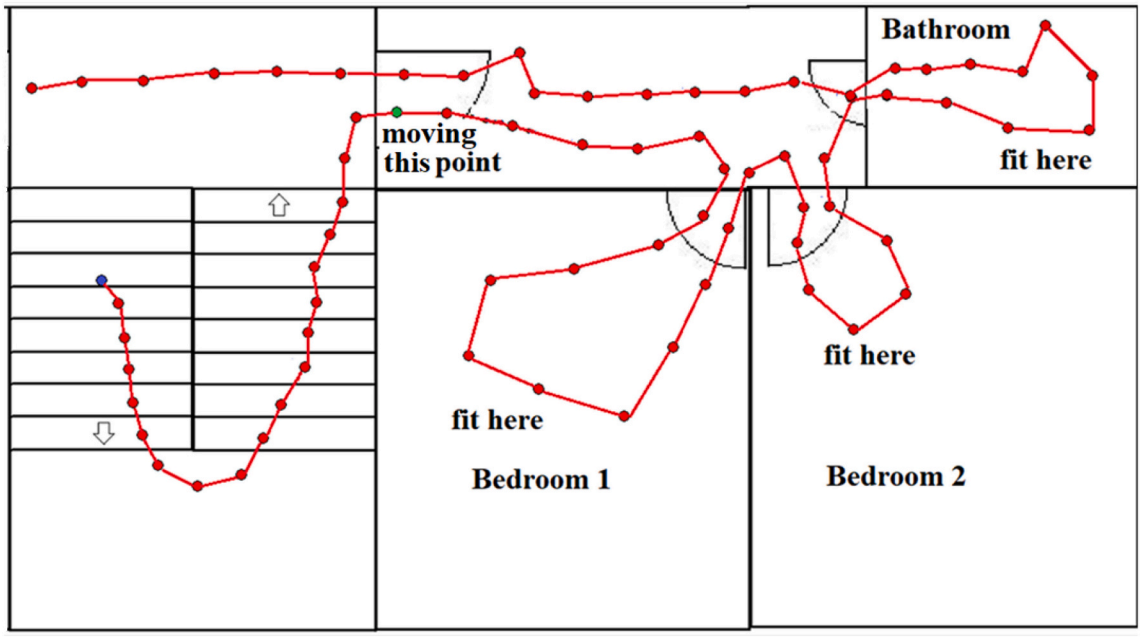


Fig. 5. Illustration of Graphical Transformation: by moving the second point (green point), the rest of the location points (except the point in blue color) move accordingly, and finally the path is fitted into the rooms.

random variables (*rvX*, *rvY,* and *rvZ)*, *rvY* (vertical movement) is uncorrelated with *rvX* and *rvZ* (horizontal movement). In other words, to make the xz plane of the camera coordinate system parallel to the floor plane to avoid the 3D-to-2D scale drift, *rvY* has to be uncorrelated with *rvX* and *rvZ*. This is exactly what PCA is capable of doing.

In summary, the camera coordinate system standardization algorithm enables that the 2-point scheme is practicable to improve productivity, and Graphical Transformation offers an instantaneous way to facilitate the 2-point scheme. The scope in this paper is limited to the issue of camera location for indoor environments.

2. Methodology

In order to simplify the discussion, we define and explain some terminologies and abbreviations, which will be used throughout the rest of the paper.

Standard camera system: a kind of camera coordinate system whose xz plane is parallel to the floor plane (Fig. 2.). It is also called the standard form of a camera system coordinate system.
Camera system: refers to an ordinary camera coordinate system that mostly is not a standard camera system.
SLAM_3DP: the abbreviation of "SLAM 3D path". It is an original path reconstructed by OpenVSLAM, consisting of a series of points expressed in (x, y, z) in a camera system.
SLAM_2DP: the abbreviation of "SLAM 2D path". It is generated by discarding the y coordinates of a SLAM_3DP, and expressed in (x, z), where the z coordinates are equivalent to the y coordinates of a 2D coordinate system.
PCA_STAN_ALGO: the abbreviation of "Principal Component Analysis Camera Standardization Algorithm". It is the standardization algorithm adopting PCA to convert an ordinary camera system into a standard camera system. In order to avoid any ambiguity, we use the word "standardization" rather than "calibration" to describe the process of PCA_STAN_ALGO in this paper.
Standardized SLAM_3DP: it is called standardized SLAM_3DP if a SLAM_3DP is transformed from a camera system into a standard camera system via PCA_STAN_ALGO.
Standardized SLAM_2DP: a SLAM_2DP derived from a standardized SLAM_3DP by directly removing y coordinates.
GRPX_TRANS: the abbreviation of "Graphical Transformation". The GUI algorithm for carrying out the affine transformation. The process of GRPX_TRANS is called "alignment". The words "calibration" and "alignment" are used interchangeably in this paper.
PAMP_ALGO: the abbreviation of "Pivots and Moving Point Algorithm". It is the core algorithm of GRPX_TRANS, which selects one point as a pivot and another one as a moving point to instantly update the locations of nearby points.
2-point scheme: mentioned previously in Chapter 1. It is a scheme moving just two points to complete the affine transformation. PCA_STAN_ALGO is the key for the assurance of the 2-point scheme while GRPX_TRANS is a tool implementing it.
Feature points: a SLAM_2DP or standardized SLAM_2DP contains some branches, each of which represents an inspection route in a room (Fig. 5). By observing a SLAM_2DP or standardized SLAM_2DP, the locations of doors can be recognized. The location points that are near the doors are therefore called feature points.
Floor map system: denotes the coordinate system of a floor map (Fig. 2). Its positive y-axis is in a downward direction, which is the opposite of the z-axis of a camera system.
Aligned SLAM_2DP: a SLAM_2DP or standardized SLAM_2DP is called an aligned SLAM_2DP, if it is transformed onto a floor map.
rvX, rvY and rvZ: refer to three random variables representing the x, y, and z coordinates of a SLAM_3DP separately.
COV() and VAR(): COV() is the covariance of two random variables, and VAR() is the variance of a random variable.

The principle of the methodology presented in the study can be better described by the above terminologies and abbreviations (Fig. 6).

The relationship between a camera system and its standard form is expressed as:

$$\begin{bmatrix} x^{cs} \\ y^{cs} \\ z^{cs} \end{bmatrix} = P \begin{bmatrix} x^{scs} \\ y^{scs} \\ z^{scs} \end{bmatrix} \quad (1)$$

where *cs* is "camera system", *scs* denotes "standard camera system", (x^{cs},

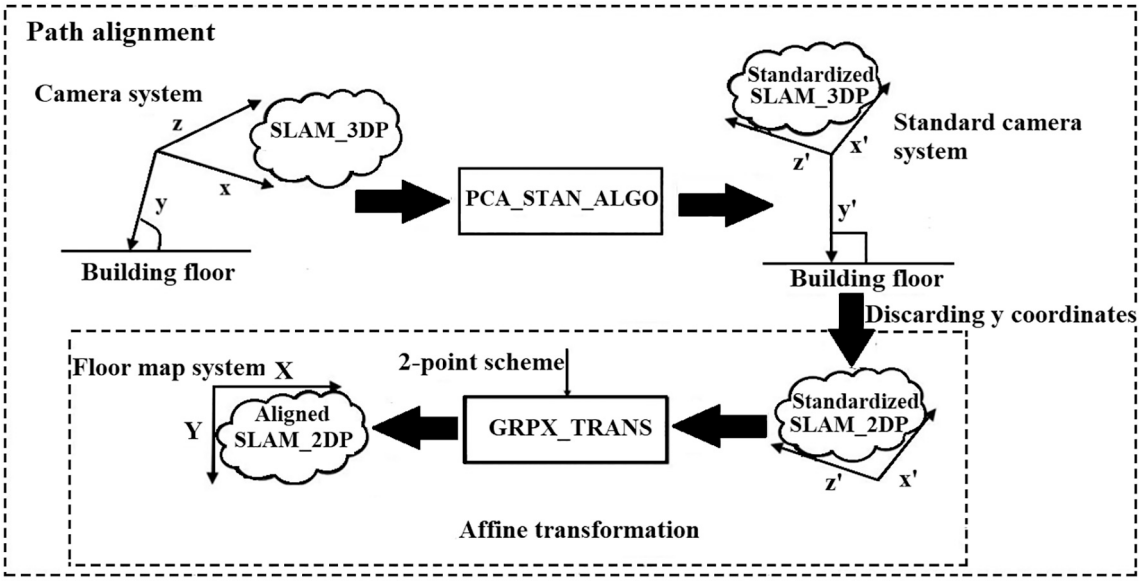


Fig. 6. Schematic diagram of the path alignment of a SLAM path using the proposed methodology. Note: Before PCA_STAN_ALGO, the y-axis of the camera system is not normal to the floor plane, and after PCA_STAN_ALGO, the y-axis is adjusted to be normal to the floor plane.

y^{cs}, z^{cs}) is a path point of a SLAM_3DP in a camera system, (x^{scs}, y^{scs}, z^{scs}) is the same point expressed in a standard camera system, and P is a 3 × 3 rotation matrix. P can be written as [30]:

$$\begin{bmatrix} cos(\theta)+u_x^2(1-cos(\theta)) & u_xu_y(1-cos(\theta))-u_zsin(\theta) & u_xu_z(1-cos(\theta))+u_ysin(\theta) \\ u_xu_y(1-cos(\theta))+u_zsin(\theta) & cos(\theta)+u_y^2(1-cos(\theta)) & u_xu_y(1-cos(\theta))-u_xsin(\theta) \\ u_xu_z(1-cos(\theta))-u_ysin(\theta) & u_zu_y(1-cos(\theta))+u_xsin(\theta) & cos(\theta)+u_z^2(1-cos(\theta)) \end{bmatrix} \quad (2)$$

where $u = (u_x, u_y, u_z)$ is the rotation axis and θ is the rotation around u.

2.1. The description of 3D-to-2D scale drift and PCA_STAN_ALGO

2.1.1. 3D-to-2D scale drift

Supposing a standardized SLAM_3DP SP_3^{scs}, where SP stands for "SLAM path", scs denotes "standard camera system" and the subscript 3 means "3D", consisting of a series of points (x_0, y_0, z_0), (x_1, y_1, z_1), …., (x_n, y_n, z_n), the standardized SLAM_2DP SP_2^{scs} is: (x_0, z_0), (x_1, z_1), …., (x_n, z_n). In reality, a camera system cannot be considered as a standard one because its xz plane cannot be absolutely parallel to the floor plane, unless stabilized. We take a simple case as an example, namely, supposing that we rotate the standard camera system of SP_3^{scs} by an angle θ clockwise around the x-axis to get a new camera system (in practice, it is usually a rotation around an arbitrary axis, see Eq. (2)). Based on Eq. (1), SP_3^{scs} is expressed in this new camera system as:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta) & -sin(\theta) \\ 0 & sin(\theta) & cos(\theta) \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} = \begin{bmatrix} x_i \\ y_icos(\theta)-z_isin(\theta) \\ y_isin(\theta)+z_icos(\theta) \end{bmatrix} \quad (3)$$

In this new camera system, the path is: (x_0, $y_0$cos(θ)-$z_0$sin(θ), $y_0$sin(θ) + $z_0$cos(θ)), (x_1, $y_1$cos(θ)-$z_1$sin(θ), $y_1$sin(θ) + $z_1$cos(θ)), …., (x_n, $y_n$cos(θ)-$z_n$sin(θ), $y_n$sin(θ) + $z_n$cos(θ)) [denoted by SP_3^{cs}, where cs is "camera system"]. SP_3^{scs} and SP_3^{cs} are the same path, but expressed in different camera systems. If discarding y coordinates, SP_2^{cs} is: (x_0, $y_0$sin(θ) + $z_0$cos(θ)), (x_1, $y_1$sin(θ) + $z_1$cos(θ)), …., (x_n, $y_n$sin(θ) + $z_n$cos(θ)). Obviously, SP_2^{cs} and SP_2^{scs} are not the same path anymore (e.g., $z_0 \neq y_0$sin(θ) + $z_0$cos(θ)), and they describe two different paths on the floor. This phenomenon is the 3D-to-2D scale drift. The variation of y coordinates is one of the key deciding factors on the issue of the 3D-to-2D scale drift. The 3D-to-2D scale drift is much smaller on the floor than on the stairs. The only way to resolve the 3D-to-2D scale drift is to restore a camera system to its standard form, namely, finding out the rotation matrix P in Eq. (1). The traditional way is to find the rotation axis and rotation angle in Eq. (2), which is complicated as discussed in Chapter 1. However, PCA_STAN_ALGO takes a different approach to determine P without having to determine the rotation axis and rotation angle.

2.1.2. PCA_STAN_ALGO

As can be seen in Eq. (3), rvY is correlated with rvZ (i.e., COV(rvY, rvZ) ≠ 0) in SP_3^{cs}. If rotating around the z-axis, rvY will be correlated with rvX (i.e., COV(rvY,rvX) ≠ 0). So, in a camera system, rvY is most likely correlated with both rvX and rvZ.

Let [L_1 L_2….L_n] be a 3xn SLAM_3DP sample data in the form of column vector ($L_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}$). The sample mean, M, is [31]:

$$M = \frac{1}{n}(L_1 + L_2 + \ldots + L_n) \quad (4)$$

For L_1, L_2, …, L_n, we have:

$$\widehat{L}_k = L_k - M, k = 1, 2, .., \ n \quad (5)$$

A new 3xn matrix is:

$$B = \left[\widehat{L}_1 \ \widehat{L}_2 \ldots \widehat{L}_n\right], \quad (6)$$

where B has a zero sample mean and B is said to be in the mean deviation

form of $[L_1\ L_2 \ldots L_n]$. The sample covariance matrix S of $[L_1\ L_2 \ldots L_n]$ is defined as [31]:

$$S = \frac{1}{n-1} BB^T, \tag{7}$$

where B^T is the transpose of B. For the convenience of discussion, the sample covariance S can also be expressed in a matrix form using three random variables *rvX*, *rvY,* and *rvZ* as:

$$S = \begin{bmatrix} \mathrm{VAR}(rvX) & \mathrm{COV}(rvX, rvY) & \mathrm{COV}(rvX, rvZ) \\ \mathrm{COV}(rvY, rvX) & \mathrm{VAR}(rvY) & \mathrm{COV}(rvY, rvZ) \\ \mathrm{COV}(rvZ, rvX) & \mathrm{COV}(rvZ, rvY) & \mathrm{VAR}(rvZ) \end{bmatrix} \tag{8}$$

In a standard camera system, *rvY* is uncorrelated with *rvX* and *rvZ* (see Chapter 1), resulting in COV(*rvY*, *rvX*) = COV(*rvX*,*rvY*) = COV(*rvY*, *rvZ*) = COV(*rvZ*,*rvY*) =0. Rotating a standard camera system around its y-axis will generate a different camera system because the orientations of the x- and z-axis are changed. However, the resultant (new) camera system is still in a standard form because its y-axis is normal to the floor plane. This indicates that there is an infinite number of standard camera systems for the floor plane. Therefore, we can find a standard camera system so that *rvX* is also uncorrelated with *rvZ*, i.e., COV(*rvX*, *rvZ*) = COV(*rvZ*, *rvX*) =0. So, the simplest way to find a standard camera system is to make the sample covariance matrix S diagonal. PCA is the best tool for solving such a problem.

The goal of PCA is to find an orthogonal 3 × 3 matrix $P = [u_1\ u_2\ u_3]$ that determines a new variable, $\widehat{L}_i = PG_i$, or

$$\begin{bmatrix} \widehat{L}_i^x \\ \widehat{L}_i^y \\ \widehat{L}_i^z \end{bmatrix} = [u_1\ \ u_2\ \ u_3] \begin{bmatrix} G_i^x \\ G_i^y \\ G_i^z \end{bmatrix} \tag{9}$$

so that the new variables G_i^x, G_i^y, and G_i^z are uncorrelated (the superscripts denote the x, y and z coordinates of a point). Keep in mind that (G_i^x, G_i^y, G_i^z) is a point in a standard camera system and P is the rotation matrix in Eq. (1). Let C=[$G_1\ G_2 \ldots G_n$]), we have:

$$C = \left[P^{-1}\widehat{L}_1\ P^{-1}\widehat{L}_2 \ldots P^{-1}\widehat{L}_n \right] = P^{-1}\left[\widehat{L}_1\ \widehat{L}_2 \ldots \widehat{L}_n \right] = P^T B \tag{10}$$

Note: $P^{-1} = P^T$ because P is orthogonal. Substituting C for B in Eq. (7), the corresponding sample covariance matrix is obtained as:

$$S' = \frac{1}{n-1} CC^T = \frac{1}{n-1} P^T B \left(P^T B\right)^T = \frac{1}{n-1} P^T BB^T P = P^T SP \tag{11}$$

$$S = PS'P^T \tag{12}$$

$S'$is a diagonal matrix. Eqs. (9)–(12) imply that a rotation matrix P in Eq. (1) can be found as long as the covariance matrix S can be diagonalized into the form of PDP^T, where P is an orthogonal matrix and D is a diagonal matrix. One important theory is that any symmetric matrix is orthogonally diagonalizable [31]. For the sample covariance matrix, S can be orthogonally diagonalized as:

$$\mathrm{S} = PDP^T = [\,v_x \quad v_y \quad v_z\,] \begin{bmatrix} \lambda_x & 0 & 0 \\ 0 & \lambda_y & 0 \\ 0 & 0 & \lambda_z \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \tag{13}$$

where λ_x, λ_y, and λ_z are the eigenvalues of S, and v_x, v_y, and v_z are the corresponding eigenvectors. λ_x, λ_y, and λ_z also denote the (sample) variances of three principal components (i.e., G_i^x, G_i^y, and G_i^z, see Eq. (9)). Many software tools (e.g., Matlab) are able to orthogonally diagonalize symmetric matrixes like S (Eq. (8)), but they arrange the diagonal entries of a diagonal matrix (i.e., S'in Eq. (12)) in descending order. In practice, the best way is to test all possible versions of the rotation matrix P in Eq. (13) (there are totally six different arrangements for P) to make sure that the sample variance S' is indeed diagonal.

In a nutshell, PCA_STAN_ALGO first calculates the sample covariance of a SLAM_3DP using Eq. (7), and then orthogonally diagonalizes the sample covariance using Eq. (13) to get the rotation matrix P between the current camera system and a standard camera system (Eq. (1)). With the rotation matrix P, the SLAM_3DP can be transformed from the current camera system into a standard camera system using Eq. (1). In a standard camera system, the 3D-to-2D scale drift will disappear from the xz plane.

2.2. Application of PCA_STAN_ALGO for indoor construction inspection routes in buildings

Fig. 7 shows the layout of a typical Finnish apartment building and inspection route.

PCA_STAN_ALGO follows the following steps to standardize SLAM_3DP sample data:

1. The preparation of data. A video inspection normally starts on the lower floor or somewhere on the lower part of the stairs (S-1, Fig. 7), and the whole video covers all flats on the upper floor (F, Fig. 7) and the stairs (S-1 and S-2 in Fig. 7). Only the data from the same floor (F, Fig. 7) are selected for PCA_STAN_ALGO because *rvY* is correlated with *rvZ* for the sample data related to the stairs (Sections S-1 and S-2 in Fig. 7). For instance, in the camera system in Fig. 7, *rvY* decreases for the lower part of stairs (S-1, Fig. 7) while *rvZ* increases. So, they are negatively correlated.
2. The selected part of SLAM_3DP data from Step 1 are used to estimate the rotation matrix P in Eq. (1) using Eqs. (7) and (13).
3. The P rotation matrix from Step 2 is used for all SLAM_3DP data including the stairs (S-1 + S-2 + F, Fig. 7) using Eq. (1), namely, $\begin{bmatrix} x^{scs} \\ y^{scs} \\ z^{scs} \end{bmatrix} = P^T \begin{bmatrix} x^{cs} \\ y^{cs} \\ z^{cs} \end{bmatrix}$.

Now the SLAM_3DP is in a standard camera system, and the y coordinates can be neglected. The obtained (standardized) SLAM_2DP is ready for GRPX_TRANS (Fig. 6).

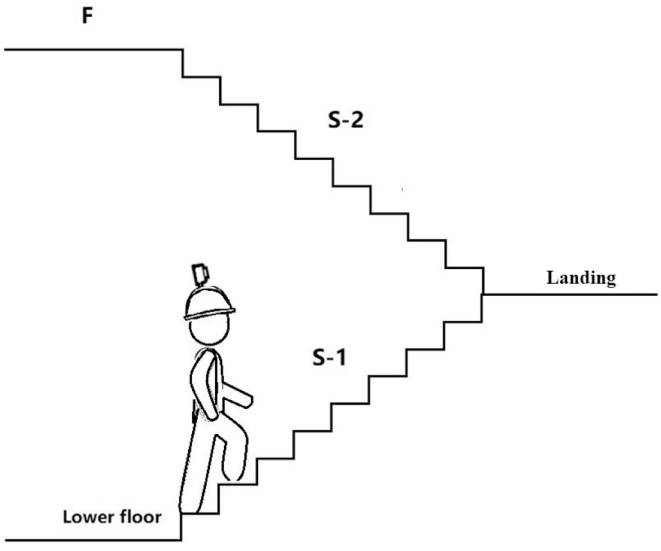


Fig. 7. Layout of a typical inspection route in an apartment building includes staircases (S-1: the lower part of the stairs, i.e., from the lower floor to the landing; S-2: the upper part of the stairs, i.e., from the landing to the upper floor; F: the upper floor including all flats and public sections).

2.3. The description of GRPX_TRANS

GRPX_TRANS was designed not only for the 2-point scheme, but also for a general purpose. Actually, the users can move much more than just two points to solve complicated cases, such as an inspection video that is cut into several parts for the vSLAM system, which in effect leads multiple camera system sections to be created from the original inspection path. This paper focuses only on the discussion of algorithms related to linear transformations. The design of the GUI is not included here.

In GRPX_TRANS, a SLAM_2DP (or standardized SLAM_2DP) will undergo three consecutive linear transformation processes to finalize the affine transformation: (1) Preliminary estimation of scaling and translation; (2) Reflection operations; (3) Update of rotation, scaling, and translation.

2.3.1. Preliminary estimation of scaling and translation

This is done automatically whenever a SLAM_2DP is loaded onto a floor map. The scaling and translation are approximated based on the sizes of the floor map and SLAM_2DP. The goal is to display all camera points belonging to the inspection path on the floor map properly for the subsequent operations.

2.3.2. Reflection operations

GRPX_TRANS supplies two reflection operations to cope with situations where the x- or/and z-axis of a camera system is/are opposite to the x- or/and y-axis of the corresponding floor map system with the following equations:

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_0 & x_1 & \dots & x_n \\ z_0 & z_1 & \dots & z_n \end{bmatrix} = \begin{bmatrix} x_0 & x_1 & \dots & x_n \\ -z_0 & -z_1 & \dots & -z_n \end{bmatrix} \text{(over the x} - \text{axis)} \tag{14}$$

$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 & x_1 & \dots & x_n \\ z_0 & z_1 & \dots & z_n \end{bmatrix} = \begin{bmatrix} -x_0 & -x_1 & \dots & -x_n \\ z_0 & z_1 & \dots & z_n \end{bmatrix} \text{(over the y} - \text{axis)} \tag{15}$$

where (x_0, z_0), (x_1, z_1), …,(x_n, z_n) are the location points of a SLAM_2DP. The users can carry out the reflection operations by using the menu without moving any point.

2.3.3. Update of rotation, scaling and translation

It is implemented by PAMP_ALGO. In a SLAM_2DP on a floor map, PAMP_ALGO classifies the points into three types: (1) Pivot points: the points that were selected by the user and moved previously (a point can also be set as a pivot point without changing its location); (2) Torque point: the point that is being moved by the user. When the user stops moving this point, it will become a pivot point; (3) Passive points: the points that have never been selected by the user. A pivot point can also be forcibly converted to a passive point by the user. The instant location of a passive point is decided by the locations of the current torque point and nearby pivot points. If there are more than one pivot point between the torque point and a passive point or this passive point is located between two pivot points, there will be no displacement for this passive point. The main contribution of PAMP_ALGO is, when the user moves a point (torque point), the related points (passive points) will move accordingly and simultaneously. In this way, a SLAM_2DP can be easily fit in a floor map based on its shapes (Figs. 4 and 5). PAMP_ALGO is better illustrated using a simple SLAM_2DP (including only eight points) as an example. Fig. 8 shows four common scenarios.

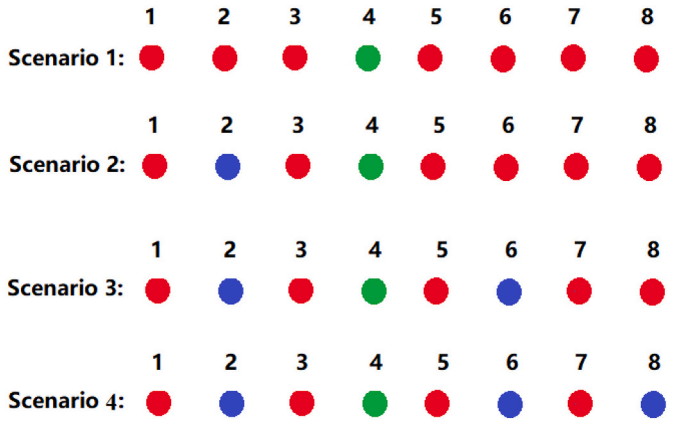


Fig. 8. Four common scenarios for GRPX_TRANS. Red, blue, and green circles represent passive, pivot, and torque points separately. Scenario 2 illustrates the 2-point scheme.

1. Scenario 1: one torque point (Point 4) and there is no pivot point around its vicinity. In this case, only a translation is carried out for all points. So $\begin{bmatrix} x_t^n \\ y_t^n \end{bmatrix} = \begin{bmatrix} x_{t-1}^n \\ y_{t-1}^n \end{bmatrix} + \begin{bmatrix} x_t^4 - x_{t-1}^4 \\ y_t^4 - y_{t-1}^4 \end{bmatrix}$, where the superscript is the point index $n = (1, \dots,8)$, t is the current time and *t-1* is the previous time.
2. Scenario 2: one torque point (Point 4) and there is a pivot point (Point 2) nearby. Therefore, the location of Point 2 is fixed, i.e., $\begin{bmatrix} x_t^2 \\ y_t^2 \end{bmatrix} = \begin{bmatrix} x_{t-1}^2 \\ y_{t-1}^2 \end{bmatrix}$. Point 4 will be moved continually. The rest of the points (Points 1, 3, 5,6,7, and 8) will have a sequence of rotations, scalings, and translations based on the relative locations between Points 4 and 2 (note that, even though Point 2 has no movement, the relative location between Points 2 and 4 is changing, which in turn keeps updating the transformation matrix). The transformation matrix is updated as follows:

$$\begin{bmatrix} a \\ b \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} x_{t-1}^4 & -y_{t-1}^4 & 1 & 0 \\ y_{t-1}^4 & x_{t-1}^4 & 0 & 1 \\ x_{t-1}^2 & -y_{t-1}^2 & 1 & 0 \\ y_{t-1}^2 & x_{t-1}^2 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} x_t^4 \\ y_t^4 \\ x_t^2 \\ y_t^2 \end{bmatrix} \tag{16}$$

where a and b are two parameters related to the rotation, (t_x,t_y) is the translation. The locations of Points 1, 3,5,6,7, and 8 are computed as:

$$\begin{bmatrix} x_t^n \\ y_t^n \\ 1 \end{bmatrix} = \begin{bmatrix} a & -b & t_x \\ b & a & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{t-1}^n \\ y_{t-1}^n \\ 1 \end{bmatrix} \tag{17}$$

where $n = 1,3,5,6,7$, and 8.

3. Scenario 3: one torque point (Point 4) and two pivot points (Points 2 and 6) to its left and right. In this case, Points 1 and 3 will have rotations, scalings, and translations based on the relative locations between Points 2 and 4 while Points 5, 7, and 8 based on Points 4 and 6 (Eqs. (16) and (17)).
4. Scenario 4: one torque point (Point 4) and three pivot points (Points 2, 6, and 8). There is no movement for Point 7. Points 1, 3, and 5 act the same way as in Scenario 3.

3. Case study

In this work, all SLAM_3DP, standardized SLAM_3DP, SLAM_2DP, and standardized SLAM_2DP data are considered as experimental data because they model real indoor construction inspection routes. If experimental data are modified artificially so that the resultant path describes a totally different path from the original one, the modified (experimental) data are called synthetic data. The case study contains both synthetic and experimental data, thus illustrating all five cases.

Cases 1–4 demonstrate the ability of PCA_STAN_ALGO to eliminate the 3D-to-2D scale drift while case 5 depicts the procedures for applying GRPX_TRANS to carry out the 2-point scheme. Quantitative evaluation was only carried out for Cases 1–4 in order to compare the accuracy between SLAM_2DP and standardized SLAM_2DP data. A simple GUI software tool, which implements the entire process of the path alignment (see Fig. 6) including PCA_STAN_ALGO and GRPX_TRANS, was developed in Python and tkinter by the authors. It is mainly used for the purpose of study and training. Most of the examples in the case study were conducted using this GUI software tool. The rest that could not be implemented in the current version of GUI software tool were performed in Matlab, for example, the step of rotating a camera system to a specified orientation. In the future, more features will be included in the GUI software tool.

Case 1. Synthetic data. A modified SLAM_3DP describing a part of a real inspection route inside an apartment building (i.e., a part of that real inspection route was removed) was assumed to be the true path. The default camera coordinate system is assumed to be in a standard format and in the same orientation as the floor map, namely, the z-axis points from top to bottom, the x-axis points from left to right, and the y-axis is into the page. All other (3D) paths were obtained by transforming the true path from the standard camera system into different camera systems. Fig. 9 shows the initial path (true path) on the floor map.

As can be seen in Fig. 9, the starting point is on the lower floor. There are two studies for Case 1:

1) Examining how it will affect the true path on the floor if having the current standard camera system rotated clockwise around the x-axis by 15 degrees followed by a clockwise rotation of 10 degrees around the z-axis (i.e., demonstrating how the 3D-to-2D scale drift comes from).
2) Presenting the efficiency of PCA_STAN_ALGO: applying PCA_STAN_ALGO to restore the obtained camera system to a standard form.

Case 1 presents a theoretical study on the issues of the 3D-to-2D scale drift and PCA_STAN_ALGO. The aim is to show that even a true path can result in the 3D-to-2D scale drift, implying that the 3D-to-2D scale drift is the outcome of improper (orthographic) projection from 3D to 2D.

Cases 2–4. all are experimental data from three inspection videos shot by Gopro Fusion 360 cameras (dual 190-degree fisheye lenses) in 2019 for a building renovation project. Figs. 10 and 11 show the SLAM_2DP data.

The true starting points were approximated by observing the videos. These cases exhibit very common 3D-to-2D scale drift. The goal of Cases 2–4 is to demonstrate the capability of PCA_STAN_ALGO in removing the 3D-to-2D scale drift.

Very high accuracy of the paths reconstructed by SLAM in indoor construction inspection is generally not desirable. For instance, in this study, the size of a room and the distance between two adjacent rooms are not particularly large. Therefore, the customers are often interested in which room an image (frame) was taken and where the inspection route starts. With modern vSLAM algorithms and the adequate path alignment, this goal can be achieved, such as the methodology discussed in this paper. In this circumstance, the digital company is normally not willing to set up the experiment to evaluate the performance of the vSLAM program because it cannot bring much benefit to indoor construction inspection in terms of time and cost. Thus, in this study, in order to carry out a quantitative evaluation, the only option is to send the original and standardized SLAM_2DP data to the inspectors who shot the videos to mark the correct locations manually. It is difficult and time-consuming to precisely estimate the camera location for a video frame although all videos are 360-degree ones. So, it is impractical for an inspector to determine every single location point for a reference path. Oftentimes it is required to make sacrifices for the scientific truth. Thus, the inspectors tried to correct the locations of the standardized SLAM_2DPs (according to the feedback from the inspectors, the standardized SLAM_2DPs are much closer to the real inspection routes) for only those locations that appear to have clear deviations from the inspection paths, e.g., a location point and the wall overlap, etc. But, the area of the stairs in a building (S-1 + S-2, Fig. 7) is an exception, where all location points were checked and corrected because the 3D-to-2D scale drift is much more serious in there. In this way, the locations determined by the inspectors are used as the substitutes of the true paths to provide a quantitative evaluation from the inspectors' point of view.

Case 5. GRPX_TRANS was used for the standardized SLAM_2DP from Case 2 to illustrate the 2-point scheme.

3.1. The path alignment for comparisons

In Cases 1–4, both a SLAM_2DP and standardized SLAM_2DP need to be presented on the same floor map for visual comparison. Careful procedures must be taken for the path alignment to guarantee that (1) the shapes of two paths remain unchanged, and (2) the distortions caused by the 3D-to-2D scale drift should be preserved as much as possible. Therefore, moving a point from the stairs (S-1 + S-2, Fig. 7) to its target location on a floor map for the path alignment should be avoided due to strong 3D-to-2D scale drift in that area. There are two types of comparisons: the comparison between the true path and the distorted path (Case 1), and the comparison between a SLAM_2DP and standardized one (Cases 2–4). For the first type of comparison, the path alignment is simple because there exists only translation. The translation was calculated based on a random location point selected from the true path as well as its corresponding point from the distorted one. The path

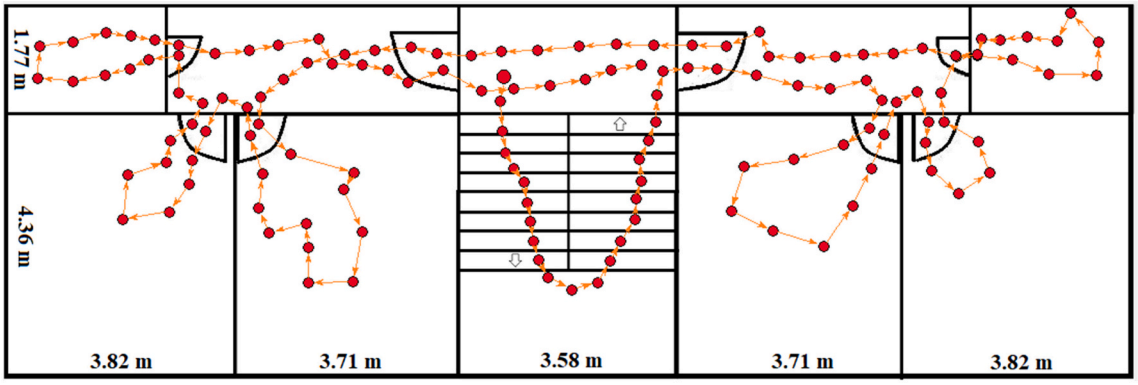


Fig. 9. Case 1: the true path on the floor starting from the lower floor (the starting point is denoted by a larger circle). The floor map is synthetic based on real floor plans, but its stairs were made larger than those in normal apartment buildings in order to emphasize the 3D-to-2D scale drift (Chapter 4).

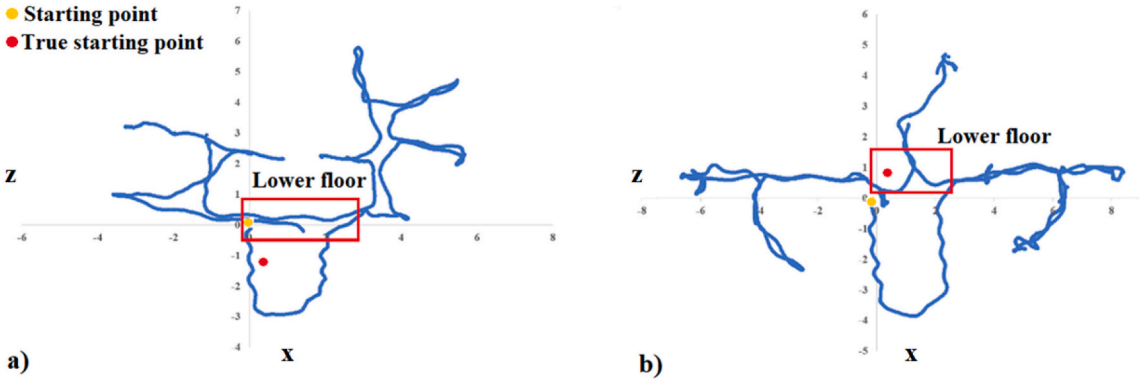


Fig. 10. The SLAM_2DP data of two real inspection routes in two buildings under renovation. a) Case 2: the true starting point is on the lower part of the stairs. b) Case 3: the true starting point is on the lower floor.

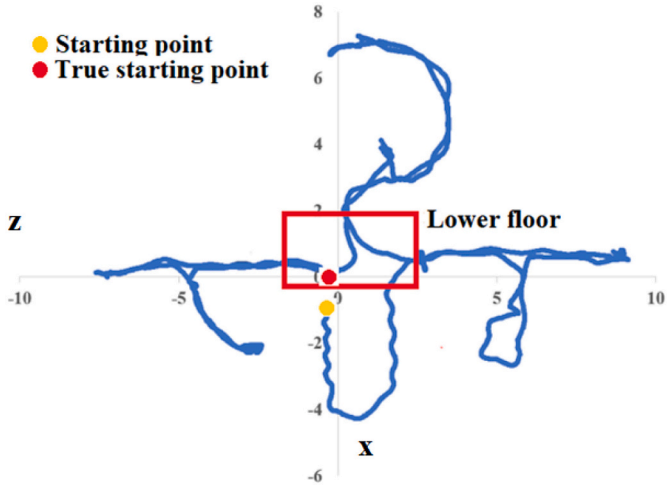


Fig. 11. The SLAM_2DP data of a real inspection route in a building under renovation for Case 4 (the true starting point is on the lower floor).

alignment is a bit complicated for the second comparison because reflection, rotation, scaling, and translation are all involved. The reflections and rotations between the camera systems and the floor map system were calibrated manually. The most difficult part is to select appropriate scalings and translations. Initial scalings and translations were automatically calculated by the developed GUI software tool to display both the SLAM_2DP and standardized SLAM_2DP on the floor map. The scalings and translations were then updated by selecting two common tie feature points for the two paths:

1. A tie point on the floor map is the common destination point of a pair of points from the two paths. A feature point quite near the entrance door to the first flat was selected as the first common tie point for Cases 2–4. The location of this tie point on the floor map was then pinpointed by carefully studying the video, which updated the translations (t_x and t_y in Eq. (16)) for both paths as well. After that, the distances between the two paths were computed for all the paired points (e.g., the distance between the two paths is zero for the first selected common tie point).
2. Then, the second common tie point was selected, which is the least distance between the two paths. The location of the second common tie point on the floor map was re-evaluated and corrected from the video. The new scalings for the two paths were calculated as the distance between the first and second common tie points on the floor map divided by their original distances (i.e., before the two paths were mapped onto the floor map).

3.2. Evaluation formula

RMSE (root mean square error) was used as the evaluation formula for Cases 1–4:

$$RMSE = \sqrt{\frac{\sum_{i=0}^{N}\left(dis0^2 + dis1^2 + \ldots + disN^2\right)}{N}} \tag{18}$$

where *dis* is the distance (in m) between the estimated location from the standardized SLAM_2DP (or SLAM_2DP) and the one by the inspector. These distances were first calculated on floor maps in pixels and then converted to meters, because the true (clear) widths of the flat entrance doors (910 mm) and their widths on the floor maps were known.

4. Results and discussion

The results are presented and discussed in this section.

4.1. Case 1

Fig. 12 shows the comparison between the true path and the distorted path caused by the rotation around both the x- and z-axis.

Some observations were made:

- When rotating around the x-axis of a camera system, the path shows displacements in the direction of the y-axis of a floor map system. Similarly, the displacements will be in the direction of the x-axis if rotating the camera around the z-axis. In practice, there will be a composite of rotations around the x-, y-, and z-axis.
- The displacements have a much bigger variance in the area of stairs [the lower floor +S-1 + S-2, Fig. 7] than on the target floor (F, Fig. 7). *rvY* (SLAM_3DP) has great impact on the displacements. If taking the target floor as the centre, the largest 3D-to-2D scale drift appears at the beginning of the path.
- The variance of *rvY* (SLAM_3DP) decides the degree of the 3D-to-2D scale drift. The greater it is, the larger the 3D-to-2D scale drift would be.
- The paths experiencing the 3D-to-2D scale drift could give the users false information, for instance in Fig. 12, the location of the starting point could be mistaken for the third step of the stairs (actually on the lower floor).

Avoiding the occurrences of misinformed paths in Fig. 12 and alike is of great interest for both digital service and construction companies. PCA_STAN_ALGO was employed to remove the 3D-to-2D scale drift in cases like that shown in Fig. 12 (Fig. 13). The comparison from Figs. 12–13 is also presented in the coordinate system (Fig. 14).

The standardized path produced by PCA_STAN_ALGO exactly

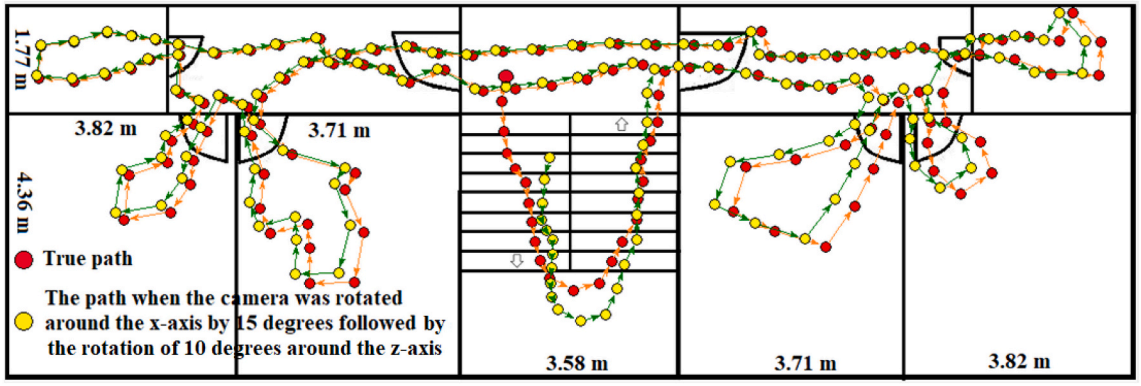


Fig. 12. Comparison of the true path and the simulated path by rotating the camera around both the x- and z-axis (i.e., 3D-to-2D scale drift) for Case 1. The true starting point is denoted by a larger red circle.

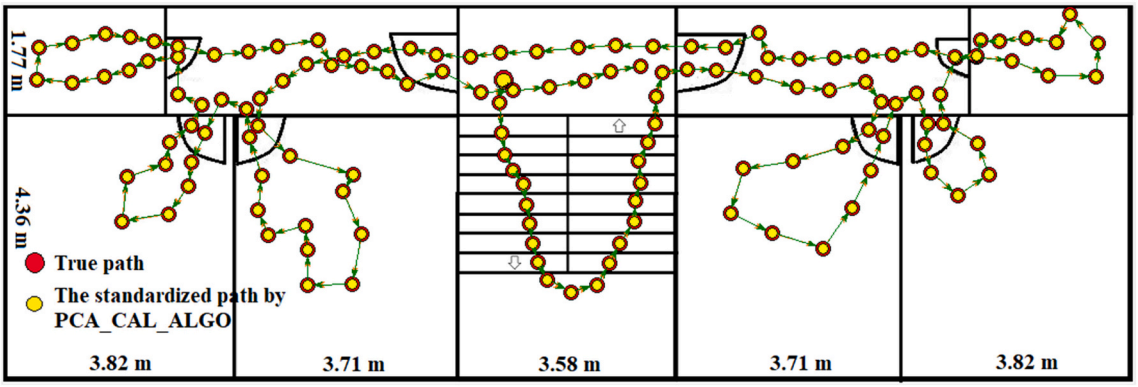


Fig. 13. Comparison of the true path and the standardized path by PCA_STAN_ALGO for Case 1 (the starting point is denoted by a larger circle). The standardized path coincides with the true path.

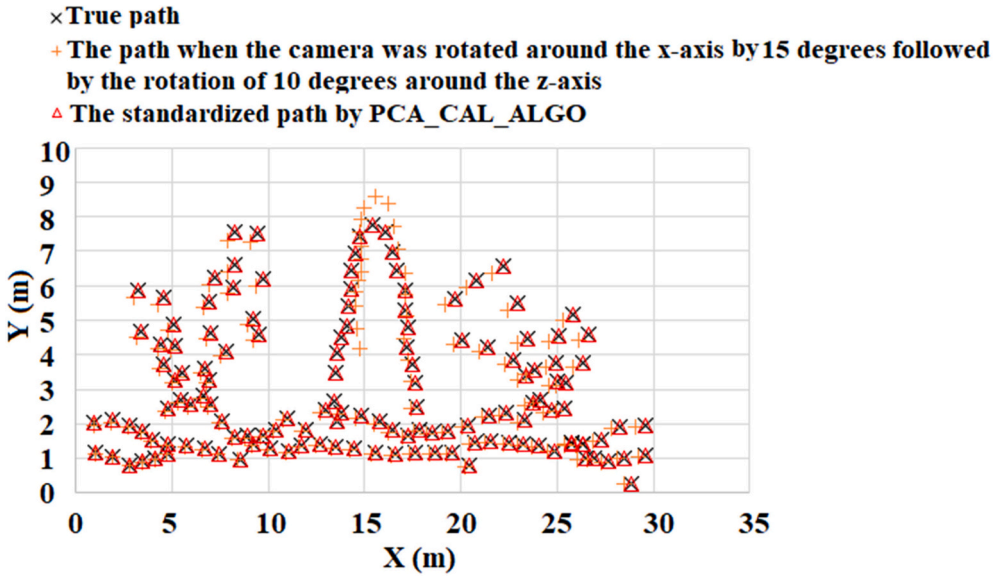


Fig. 14. Comparison of the true path, the simulated path by rotating the camera around both the x- and z-axis (i.e., 3D-to-2D scale drift) and the standardized path by PCA_STAN_ALGO in the coordinate system for Case 1.

coincides with the true reference. RMSE values are zero and 0.36 m for the standardized path (Figs. 13–14) and the distorted path by rotating around both the x- and z-axis (Figs. 12 and 14) respectively. All these show that the PCA based method is a very convenient, easy to use, and fast tool in handing the 3D-to-2D scale drift. It is worth mentioning that in PCA_STAN_ALGO we must use the data from the same floor (F, Fig. 7) to obtain the rotation matrix (P in Eq. (1)) first, and then apply the obtained rotation matrix to the entire data set to remove the 3D-to-2D

Table 1
The comparisons with the estimated path locations by inspectors.

	SLAM_2DP (RMSE)	Standardized SLAM_2DP by PCA_STAN_ALGO (RMSE)
Case 2	0.63 (m)	0.091 (m)
Case 3	0.37 (m)	0.090 (m)
Case 4	0.50 (m)	0.094 (m)

scale drift. PCA_STAN_ALGO is only applicable for indoor cases.

4.2. Cases 2–4

The results are displayed in Table 1 and Figs. 15–18.

As we can see in Table 1 and Figs. 15–18, PCA_STAN_ALGO did restore the paths for all of the cases, particularly for the beginning of each path where the 3D-to-2D scale drift can clearly be seen for the SLAM_2DP. The standardized paths follow the videos quite well. The solutions are quite close in all three cases for the standardized paths, but vary for the SLAM_2DPs (Table 1). This implies that different camera systems experienced different rotations around the x- or/and z-axis during filming. Most likely the rotation around the x-axis of the camera system dominates in Case 3 (see Fig. 16, the displacements in x direction are small) while Cases 2 and 4 experience rotations around both the x- and z-axis. As a result, the performance of the SLAM_2DP is better in Case 3 than in Cases 2 and 4. There are some minor overlaps between the standardized path and the nearby walls/kitchen counter in Case 3 (the ovals in Fig. 16), some of which were corrected by the inspector. This might be because some parts of the true path are too close to the walls/kitchen counter, or there are imprecise locations for the two selected common tie points, or errors from the vSLAM system. Figs. 15–17 also hint that moving two points can correctly map a standardized SLAM_2DP onto the floor map. For a SLAM_2DP, more point manipulation is required to fix the apparent 3D-to-2D scale drift, for example, in Case 2, some local pruning operations (e.g., in the beginning) are needed.

4.3. Case 5

Fig. 19 illustrates the procedures of moving two points (the 2-point scheme) to complete the affine transformation for the standardized SLAM_2DP of Case 2 using GRPX_TRANS (Fig. 15).

PCA_STAN_ALGO adjusts all three x-, y-, and z-axis to make *rvX*, *rvY*, and *rvZ* (SLAM_3DP) uncorrelated mutually. So, there is always a rotation around the y-axis, which can be clearly seen in Fig. 19. In Case 5, the starting point was selected as the first moving point. But, in practice, it is difficult to find the precise location of the starting point. A better way is to pick up one of feature points, instead of using the starting point. The second moving point is used for estimating the rotation and scaling (Fig. 19).

The coordinate system transformation from a camera system into a floor map system can be regarded as a 2D affine transformation including reflection, rotation, scaling, and translation. An affine transformation matrix contains six unknown parameters. However, GRPX_TRANS separates reflection from rotation, scaling, and translation, because reflection does not require any point moved. The separation reduces the number of unknown parameters to four for the transformation matrix, which makes the 2-point scheme functional. The 2-point scheme not only requires the minimum number of points moved, but also keeps the shape of the SLAM_2DP path unchanged. Moving more than two points will create a risk of twisting the SLAM_2DP.

4.4. Time used for the path alignment

Reducing the time spent for the path alignment is another goal of this work. Methodology and human factors are two key issues in this case. It is difficult to quantify the human influence because there are many factors involved, such as the engineer's skill, experience, and knowledge. Because moving two points has no risk of distorting a path, the 2-point scheme has a low demand for skill, experience, and knowledge. This indicates that the improvement of methodology will decrease the human influence on the time spent for the path alignment. Thus, in this

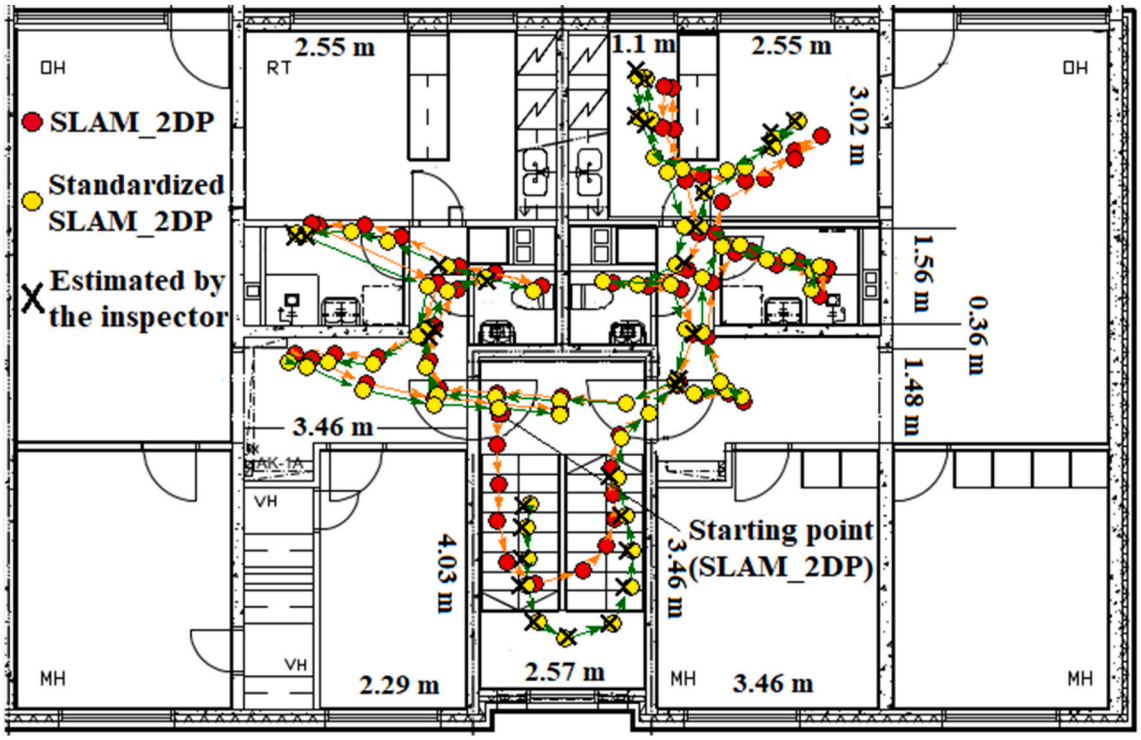


Fig. 15. Comparison of the SLAM_2DP, the standardized SLAM_2DP by PCA_STAN_ALGO and the estimated path locations by the inspector for Case 2. The time step fluctuates between 1 s and 3.25 s.

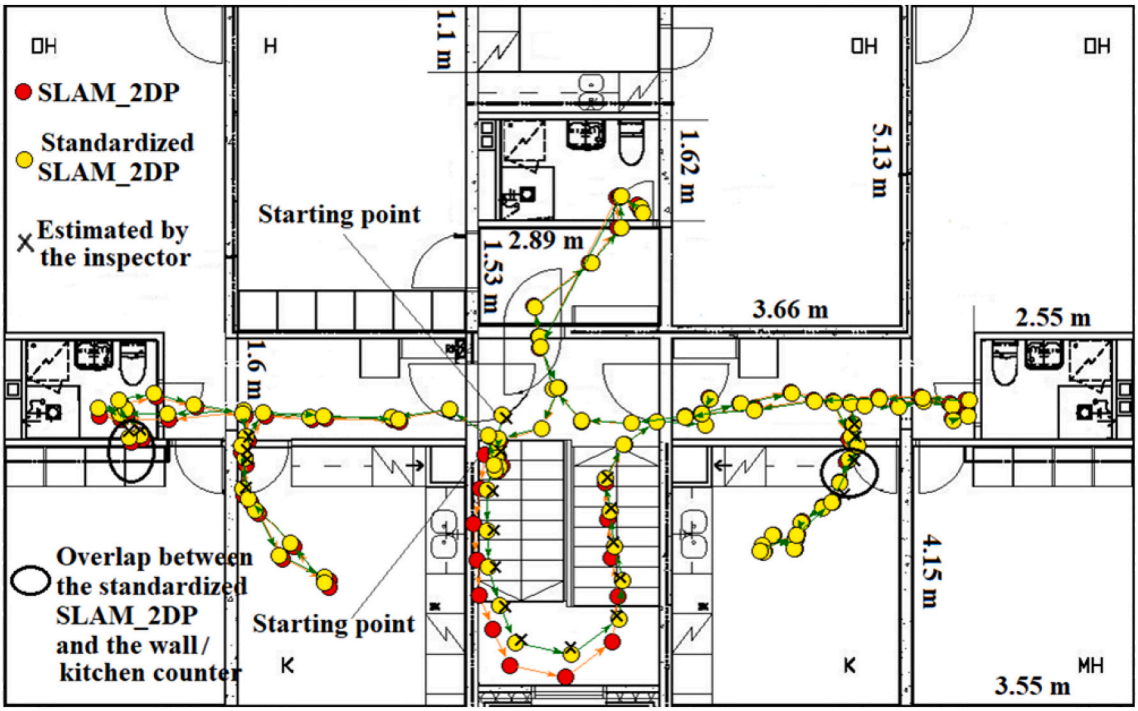


Fig. 16. Comparison of the SLAM_2DP, the standardized SLAM_2DP by PCA_STAN_ALGO and the estimated path locations by the inspector for Case 3. The time step fluctuates between 1 s and 2.5 s.

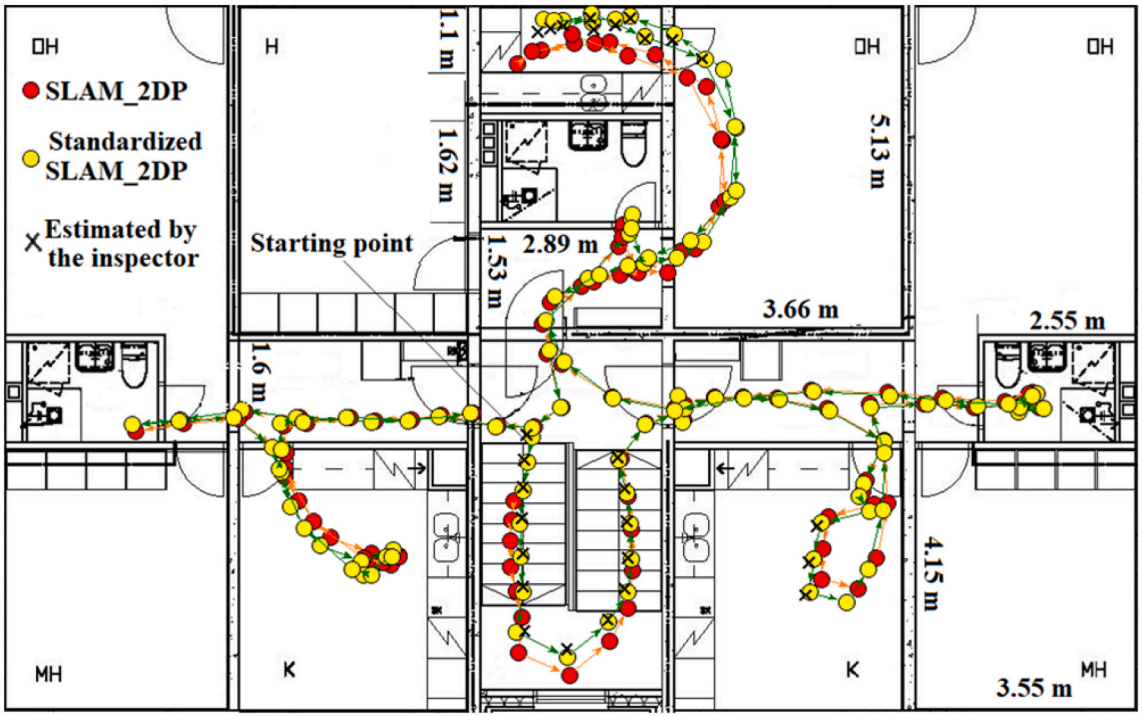


Fig. 17. Comparison of the SLAM_2DP, the standardized SLAM_2DP by PCA_STAN_ALGO and the estimated path locations by the inspector for Case 4. The time step fluctuates between 1 s and 8.7 s.

study, we conducted an experiment to quantify only the impact of methodology on the path alignment in terms of time spent by comparing two methods, using only GRPX_TRANS and using both PCA_STAN_ALGO and GRPX_TRANS, for Cases 2–4. Previous results have shown that using both PCA_STAN_ALGO and GRPX_TRANS gave very accurate results for Cases 2–4 (i.e., standardized SLAM_2DPs in Figs. 15–17 and the third column of Table 1). Therefore, we took the standardized SLAM_2DPs from Figs. 15–17 as our target paths while the original SLAM paths (i.e., SLAM_3DPs) were regarded as source paths. The aim was to carry out the path alignment for the source paths so that the aligned source paths were

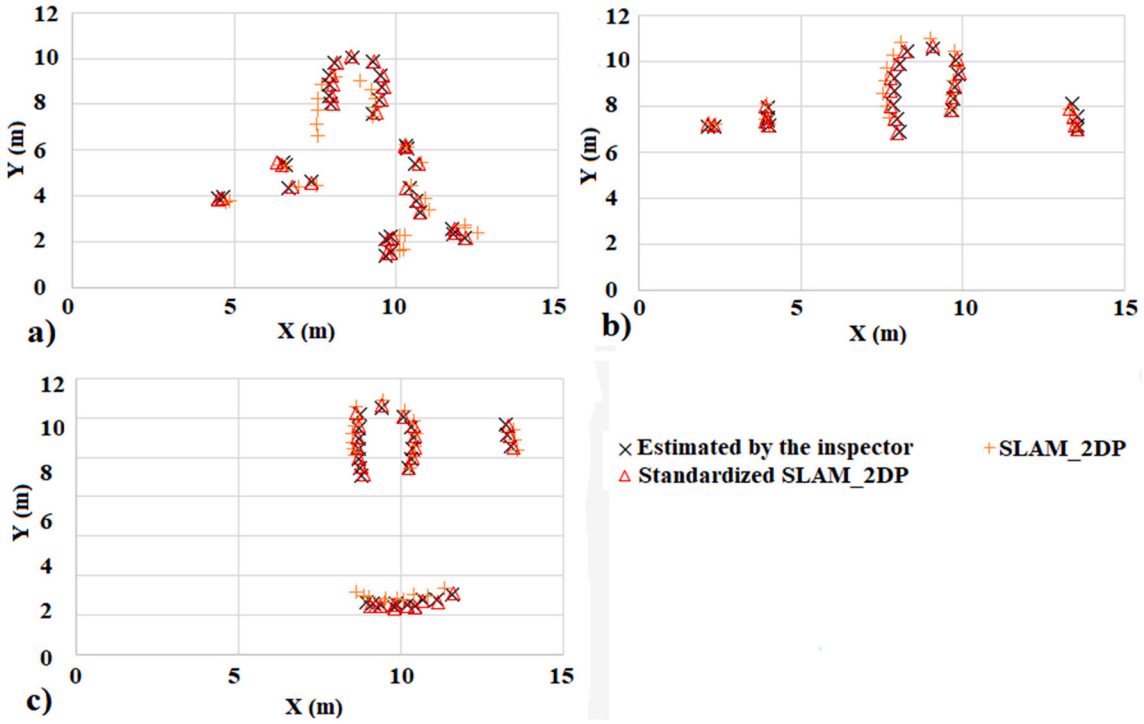


Fig. 18. Comparison of the SLAM_2DP, the standardized SLAM_2DP by PCA_STAN_ALGO and the estimated by the inspector in the coordinate system (selected locations): a) Case 2; b) Case 3; c) Case 4.

as close to our target paths as possible (the closeness between an aligned source path and the target path was evaluated by visual comparison). The experiment was conducted by one of the authors in the developed GUI tool (Chapter 3).

An inspection route can be divided into two parts: the floor part (F in Fig. 7) and the staircase part (S-1 and S-2 in Fig. 7). Figs. 15–17 show that the floor part of a SLAM_2DP is actually close to that of the standardized SLAM_2DP. Therefore, we took the following three steps to conduct a test:

1. Deleting the y coordinates of a source path, and then loading the resultant 2D source path onto the floor map. If using both PCA_STAN_ALGO and GRPX_TRANS, the source path needed to be standardized before its y coordinates were removed.
2. Selecting and moving two points from the 2D source path so that its floor part was near the floor part of the target path as much as possible. The reflection operations should be done beforehand.
3. Removing the 3D-to-2D scale drift from the staircase part of the 2D source path. At least two points needed to be relocated, such as the starting point and a point from the landing of the stairs (see Fig. 7). This step was only required for using only GRPX_TRANS.

In each case, 1) there were two tests for each method (four tests in total), and 2) it was not allowed to use a point for the path alignment that had been used in the previous tests. The time spent was recorded and RMSE was calculated for each test. The results are presented in Table 2.

Compared with using only GRPX_TRANS, using both PCA_STAN_ALGO and GRPX_TRANS saved about 51.4% of time for Cases 2–4 and achieved a higher accuracy. The 2-point scheme has the best time saving performance. Even if only 12 tests were complete, this simple experiment is worthwhile as it indicates that methodology is one of key factors on the issue of time saving for the path alignment.

In addition, an excess of 69 cases were further tested (RMSE was not calculated). There were two tests for each case: the first test for using both PCA_STAN_ALGO and GRPX_TRANS, and the second test for using only GRPX_TRANS (the aligned path obtained from the first test was used as a reference to guide the second test). On average 2.12 points (we counted the number of pivot points used for the path alignment because a torque point will eventually become a pivot point) were needed for the first test while the number increased to 4.51 for the second test. Only for four cases, more than two points were needed for successful GRPX_TRANS in the first test. These four cases experienced similar problems with video quality, where OpenVSLAM got lost permanently at some point and had to be restarted again. This in effect cuts the path into two different parts and each has its own initial coordinate system. In general, it took less than 10 min for the first test in each case, while it took the double time for the second test. About 50% time and workload were saved.

The issue about the orientation of a camera is not covered in this paper. The main reason is that the 360-degree camera is widely adopted as it can improve the quality of digital inspection and solving the vSLAM by providing a large visual field of view. This also makes the orientation of the camera less important than the location. However, when using a conventional camera, the orientation is important. PCA_STAN_ALGO can be directly applied to deal with the issue of orientation in the same way as that for the location.

PCA_STAN_ALGO can be used to remove the 3D-to-2D scale drift, but the scale drift caused by the depth scaling ambiguity in monocular SLAM may still exist, however, the accuracy produced by OpenVSLAM and the proposed methodology in this study is accurate enough because:

- an indoor inspection route is comprised of many small loops (e.g., the inspection route in a room forms a loop), and OpenVSLAM is capable of detecting and closing loops. To some extent, this reduces the scale drift.
- Very high accuracy of the paths reconstructed by SLAM in indoor construction inspection is not desirable. Rather than the precise

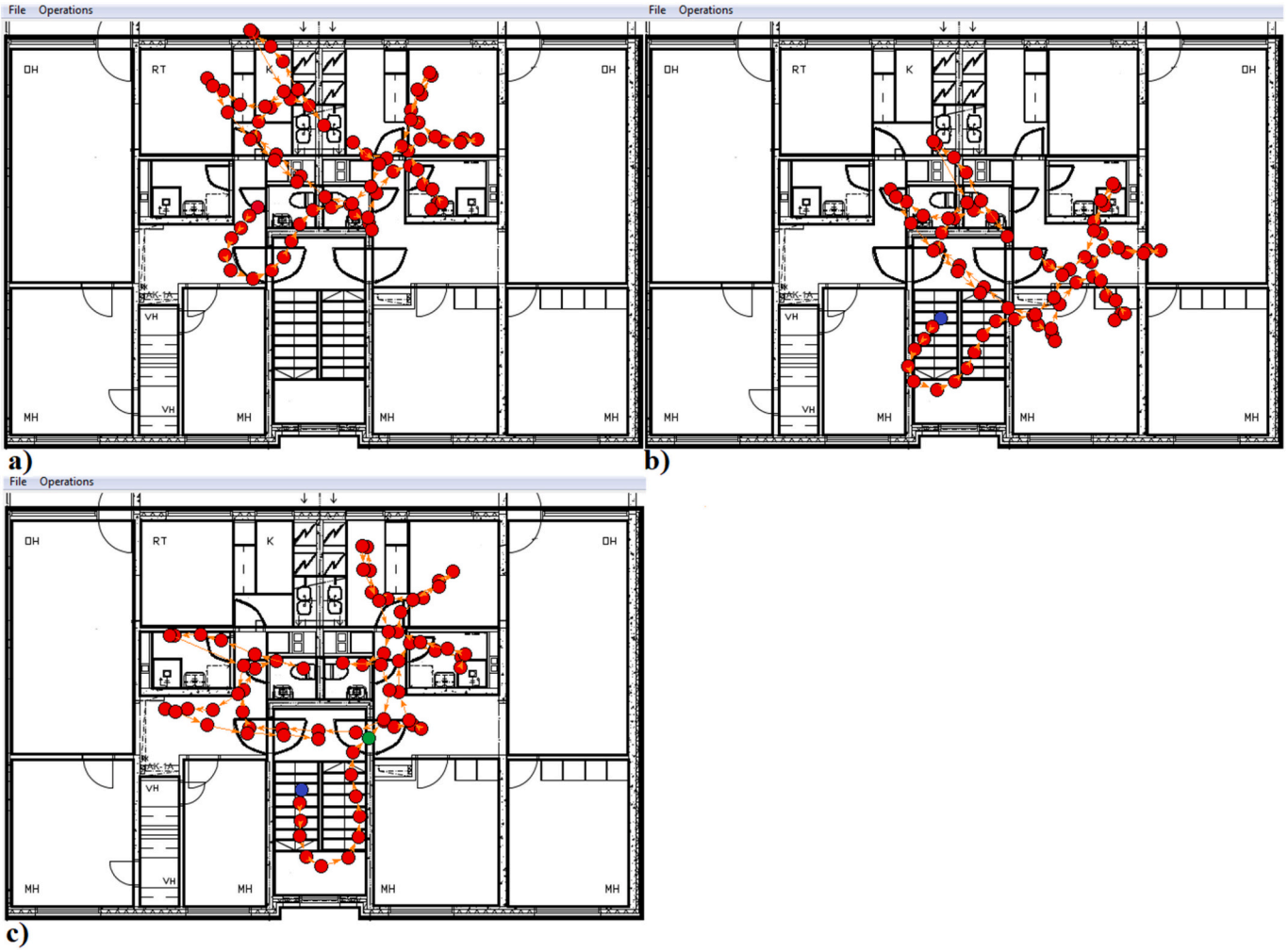


Fig. 19. Demonstration of the 2-point scheme for the standardized SLAM_2DP from Case 2. a). Initially estimating the scaling based on the size of the floor map (there is no reflection for this case. Reflection operations can be selected from the menu "Operations"). b) Moving the starting point (blue color) to the desired location as a pivot point. c) Moving the green point as a torque point to a correct location so that the branches of the path are well fitted into the rooms of the two flats. In this step, the rotation and scaling (larger than the initial estimation) are estimated.

Table 2
Comparison of time spent for Cases 2–4 using only GRPX_TRANS and using both PCA_STAN_ALGO and GRPX_TRANS.

		[a]The number of points used	The time spent (mins)	RMSE
Case 2	Using only GRPX_TRANS	4.5[b]	4.6[b]	0.17 (m) [b]
	Using both PCA_STAN_ALGO and GRPX_TRANS	2[b]	2.2[b]	0.094 (m) [b]
Case 3	Using only GRPX_TRANS	4[b]	4.3[b]	0.14 (m) [b]
	Using both PCA_STAN_ALGO and GRPX_TRANS	2[b]	2.1[b]	0.093 (m) [b]
Case 4	Using only GRPX_TRANS	4.5[b]	4.5[b]	0.19 (m) [b]
	Using both PCA_STAN_ALGO and GRPX_TRANS	2[b]	2.2[b]	0.096 (m) [b]

[a] When stopping moving a torque point, this torque point will become a pivot point. Therefore, we finally counted the number of pivot points used for the path alignment.
[b] The average value of two tests.

location of the camera, the customers are more interested in which room the current scene of a video belongs to and where the inspection route starts. Thus, it will provide sufficient accuracy for the customers to observe indoor construction processes if the branches (loops) of an indoor construction inspection route can be properly located in the target rooms and (public) areas (e.g., stairs). GRPX_TRANS can accomplish this task with sufficient accuracy.

Even more, despite the good accuracy, higher or ultra-high accuracy than 'good' one can be attained by applying the corrections to the existing scale drift. For instance, in the future, additional sensors will be used to assist vSLAM in improving the accuracy and quality of the path reconstruction (indoors and outdoors) and 3D maps (see Chapter 5 for the future work).

5. Conclusions and future work

A novel methodology was developed to transform a 3D path reconstructed by a vSLAM system onto a 2D map with a minimum effort. This study makes the following four original contributions:

1. A new type of scale drift (i.e., 3D-to-2D scale drift), which is often ignored in practice, is identified. The 3D-to-2D scale drift often

becomes noticeable in an area where the vertical movement of the camera is large (e.g., walking up the stairs in a building).
2. A PCA based algorithm (i.e., PCA_STAN_ALGO) is developed to resolve the 3D-to-2D scale drift. PCA_STAN_ALGO increases the possibility of realizing the 2-point scheme.
3. A GUI algorithm (i.e., GRPX_TRANS) is developed to facilitate the path alignment.
4. A novel methodology is proposed to couple PCA_STAN_ALGO and GRPX_TRANS together to implement the 2-point scheme.

The developed methodology was verified by five different case studies of indoor construction inspection. The results are promising and suggest:

- The xz plane of a camera system is not generally parallel to the floor plane due to human's spontaneous movement. Therefore, x and z coordinates present a path on the floor plane that experiences the 3D-to-2D scale drift. Because an inspector needs to handle many videos daily and is not familiar with every path, the orthographic projection of a SLAM path on the xz plane is often presumed undistorted. Sometimes this leads to false paths, for examples, wrongly located starting points can usually be seen on floor maps (Figs. 15-17).
- Not all 3D-to-2D scale drift problems result in false paths. However, when there is a significant variation in the y coordinates of a SLAM path in the initial camera system, a false path will be developed. We recommend using PCA_STAN_ALGO for all paths.
- Reflections over the x- or/and z-axis exist almost in every SLAM path (on the xz plane).
- If there is more than one pivot point, the choice of a torque point is crucial because GRPX_TRANS determines the locations of passive points based on the number of pivot points close to the torque point and the distribution of these pivot points. However, the 2-point scheme does not depend very much on the choice of the torque point, thus it is more reliable. It takes much more time to manipulate more than two points than to manipulate only two points.

The path alignment of SLAM reconstruction was not as straightforward as we originally thought it would be. This deserves more attention because improper path alignment could destroy the prior efforts. The biggest challenge that digital service companies are facing is that many inspectors lack basic knowledge of the principles of linear transformations. Many do not realize or do not know that (1) reflections often exist between two different coordinate systems, and (2) the 3D-to-2D scale drift can easily occur on a floor map even the SLAM path is correct. As a result, distortions or incorrect orientations often occur for some paths on floor maps. Some are difficult to detect. The methodology proposed in this paper helps inspectors solve these problems. The results are satisfactory in terms of time spent, accuracy of camera locations, and usability.

5.1. Future work

The novel methodology described her works effectively for indoor conditions. However, for outdoor conditions, PCA_STAN_ALGO cannot resolve the problem of scale drift, which is most likely due to scale ambiguity in monocular SLAM because outdoor inspection routes are large and have very few loops. Another issue is, even though 360-degree cameras are used, monocular SLAM can only build sparse 3D maps, which can hardly be used in practice, for instance, to compare 3D maps with BIM (building information modeling) models to check whether the design demands are met. In the future, these two issues will be targeted, and therefore additional sensors are needed. There are two possibilities: affordable RGB-D (e.g., Kinect) and LiDAR.

RGB-D cameras can provide good depth information. For instance, Kinect includes an infrared emitter and two cameras: one RGB camera and one infrared camera. The two cameras are used to simultaneously capture an RGB image and a depth image in the same scene [32]. The infrared emitter emits a fixed pattern of speckles to some object to estimate the 3D distance between Kinect and the object [32]. The 3D distance is then presented as the disparity image by the infrared camera. A LiDAR sensor illuminates the target with laser light and measures the reflection. Differences in laser return times and wavelengths can then be used to make digital 3D representations of the target. Pezzuolo et al. [33] carried out a metrological analysis of SfM (structure from motion) photogrammetry approach, low-cost LiDAR scanning and Microsoft Kinect v1 depth camera to 3D pig body measurement. Preliminary test results suggested that Microsoft Kinect v1 is the most cost-effective technique among three low-cost 3D techniques in terms of costs, working times and the scanning performance (noise, non-linearity and maximum detectable slope) [33].

Because OpenVSLAM supports RGB-D and Kinect is more cost-effective in comparison to LiDAR, Kinect will be selected for the future work. Another benefit that RGB-D can bring to indoor inspection is that it can produce photorealistic 3D models using SLAM. Tykkälä et al. [34] developed a process utilizing RGB-D input stream to produce standard 3D models and then exploring them virtually. In their work, the SLAM map was used as a geometrical backbone, and the Poisson method was employed to generate a watertight polygon mesh for that [34]. Because the mesh appearance was directly mapped from the keyframe image, the resultant 3D model is photorealistic [34]. Photorealistic 3D models offer high quality of visual inspection experience. Future research will address the issue of whether integrating existing resources, such as 360-dgree cameras and IMU, into Kinect can bring more benefits to photo-realistic 3D mapping for indoor conditions. Under outdoor conditions, 3D model may be not needed, and the combination of 360-degree camera and Kinect can provide more accurate inspection paths.

Declaration of Competing Interest

The authors declare no conflict of interest.

Acknowledgements

Video materials and floor maps were provided by Aiforsite Oy, which is much appreciated. The authors gratefully acknowledge the financial support received throughout STARCLUB Project (Grant No. 324023) from the Academy of Finland. The authors would also like to thank National Natural Science Foundation of China (Grant No. 41972324) for its partial support for this research.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.autcon.2021.103723.

References

[1] J. Yang, O. Arif, P.A. Vela, J. Teizer, Z. Shi, Tracking multiple workers on construction sites using video cameras, Adv. Eng. Inform. 24 (4) (2010) 428–434, https://doi.org/10.1016/j.aei.2010.06.008.
[2] M. Memarzadeh, M. Golparvar-Fard, J.C. Niebles, Automated 2D detection of construction equipment and workers from site video streams using histograms of oriented gradients and colors, Autom. Constr. 32 (2013) 24–37, https://doi.org/10.1016/j.autcon.2012.12.002.
[3] I. Brilakis, M.W. Park, G. Jog, Automated vision tracking of project related entities, Adv. Eng. Inform. 25 (2011) 713–724, https://doi.org/10.1016/j.aei.2011.01.003.
[4] M.W. Park, C. Koch, I. Brilakis, Three-dimensional tracking of construction resources using an onsite camera system, J. Comput. Civ. Eng. 26 (2012) 541–549, https://doi.org/10.1061/(ASCE)CP.1943-5487.0000168.
[5] T. Weerasinghe, J.Y. Ruwanpura, Automated multiple objects tracking system (AMOTS), in: Construction Research Congress, 2010, pp. 11–20, https://doi.org/10.1061/41109(373)2.
[6] M.Y. Cheng, J.C. Chen, Integrating barcode and GIS for monitoring construction progress, Automat. Construct. 11 (1) (2002) 23–33, https://doi.org/10.1016/S0926-5805(01)00043-7.

[7] H. Son, C. Kim, 3D structural component recognition and modeling method using color and 3D data for construction progress monitoring, Autom. Constr. 19 (7) (2010) 844–854, https://doi.org/10.1016/j.autcon.2010.03.003.
[8] S. Guo, C. Xiong, P. Gong, A real-time control approach based on intelligent video surveillance for violations by construction workers, J. Civ. Eng. Manag. 24 (1) (2018) 67–78, https://doi.org/10.3846/jcem.2018.301.
[9] H. Lingard, S. Pink, J. Hayes, V. McDermott, Using participatory video to understand subcontracted construction workers' safety rule violations, in: Proceedings of the 32nd Annual ARCOM Conference: Construction Work and the Worker Vol. 1, Association of Researchers in Construction Management, Manchester, UK, 2016, pp. 457–466, in: http://www.arcom.ac.uk/-docs/proceedings/4e39056b6026023e0f63e67e61c767c5.pdf.
[10] H. Ishimoto, T. Tsubouchi, Stereo vision based worker detection system for excavator, in: International Symposium on Automation and Robotics in Construction and Mining, 2013, pp. 1004–1012, https://doi.org/10.22260/ISARC2013/0110.
[11] S. Chi, C. Caldas, Image-based safety assessment: automated spatial safety risk identification of earthmoving and surface mining activities, J. Constr. Eng. Manag. 138 (2012) 341–351, https://doi.org/10.1061/(ASCE)CO.1943-7862.0000438.
[12] S. Thrun, Probabilistic Robotics (Intelligent Robotics and Autonomous Agents Series), The MIT Press, US, 2005 (ISBN: 9780262201629).
[13] J. Fuentes-Pacheco, J.R. Ascencio, J.M. Rendon-Mancha, Visual simultaneous localization and mapping: a survey, Artif. Intell. Rev. 43 (2015) 55–81, https://doi.org/10.1007/s10462-012-9365-8.
[14] H. Durrant-Whyte, T. Bailey, Simultaneous localization and mapping: part I, IEEE Robot. Automat. Magaz. 13 (2) (2006) 99–110, https://doi.org/10.1109/MRA.2006.1638022.
[15] N. Yang, R. Wang, D. Cremers, Feature-based or Direct: An Evaluation of Monocular Visual Odometry, ArXiv, 2017. https://www.researchgate.net/publication/316875491_Feature-based_or_Direct_An_Evaluation_of_Monocular_Visual_Odometry.
[16] D. Schubert, N. Demmel, V. Usenko, J. Stückler, D. Cremers, Direct Sparse Odometry With Rolling Shutter, arXiv, 2018, https://doi.org/10.1007/978-3-030-01237-3_42.
[17] J. Engel, V. Koltun, D. Cremers, Direct sparse odometry, IEEE Trans. Pattern Anal. Mach. Intell. 40 (3) (2018) 611–625, https://doi.org/10.1109/TPAMI.2017.2658577.
[18] R. Mur-Artal, J.M.M. Montiel, J.D. Tardós, ORB-SLAM: a versatile and accurate monocular SLAM system, IEEE Trans. Robot. 31 (5) (2015) 1147–1163, https://doi.org/10.1109/TRO.2015.2463671.
[19] E. Rublee, V. Rabaud, K. Konolige, G. Bradski, ORB: an efficient alternative to SIFT or SURF, in: Proceedings of 2011 International Conference on Computer Vision, Barcelona, 2011, pp. 2564–2571, https://doi.org/10.1109/ICCV.2011.6126544.
[20] ORB-SLAM Monocular. https://github.com/raulmur/ORB_SLAM, 2020 (Accessed date: 11 October 2020).
[21] DSO: Direct Sparse Odometry. https://github.com/JakobEngel/dso, 2020 (Accessed date: 11 October 2020).
[22] S. Sumikura, M. Shibuya, K. Sakurada, OpenVSLAM: a versatile visual SLAM framework, in: Proceedings of the 27th ACM International Conference on Multimedia, Nice, France, 2019, pp. 2292–2295, https://doi.org/10.1145/3343031.3350539.
[23] D. Schlegel, M. Colosi, G. Grisetti, ProSLAM: graph SLAM from a programmer's perspective, in: Proceedings of IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, 2018, pp. 3833–3840, https://doi.org/10.1109/ICRA.2018.8461180.
[24] R. Muñoz-Salinas, R. Medina Carnicer, UcoSLAM: Simultaneous Localization and Mapping by Fusion of KeyPoints and Squared Planar Markers, arXiv, 2019, https://doi.org/10.13140/RG.2.2.31751.65440.
[25] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, J. J. Leonard, Past, present, and future of simultaneous localization and mapping: towards the robust-perception age, IEEE Trans. Robot. 32 (6) (2016) 1309–1332, https://doi.org/10.1109/TRO.2016.2624754.
[26] S.J. Janke, Mathematical Structures For Computer Graphics, John Wiley & Sons, Inc., US, 2015 (ISBN: 978-1-118-71219-1).
[27] T. Botterill, S. Mills, R. Green, Correcting scale drift by object recognition in single-camera SLAM, IEEE Trans. Cybernet. 43 (6) (2013) 1767–1780, https://doi.org/10.1109/TSMCB.2012.2230164.
[28] D. Frost, V. Prisacariu, D. Murray, Recovering stable scale in monocular SLAM using object-supplemented bundle adjustment, IEEE Trans. Robot. 34 (3) (2018) 736–747, https://doi.org/10.1109/TRO.2018.2820722.
[29] Principal Component Analysis. https://en.wikipedia.org/wiki/Principal_component_analysis, 2020 (Accessed date: 11 October 2020).
[30] Rotation Matrix. https://en.wikipedia.org/wiki/Rotation_matrix, 2020 (Accessed date: 11 October 2020).
[31] D.C. Lay, S.R. Lay, J.J. McDonald, Linear Algebra and Its Applications, Pearson Education, Inc, 2016 (ISBN: 978-0321982384).
[32] G. Feng, L. Ma, X. Tan, Visual map construction using RGB-D sensors for image-based localization in indoor environments, J. Sens. (2017), https://doi.org/10.1155/2017/8037607.
[33] A. Pezzuolo, D. Giora, H. Guo, Q. Ma, S. Guercini, F. Marinello, A comparison of low-cost techniques for three-dimensional animal body measurement in livestock buildings, in: IOP Conf. Series: Earth and Environmental Science 275, 2019, https://doi.org/10.1088/1755-1315/275/1/012015.
[34] T. Tykkälä, A.I. Comport, J. Kämäräinen, Photorealistic 3D mapping of indoors by RGB-D scanning process, in: Proceedings of 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, 2013, pp. 1050–1055, https://doi.org/10.1109/IROS.2013.6696480.