IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 16 May 2024, accepted 29 July 2024, date of publication 2 August 2024, date of current version 12 August 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3437658

RESEARCH ARTICLE

Enhanced Monocular Visual Odometry: A Robust Appearance-Based Method for Accurate Vehicle Pose Estimation

R. RAJESH AND P. V. MANIVANNAN
Department of Mechanical Engineering, Indian Institute of Technology Madras, Chennai 600036, India

Corresponding author: R. Rajesh (me19d755@smail.iitm.ac.in)

This work was supported in part by Indian Institute of Technology Madras funded by Indian Government [Prime Minister Research Fellowship (PMRF)] under Grant SB22230168MEPMRF000758.

ABSTRACT Monocular Visual Odometry (MVO) is a fundamental element in autonomous navigation systems, providing vehicles/robots with the capability to estimate their positions by analyzing visual images from a single camera. This work delves into a pure appearance-based MVO algorithm that estimates the vehicle displacement and orientation between consecutive image frames alone, without using an Inertial Measurement Unit (IMU) sensor. The proposed method comprises four stages: ground spatial calibration, vehicle displacement, orientation estimation modules, and an actual vehicle heading estimation module. In the first stage, the image pixel coordinates are converted into world coordinates through ground spatial calibration. In the second stage, cross-correlation-based template matching is performed between two successive image frames and vehicle displacement is computed using the obtained world coordinates. Next, the orientation of the matched template is estimated along the 'u' and 'v' axis of the image. Subsequently, the actual vehicle heading is computed in the fourth stage with respect to the global coordinate system to estimate the vehicle pose. Experimental evaluations demonstrate the superior performance of the developed MVO algorithm compared to existing appearance-based methods that additionally utilize IMU to obtain orientation. When the vehicle is driven for a distance of 1406.35 meters, the average percentage distance error obtained is 1.41%, thereby highlighting the improved performance of the MVO algorithm in terms of higher accuracy and efficacy in real-world applications.

INDEX TERMS Autonomous navigation, ground spatial calibration, template matching, visual odometry.

NOMENCLATURE

(u, v) or (u_I, v_I)	center location of the reference template; in general, it is a pixel coordinate [pixel].
(x, y)	real-world coordinate of the (u, v) pixel coordinate [meter].
D	the measured distance between the vehicle starting line to the checkerboard bottom line [meter].
D_f	estimated distance of any image pixel with respect to v-axis [meter].
P_w	measured width of a single pixel [meter/pixel].
P_{wf}	estimated pixel width at any v-location [meter/pixel].
E_l	left edge of the checkerboard pattern at the bottom line [pixel].
E_r	right edge of the checkerboard pattern at the bottom line [pixel].
I_w	width of the captured image [pixels].
I_h	height of the captured image [pixels].
T_w	width of the reference template [pixel].
T_h	height of the reference template [pixel].
f	input image to the cross-correlation estimation.
$\bar{f}$	mean of the input image.
n	number of pixels [pixel].
t	template which is a small region taken from the input image [pixel].

The associate editor coordinating the review of this manuscript and approving it for publication was Jenny Mahoney.

© 2024 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

$\bar{t}$	mean of the template (t).
K	number of basis functions.
k_i	coefficient for the basis function.
(u_1, v_1) or (u_{I+1}, v_{I+1})	center location of the matched template; in general, it is a pixel coordinate [pixel].
(x, y)	real-world coordinate of the (u, v) pixel coordinate [meter].
$\overrightarrow{m_1}$	displacement vector from points (x_1, y_1) and (x, y).
$\overrightarrow{m_2}$	displacement vector from points (x_1, y_1) and $(1, y_1)$.
$\overrightarrow{n_1}$	displacement vector from points (0, 0) and (x, y).
$\overrightarrow{n_2}$	displacement vector from points (0, 0) and (x_1, y_1).
(X_i, Y_i)	estimated pose of the vehicle in the world coordinate system at time i [meter].
(X_{i+1}, Y_{i+1})	estimated pose of the vehicle in the world coordinate system at time $i + \delta i[meter]$.
N_{tf}	correlation coefficient of the FNCC process.
d	vehicle displacement [meter].
N	Number of image frames.

GREEK LETTERS

θ	estimated orientation with respect to u-axis [degree].
φ	estimated orientation with respect to v-axis [degree].
θ_i	previous orientation estimated with respect to u-axis for the i^{th} image frame [degree].
φ_{i+1}	current orientation estimated with respect to v-axis for the $i + 1^{th}$ image frame [degree].
ψ	estimated actual vehicle heading [degree].

I. INTRODUCTION

Providing a vehicle with Autonomous Driving capability relies heavily on accurate pose estimation in real-time. Pose estimation or odometry is used to estimate the vehicle's current location (x, y, z) and orientation (θ) about the starting point (assuming the origin). Primarily, visual information is the first and foremost thing for a human driver because visual cues such as other road participants' size, shape, color (i.e., car, bus), road signs, traffic lights, and lane markings are easily perceived. However, 3D information can be obtained from non-visual sensors like Light Detection And Ranging (LiDAR), ultrasonic sensors, and RAdio Detection And Ranging (RADAR); the important visual cue (i.e., color) is missing; hence, Visual Odometry (VO) plays a vital role [1] in autonomous navigation. VO is a method used in computer vision and robotics to estimate the motion of a camera (installed on the vehicle/system) by analyzing the consecutive images perceived by the environment.

In general, based on the vision sensor being used, VO can be divided into Monocular Visual Odometry (MVO), Stereo Visual Odometry (SVO), and RGB-D VO. MVO uses 2D images acquired from a single camera to estimate pose; however, it suffers from scale ambiguity and lack of depth information, which restricts its direct use in odometry estimation. Recent research has shown positive signs in deriving the depth from 2D images using various sophisticated approaches [2], [3]. MVO uses a camera model and imaging geometry (i.e., traditional method) to compute depth information. Moreover, MVO is an economical solution with notable advantages over other approaches. In contrast, SVO [4], [5], [6] uses the disparity information between two horizontally displaced monocular cameras to estimate the depth more accurately than MVO. RGB-D VO uses a monocular camera along with a depth sensor to accurately estimate 3D information about the environment.

Next, based on the image-processing algorithm, the approaches can be classified into indirect, direct, semi-direct, and appearance-based VO. The indirect method [7], [8], [9] is also known as the feature-based approach, which relies on image features, such as corners, edges, and blobs. This method makes odometry estimation robust to illumination variations since it uses illumination-invariant feature extraction methods. In the case of the direct method, raw pixel intensity values are used rather than the image features. This method estimates pose by minimizing the photometric error, which measures the difference between image intensities in consecutive frames [10], [11], [12]. The direct method requires more computational power (typically implemented in GPU); however, it is robust in featureless environments. The semi-direct method combines both indirect and direct approaches; it uses a small number of key points or features to aid in motion estimation while also considering raw image intensity data [13]. Finally, the appearance-based approach uses a cross-correlation technique between successive image frames (well known as template matching) to estimate the displacement. In template matching, a small template is searched in the next frame to compute the displacement [14], [15]. In addition, some methods use IMU for orientation estimation, and displacement estimation using images is termed visual-inertial odometry [15], [16], [17], [18].

In the present work, a novel Monocular Camera-based Visual Odometry (MVO) that uses the Ground Spatial Calibration (GSC) technique has been developed and tested

to estimate the displacement and orientation of a vehicle for autonomous driving accurately. The key components of the developed MVO are as follows:

(1) *Ground spatial calibration:* It establishes the relationship between pixel coordinates and metric coordinates irrespective of the image geometry, which helps in direct depth estimation at a reduced computational cost.

(2) *Displacement estimation module:* This module estimates the vehicle displacement with the GSC-established

relation that uses pixel coordinates having a maximum correlation coefficient between the reference and matched template.

(3) *The orientation estimation module* estimates the orientation of the vehicle using a matched template between consecutive road image frames obtained from a monocular camera. Furthermore, *actual vehicle heading estimation* has been introduced for an accurate representation of the vehicle heading about the global frame.

II. RELATED WORK

MVO has various advantages over other VO approaches, such as cost-effectiveness and reduced hardware and computational complexity. The authors of [19] discussed the problem of VO, its basic principle, and a few existing algorithms that can compute the camera pose using consecutive images obtained from a monocular camera.

A. FEATURE-BASED METHODS

Davison [7] successfully demonstrated a pure vision-based 3D trajectory estimation method called MonoSLAM for an HRP-2 high-performance full-size humanoid robot using live augmented reality (AR). In their study, the available indoor features were tracked and updated on a 3D map. Subsequently, on the updated map, an AR object is inserted at each time frame, thus helping the system compute the orientation. The downside of this approach is that it works well under feature-rich indoor conditions. However, for outdoor conditions, an improved feature selection approach based on the survival of the fittest strategy was presented in [9]. The authors used ORB features (Oriented FAST and Rotated BRIEF) for pose estimation along with loop closing to optimize the pose graph and update it in the map in real-time. This algorithm (ORB-SLAM) has been extensively tested with the KITTI dataset, and it has been reported that the trajectory error varies between 0.3% and 5%, and the error increases when loop closure is not implemented.

An indoor landmark-feature-based approach was proposed in [20]. In this study, an improved scale-invariant feature transform (iSIFT) descriptor was used to recognize landmarks from the RGB-D dataset. Using the iSIFT descriptor, the computation time for feature recognition was reduced to 0.0583 s, which was less than that of the other descriptors. However, the Root Mean Squared Error (RMSE) of the trajectory is 0.03 m for every 4-meter travel distance (appx). A particle filter was used to recognize the features for odometry estimation proposed in [21]. In this approach, the position error decreased when the number of particles increased. Since the indoor environment is semi-structured, the number of available features is higher; hence, the number of particles is also high, which ensures fewer positional errors. A semi-direct VO was proposed in [13] for large-scale outdoor localization. In this study, Features from Accelerated Segment Test (FAST) descriptors were used to obtain more corner points, and the Lucas-Kanade (LK) method was used to establish the correspondence between consecutive frames. When this modified approach is compared with ORB-SLAM on the KITTI dataset, it is evident that the modified approach has a similar position error and orientation error, with a slight reduction in computation time. To reduce the drift error, an Extended Kalman Filter (EKF) based tightly coupled MVO was proposed in [16]. Furthermore, a local bundle adjustment was implemented based on the visual-inertial map points. A Learning Kalman Network (LKN) was implemented for an MVO in [22]. This approach showed an average translational RMSE of 2.11% and a rotational RMSE of 1.05°/100 m. In [23], a feature-based MVO tested in a harsh environment was presented, which focused on improving the trajectory accuracy by fusing the magnetometer and gyroscope data to estimate an accurate robot orientation. The test result shows that their approach reduces the average closed loop error to 0.93m, which is less than the error (7.89 m) obtained with wheel odometry. A similar algorithm was tested in a complex agricultural environment [24]. This algorithm uses IMU for orientation estimation and was tested on a dataset that contains two 180° turns, and the average trajectory error observed was found to be 10.84 m. An error relaxation model was introduced in a recently presented study [25] for frame-to-frame VO estimation. The error relaxation model utilizes a bidirectional loss function, and a pose correction function is used to reduce the drift error introduced in the initial estimation. The authors tested this method with the KITTI dataset and achieved an average translation error of 6.63%, which is better than that of the existing methods. However, the rotation error was higher (2.67°/100 m).

B. APPEARANCE-BASED METHODS

An appearance-based VO for the standard Toyota Prado SUV and autonomous industrial forklift was presented in [26]. A ground-facing camera is used to compute the camera displacement from the template matching on the low-textured images, and the orientation is computed from the Ackerman steering model, which helps simplify the calculation process and time. From the test results, it can be noted that this method achieved a positional error of 4% and a heading error of 35° for a test distance of 100 m. In [27], a linear forward prediction filter was used to find the best template from multiple templates to estimate the displacement between consecutive images in a shorter computational time. A multi-camera approach was presented in [28], where the displacement was computed from the ground-facing camera, and the orientation was computed from the template matching on the horizontally installed camera. However, the major drawback of this approach is that it performs dual-template matching, which makes the approach computationally expensive. In addition, the mean distance error was 2.23% of the total travel distance, and the heading error was 9.34°. An adaptive-search template matching based on vehicle acceleration for MVO was presented in [29]. This adaptive technique reduces the template search time by approximately 87% and improves the quality of matching compared with the full search method.

TABLE 1. Merits and demerits of a visual localization system.

Classification	Merits	Demerits	Typical studies
Single camera + feature-rich environment	• Works well for indoor conditions • Robustness to lighting changes	• Application is limited • Trajectory error is high	[7] Davison et al. [20] Sharma et al. [13] Naixin et al. [23] Kim et al.
Single camera + cross-correlation + limited feature environment	• Low cost • Suitability to texture-less environments • Continuous motion estimation	• Computationally expensive • Challenging orientation estimation makes it rely on additional proprioceptive sensor • The accuracy is less for long-run experiments	[26], [27] Nourani et al. [28] Gonzalez et al. [29] Aqel et al. [30] Birem et al.
Single camera + cross-correlation + IMU	• Robustness to occlusions and lighting changes • Scale estimation is possible • Accuracy is high	• Cost ineffective • Complex sensor fusion algorithms • Accurate calibration is needed	[15] Zeng et al. [24] Song et al.

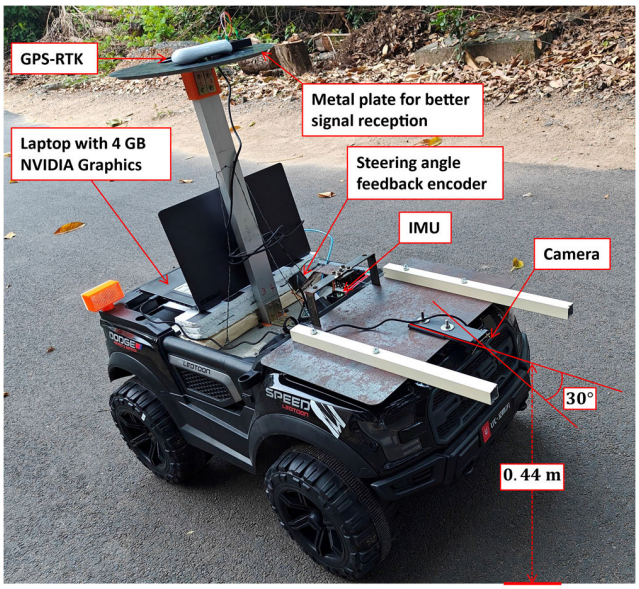

FIGURE 1. The experimental data collection platform contained a monocular camera, IMU, and GPS-RTK. The platform movement is controlled by a joystick.

Instead of template matching, Fast Fourier Transform (FFT)-based displacement estimation was presented in [30], which reduces the computation time. In addition, the rotation is estimated from the simple idea of the FFT output obtained on specific regions of the image frame, instead of taking the FFT of the whole image. This technique reduces the distance error by 1.33% for a total distance of 20 m, and the rotation error is 17.2057°. Zaman et al. [14] developed a cost-effective MVO solution for agricultural environments. In this work, the authors used the normalized cross-correlation (NCC) technique to match the templates between images for displacement estimation, and the orientation was computed as the angle between the templates. It was found that as the template size increased, the orientation error decreased; however, there was no significant improvement in displacement estimation. On the other hand, Zeng's approach in [15] is similar to that of [14]; however, an IMU is used for orientation estimation; which resulted in improved odometry estimation. This approach achieved a position error of 2.32% for an experimental travel distance of 335 meters.

The merits and demerits of the existing VO methods are presented in Table 1. It is ascertained that feature-based methods require feature-rich environments that restrict them to indoor applications. In contrast, appearance-based methods rely heavily on an additional sensor (IMU) for orientation estimation, which makes them cost-ineffective. Moreover, existing methods are tested for smaller travel distances, which makes it difficult to draw robust conclusions about their applicability for long travel distances. The major challenge with MVO is obtaining depth information from a single image using lightweight models that do not consume unnecessary computational power. Therefore, the MVO presented in this paper addresses the above challenges through a novel GSC for converting pixel coordinates into metric coordinates without unnecessary computation. Furthermore, the vehicle displacement is estimated purely from the captured images through a template-matching process along with a new orientation estimation method that uses images alone (no additional proprioceptive sensors are needed), thus making the presented method cost-effective and computationally effective.

III. MATERIALS AND METHODS

This section discusses the proposed Ground Spatial Calibration and vehicle displacement and orientation estimation modules for estimating odometry. Figure 1 illustrates an experimental vehicle equipped with various sensor modalities. The camera used in our data-collection platform is pre-calibrated to remove image distortions and the undistorted image is used for GSC, displacement, and orientation estimation modules.

A. GROUND SPATIAL CALIBRATION

In Computer Vision, the concept of spatial calibration is used to establish the relationship between camera coordinates (pixels) and real-world coordinates (metric units). This relationship is only valid if the camera is fixed at a particular height and orientation. The present work extends this concept as a Ground Spatial Calibration (GSC) technique for estimating the distance and orientation using monocular vision. The objective of GSC is to establish a function that converts any pixel coordinates (u, v) with sub-pixel accuracy into

real-world coordinates (x, y) in meters. To perform this function, a monocular camera was fixed onto the vehicle using an additively manufactured plastic camera holder, as shown in Figure 1. Unlike regular camera placement (as in [14], [15], [26], [27], [28], [29], and [30], the camera is fixed at a pitch angle of 30° (facing the ground), which facilitates capturing more road surface features for implementing the MVO. Hence, this method enables the estimation of vehicle displacement at a higher traveling speed. To perform the GSC, a checkerboard pattern of size 0.297 m x 0.42 m was used (Figure 2). During calibration, the distance between the camera and the checkerboard pattern is varied by moving the checkerboard relative to the camera (i.e., the camera is fixed), and images are captured. Therefore, the scene area captured by the camera is constant (i.e., (u, v) pixel coordinates); however, the size of the checkerboard pattern projected onto the image sensor varies. Hence, by knowing the v-axis location of the checkerboard, it is possible to estimate the distance (D) at which the checkerboard pattern is placed. In Figure 2, it can be seen that the bottom center (marked as a RED dot) of the checkerboard pattern is placed at the bottom center of the image (for the present case, it is $(u, v) = (640, 720)$). By keeping the checkerboard position constant in the horizontal direction (i.e., no movement in the u-axis) and moving the checkerboard along the v-axis, the distance moved by the board (in meters) could be recorded. Since, a 2D image is a perspective projection of a 3D real-world scene, the distance between the camera and real-world points exhibits an exponential relationship (represented as a BLUE curve in Figure 3). The measured distances (D) versus the v-locations are shown in Figure 3. Using exponential curve fitting, calibration equation (1) is obtained. Hence, using equation (1), distance of any image pixel with respect to the v-axis (i.e., distance D_f) can be computed.

$$D_f = 0.913e^{-0.006709v} + 1.198e^{-0.001197v} \quad (1)$$

Subsequently, the pixel width (i.e., the distance between two pixels) relationship can be obtained from the number of pixels occupied by the checkerboard at various v-locations (GREEN line) as shown in Figure 2. To obtain this relationship, effective pixels that capture the actual width of the checkerboard pattern (0.42 m) are required. This can be obtained from the difference in pixels (482.23 pixels) between the left (E_l) and right edge (E_r) pixel locations of the checkerboard pattern at 72.6 cm distance.

It should be noted that, before calculating the difference (in number of pixels), E_l and E_r for all the v-locations are linearized using the least squares method to compute the pixel width with sub-pixel accuracy.

$$P_w = \frac{checkerboard\ pattern\ width}{number\ of\ pixels} \quad (2)$$

The calculated $\boldsymbol{P_w}$ using equation (2) is the pixel width data at various $\boldsymbol{v}$-locations. From Figure 3, it can be seen that the pixel width data at various $\boldsymbol{v}$-locations exhibit the exponential trend (RED curve) and the obtained relationship can be seen

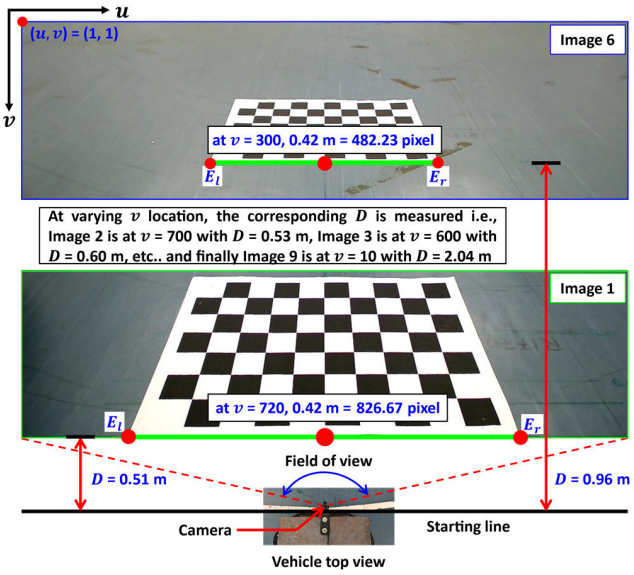

FIGURE 2. Ground-spatial calibration setup. D is the actual distance between the vehicle and respective v-pixel coordinates. The actual width of the checkerboard pattern in the image is marked with a GREEN line.

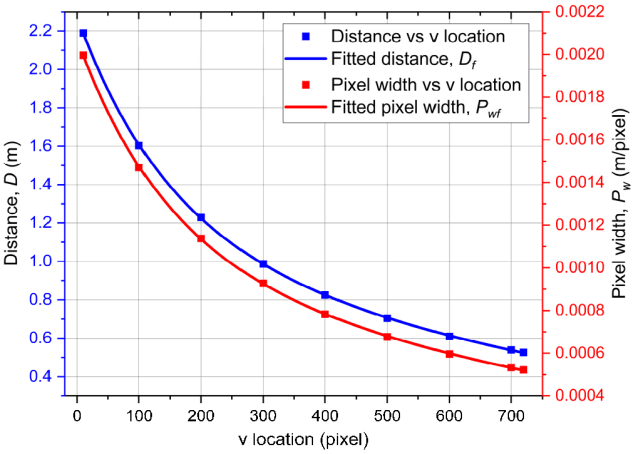

FIGURE 3. Ground Spatial Calibration results. The RED curve shows the fitted pixel width and the BLUE curve shows the fitted distance from the vehicle starting line. Both sets of data exhibited an exponential relationship.

in equation (3). Using equation (3), the fitted pixel width $\boldsymbol{P_{wf}}$ at any $\boldsymbol{v}$-location can be computed.

$$P_{wf} = 0.0008189e^{-0.006648v} + 0.00108e^{-0.00105v} \quad (3)$$

By using equation (1) and equation (3), the distance (D_f) and pixel width (P_{wf}) can be computed for any v-location. In other words, if any image coordinates (u, v) are provided, they can be converted into physical units (i.e., (x, y) in meters) which is an important requirement for odometry estimation using monocular vision. Unlike depth-estimation network methods [2], [3], the proposed GSC method directly converts the required pixel coordinates into metric units without processing the entire image, which generally consumes unnecessary computational power and time.

1) GSC EVALUATION

To assess the accuracy of the GSC used for displacement estimation through template matching, ten random pixel locations $(\boldsymbol{u}, \boldsymbol{v})$ have been chosen, and their real-world locations $(\boldsymbol{x}, \boldsymbol{y})$ are manually measured to establish ground truth. Subsequently, the GSC is applied to estimate real-world locations based on pixel locations using equations (1) and (3). The pixel locations, their corresponding ground truth locations, and estimated locations using the GSC are tabulated in Table 2. The evaluation metrics used to analyze the performance of the GSC included the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). The observed MAE for the presented data is 0.0225 m, and the RMSE is 0.0221 m. It is noteworthy to mention that the proposed GSC-based pixel-to-metric conversion exhibited a maximum distance measurement error of less than 2 cm, indicating the accuracy of the proposed technique. Consequently, under this criterion, the computed MAE for the matched template location is approximately 1.74 cm, and an RMSE of 1.64 cm. These findings highlight that the displacement estimation error with the proposed GSC technique is constrained to not exceed 1.74 cm for every 64.16 cm distance traveled in the case of the 120×120 template. This underscores the robustness and precision of the proposed GSC for pixel-to-metric conversion, affirming its reliability in odometry estimation.

B. MONOCULAR VISUAL ODOMETRY

As mentioned earlier, this paper aims to estimate the pose of the vehicle using the computed displacement and orientation between image frames obtained from a monocular camera. Since images are represented as pixels, they are converted into metric units through the ground spatial calibration as discussed in Section III-A. Subsequently, this section discusses the displacement estimation, orientation estimation, actual vehicle heading estimation, and pose estimation.

1) DISPLACEMENT ESTIMATION MODULE

Unlike feature-based approaches, this paper focuses on texture information as appearance-based methods do. To facilitate this, the monocular camera has been fixed on the vehicle at an angle facing the ground that captures consecutive images for estimating displacement. In general, the best way to find a match between two images based on texture is to perform template matching. Images are continuously captured at each time step. At the time 'i', from the captured image (I), the template to be matched is taken as a square area in the image having a width and height of 120 pixels $(T_w, T_h) = (120, 120)$ as shown in Figure 4 in RED dotted line. Then, in the next image frame $(I + 1)$ captured at a time '$i + \delta i$', the location of this template is identified (enclosed in BLUE line) based on the highest matching score. The direct template matching technique using Normalized Cross Correlation (NCC) is not computationally efficient. Hence, the FFT-based Fast NCC (FNCC) method is used for template matching, where the computational complexity of FFT is

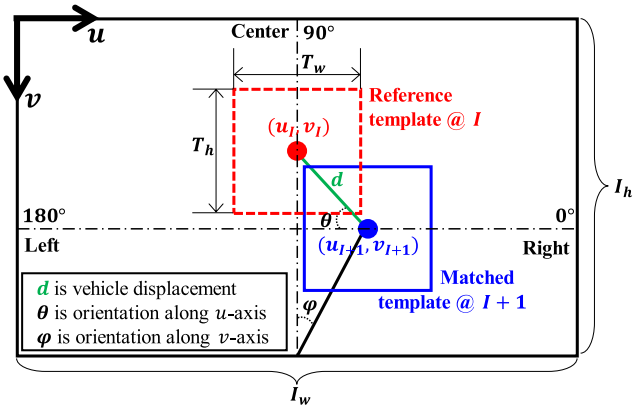

FIGURE 4. Illustration of template matching. I_w, and I_h are the width and height of the captured image which is 1280 × 720 pixels. The RED box is the template to be matched obtained from image (I) and the BLUE box is the matched template at the image $(I + 1)$. T_w, and T_h are the width and height of the template respectively. The GREEN line is the displacement (d) to be computed from the template matching which can be converted into metric units using GSC. The notations θ and φ denote the orientations computed with respect to u and v axes respectively.

$O(nlogn)$, which is lesser than the direct method (*i.e.*, $O(n^2)$). The FNCC-based correlation coefficient (N_{tf}) is calculated using equation (4).

$$N_{tf}(u,v) = \frac{\sum_{i=1}^{K} k_i(s(x,y) - s(x,y-1) - s(x-1,y) + s(x-1,y-1))}{\sqrt{\left\{\sum_{x,y}[f(x,y) - \overline{f_{u,v}}]^2 \sum_{x,y}[t(x-u,y-v) - \bar{t}]^2\right\}}} \tag{4}$$

where f is the input image, t is the template which is a small region taken from the input image, $\bar{f}$ is the mean of the input image (f), $\bar{t}$ is the mean of the template (t), K is the number of basis functions (i.e., zero mean template), k_i is the coefficient for the basis function, and N_{tf} is the correlation coefficient.

Using (4), the center location (u_{I+1}, v_{I+1}) of the matched template in the image frame $(I + 1)$ is obtained. Next, the displacement can be computed by following the procedure described below. The reference template's center point (u_I, v_I) coordinates are fixed at (640, 164). These center point location values in pixels can be converted into metric units (x, y) using equation (5) and equation (6), which are away from the actual starting location of the vehicle.

$$x = \left(\frac{I_w}{2} - u\right) * P_{wf} \tag{5}$$

$$y = y_1 = D_f \tag{6}$$

Next, the matched template's center point (x_1, y_1) which is (u_{I+1}, v_{I+1}) in the image frame is computed using equations (7) and (6) respectively. Any center location of the matched template's u- value is less than 640 pixels, it is considered a negative x-axis and vice versa according to

equation (8).

$$x_1 = \left(\frac{I_w}{2} - u_1\right) * P_{wf} \tag{7}$$

$$x_1 = \begin{cases} -x_1, & u_{I+1} > u_I \\ x_1, & u_{I+1} < u_I \end{cases} \tag{8}$$

Vehicle displacement (d) can be computed as the Euclidean distance between the center points in reference and matched templates (i.e., (x, y) and (x_1, y_1) using equation (9). It should be noted that for the forward movement of the vehicle, the d is positive and for backward movement d is negative according to equation (10).

$$d = \sqrt{(x - x_1)^2 + (y - y_1)^2} \tag{9}$$

$$d = \begin{cases} -d, & v_I > v_{I+1} \\ d, & v_I < v_{I+1} \end{cases} \tag{10}$$

Since the vehicle was operated at the speed of 1 m/s to 1.2 m/s, certain proportions of the captured images may have blurred. However, template-matching handles this effectively. Once the template is matched at $I + 1^{th}$ image, the $I+1$ image will become I and continue.

2) ORIENTATION ESTIMATION MODULE

Similar to existing methods that rely on the cumulative angle acquired from orientation along the $\boldsymbol{v}$-axis $(\boldsymbol{\varphi})$, literature has indicated an elevated level of position error in [14], [28], and [30]. Consequently, some approaches have opted to utilize orientation data sourced from IMU sensors [15]. Nevertheless, it is important to note that IMU sensors are susceptible to drift error, which invariably impacts the accuracy of position estimation. Hence, orientation data derived from both the $\boldsymbol{u}$-axis $(\boldsymbol{\theta})$ and $\boldsymbol{v}$-axis $(\boldsymbol{\varphi})$ is used to facilitate a more precise estimation of the vehicle's position. To compute '$\boldsymbol{\theta}$', first the pixel coordinates $(\boldsymbol{u_I}, \boldsymbol{v_I})$ and $(\boldsymbol{u_{I+1}}, \boldsymbol{v_{I+1}})$ are converted into metric units $(\boldsymbol{x}, \boldsymbol{y})$ and $(\boldsymbol{x_1}, \boldsymbol{y_1})$ respectively as discussed in the previous section. Using equations (11), (12), and (13) the '$\boldsymbol{\theta}$' can be computed.

$$\vec{m_1} = \begin{bmatrix} x - (-x_1) \\ y - y_1 \end{bmatrix} \tag{11}$$

$$\vec{m_2} = \begin{bmatrix} 1 - (-x_1) \\ y_1 - y_1 \end{bmatrix} \tag{12}$$

$$\theta = cos^{-1}\left(\frac{\vec{m_1}.\vec{m_2}}{\|\vec{m_1}\|\|\vec{m_2}\|}\right).\frac{180}{\pi} \tag{13}$$

While the vehicle is moving, if it turns toward the left side, $u_{I+1} > u_I$ and the 'θ' value is between 90° and 180°. Instead, if it turns towards the right side, the 'θ' is between 0° and 90° as per the equation (14).

$$\theta = \begin{cases} 180 - \theta, & u_{I+1} > u_I \\ \theta, & u_{I+1} < u_I \end{cases} \tag{14}$$

Next, the orientation along v-axis can be computed using equations (15), (16), and (17). Similar to equation (14), for the left-side movement, the 'φ' is negative, and for the right-side movement, it is positive according to equation (18).

$$\vec{n_1} = \begin{bmatrix} x \\ y \end{bmatrix} \tag{15}$$

$$\vec{n_2} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \tag{16}$$

$$\varphi = cos^{-1}\left(\frac{\vec{n_1}.\vec{n_2}}{\|\vec{n_1}\|\|\vec{n_2}\|}\right).\frac{180}{\pi} \tag{17}$$

$$\varphi = \begin{cases} \varphi, & u_{I+1} > u_I \\ -\varphi, & u_{I+1} < u_I \end{cases} \tag{18}$$

Individually, using 'θ' and 'd', the pose can be computed; however, the 'θ' computed along u-axis is limited to 45° between the reference and matched templates (i.e., the computed 'θ' will not exceed 135° during a 90° left turn, while during a 90° right turn, the 'θ' will not be less than 45°). Hence, pose estimation using 'θ' is accurate for straight movements and the vehicle heading is less than 45°. In contrast, the pose estimated with 'd' and cumulatively computed 'φ' along v-axis has a large straight-line error however, it accurately captures the rotation. Hence, in the present work, the orientation along u-axis and v-axis to estimate the pose of the vehicle.

3) ESTIMATING ACTUAL VEHICLE HEADING (ψ)

While estimating the pose of the vehicle, to compute the vehicle heading angle (ψ), 'θ' (orientation measured along the lateral axis) is used as 'ψ' when the vehicle is moving a straight line, and 'φ' (orientation measured along the longitudinal axis) value is added with or subtracted from 'ψ' for the successive frame when the vehicle turns left or right. This is tested with two cases 1) a left turn, and 2) a right turn.

a: CASE 1: LEFT TURN

Based on Figure 4, the actual vehicle orientation range (i.e., ψ) for a left turn span from 90° to 180°. However, analysis of the experimental dataset reveals that the orientation estimation module consistently provides estimates of the 'θ', as long as its value is within the range of 90° to 135° ($90° < \theta < 135°$), while the vehicle starts taking a left turn. At this condition, the φ_{i+1} will be greater than zero (as shown in Figure 5 'φvsN') then 'ψ' will be estimated as the summation of the previous orientation (u-axis) θ_i and current orientation (v-axis) φ_{i+1} according to equation (19).

$$\psi = \theta_i + \varphi_{i+1} \tag{19}$$

After computing 'ψ', update it as ($\psi = \psi + \varphi_{i+n}$) where n is the number of image frames. For the consecutive image frames the 'ψ' will exceed 180° (if the vehicle completed the left turn), and also the θ_i will be less than 90°. In that case, since the vehicle takes the left turn the vehicle pose with reference to the starting point will be in 90° offset. Hence, 'ψ' is estimated by adding 90° (i.e., $\psi = \theta + 90°$). The 'ψ' computed using equation (20) represents the accurate actual

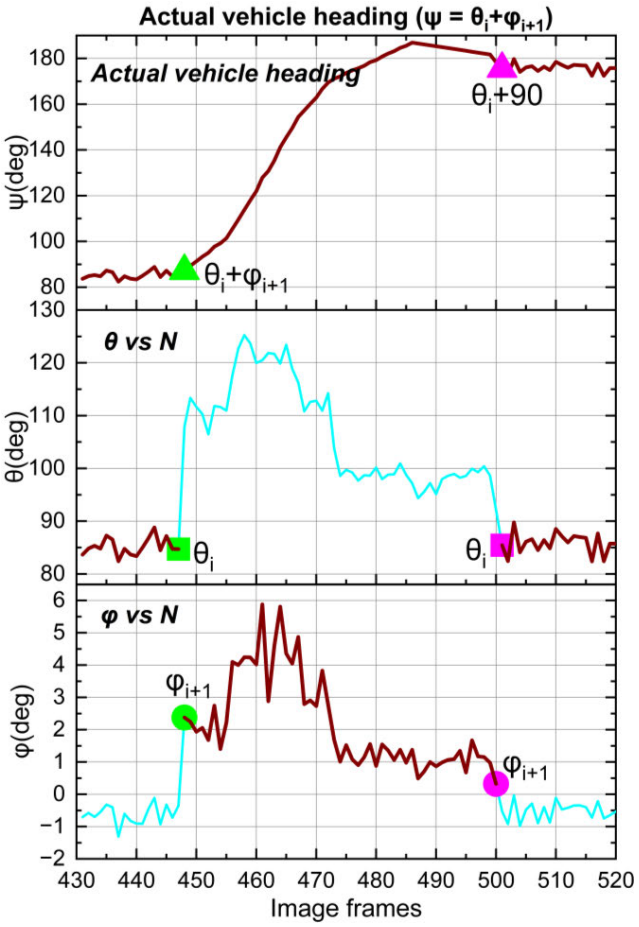

FIGURE 5. Actual vehicle heading with reference to the global frame. θ is orientation with respect to u-axis, φ is orientation with respect to v-axis, and N is image frame number.

vehicle heading angle.

$$\psi = \begin{cases} \psi, & (90^\circ < \theta < 135^\circ) \\ \theta + 90^\circ, & \psi > 180^\circ \end{cases} \tag{20}$$

b: CASE 2: RIGHT TURN

If the vehicle takes a right turn, the orientation varies between $90^\circ and 0^\circ$. Subsequently, the 'ψ' computed between consecutive image frames in such cases varies between 45° to $90^\circ (45^\circ < \theta < 90^\circ)$. Using equation (19), the 'ψ' can be computed, the only difference is that the orientation along v-axis (φ) is negative. Hence, the updated 'ψ' decreases towards 0°, which indicates that the vehicle has taken a right turn. Further, unlike (20), the 'ψ' is updated as $\theta - 90^\circ$ for the subsequent image frames if $\psi < 0$. With this, accurate pose estimation is achieved for the right turn.

Figure 5 helps estimate the actual vehicle orientation using the method described above. Let's assume that the vehicle is moving in a straight-line path; then the estimated θ_i is approximately close to 90^o; and corresponding φ_i is close to 0^o or slightly negative. However, if the vehicle is making a turn, φ_i will be positive for a left turn and negative for a right turn and its magnitude will be greater than 0^o for both cases. Since, the θ_i does not accurately capture the vehicle heading while turning, to accurately compute the current vehicle heading (ψ), which is the summation of the estimated first positive value of φ_{i+1} (marked as GREEN filled circle in sub-plot 'φ *vs* N'); and the previous θ_i (marked as GREEN square in sub-plot 'θ *vs* N').

4) VEHICLE POSE ESTIMATION

The computed displacement (d) and actual vehicle heading (ψ) between consecutive image frames from the previous sections are used to compute the vehicle pose. The starting location of the vehicle is considered as $(X_i, Y_i, \psi) = (0, 0, 90^o)$. Next, from the consecutive image frames this (X_i, Y_i, ψ) is estimated and updated using equation (21).

$$\begin{bmatrix} X_{i+1} \\ Y_{i+1} \end{bmatrix} = \begin{bmatrix} X_i + d * cos\psi \\ Y_i + d * sin\psi \end{bmatrix} \tag{21}$$

where, (X_{i+1}, Y_{i+1}) represents the estimated pose of the vehicle in the world coordinate system at the time '$i + \delta i$' and (X_i, Y_i) represents the estimated pose of the vehicle in the world coordinate system at time 'i'.

IV. EXPERIMENTAL VERIFICATION AND ANALYSIS

This section presents the methods used for experimental data collection and verification of the developed MVO algorithm. Furthermore, to demonstrate the efficacy of the developed algorithm, the results are analyzed qualitatively and quantitatively and are presented.

A. EXPERIMENTAL DATA COLLECTION

A Personal Computer (PC) powered by an AMD Ryzen 9 5900HS CPU clocked at 3.3GHz along with 16GB RAM, and a dedicated NVIDIA RTX 3050Ti graphics card with 4GB memory is used for multi-sensor data collection and algorithm testing. A monocular camera (IMX335 – 5MP) having 1280 × 720 pixels image resolution, BNO055 IMU (interfaced with Arduino DUE), and GPS-RTK are used as sensors and connected to the PC through USB serial ports. The experimental vehicle equipped with the above sensors to collect data is shown in Figure 1, and the data sampling frequency is presented in Table 3.

The experimental data is collected by driving the experimental vehicle on five different paths (both under indoor and outdoor conditions). The outdoor paths were composed of hot-mix asphalt. The indoor paths are made of BLACK dotted granite and mixed floors such as a few meters of fine cement floor, followed by tiled floor and granite floor, as shown in Figure 6. The multi-sensor data collection is performed using the script coded in MATLAB R2022a that stores the images captured by the camera (10 frames per second), orientation data from IMU for every 0.1 seconds, and GPS-RTK updates for every second. The GPS-RTK geographic coordinates (i.e., latitude, longitude, and altitude) are converted into local Cartesian coordinates using Gauss-Kruger projection

TABLE 2. Computed error for the GSC technique.

S. No	Pixel location (u, v) in pixels	Actual location (x, y) in meters	Estimated location using GSC (x, y) in meters	Squared error (m)	Absolute error (m)
1	(580,600)	(-0.032,0.085)	(-0.0354,0.0871)	0.00001597	0.0055
2	(350,500)	(-0.188,0.172)	(-0.1938,0.177)	0.00005864	0.0108
3	(100,400)	(-0.397,0.278)	(-0.414,0.291)	0.00045800	0.0300
4	(640,300)	(0,0.441)	(0,0.445)	0.00001600	0.0040
5	(1070,350)	(0.364,0.36)	(0.3559,0.3618)	0.00006885	0.0099
6	(900,500)	(0.176,0.171)	(0.173,0.177)	0.00004500	0.0090
7	(390,320)	(-0.206,0.402)	(-0.217,0.41)	0.00018500	0.0190
8	(370,170)	(-0.29,0.73)	(-0.315,0.7559)	0.00129581	0.0509
9	(1000,150)	(0.462,0.803)	(0.4408,0.8215)	0.00079169	0.0397
10	(780,100)	(0.197,1.06)	(0.195,1.016)	0.00194000	0.0460
Start	(640,164)	(0,0.78)	(0,0.775)	0.00002500	0.0050

TABLE 3. Experimental data collection parameters at 1 m/s.

Parameters	Values
Monocular camera sampling frequency	10Hz
IMU sampling frequency	10Hz
GPS-RTK sampling frequency	1Hz
Image resolution	720x1280 pixel
Camera installation height	0.44 meter
Camera facing angle	30°

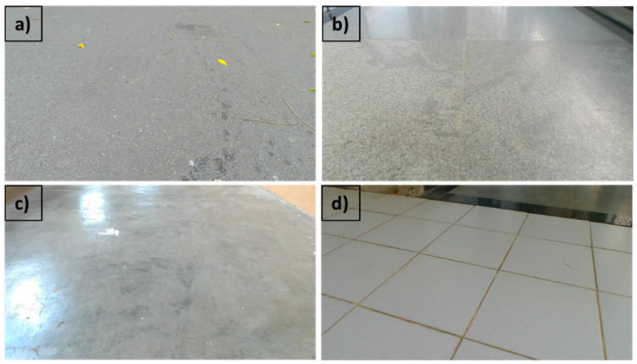

FIGURE 6. Images were collected in different ground conditions. a) Asphalt road, b) BLACK dotted granite, c) Cement floor, and d) Tile floor.

transformation [31]. The collected experimental dataset can be found here.

B. DATA PROCESSING AND ANALYSIS

The Ground Spatial Calibration (GSC) and Monocular Visual Odometry (MVO) algorithms explained in Section III-A and Section III-B have been coded and tested in MATLAB R2022a environment. During the experimental investigations, the developed algorithms are tested with three different-sized image templates such as 80×80 pixels, 120×120 pixels, and 160×160 pixels for a matching operation that computes the vehicle displacement and orientation. The vehicle pose was estimated using computed displacement and orientation values. Furthermore, the method proposed in this work provides better results using MVO alone, than the method proposed in [15], which uses the orientation data from the IMU along with the MVO.

The estimated vehicle pose (also called the trajectory) is shown in Figure 7a–11a. Furthermore, the vehicle position error in terms of the X and Y directions is shown in Figure 7b–11b. In addition, the percentage distance error is depicted in Figure 7c–11c. In all figures, the BLUE, GREEN, RED, MAGENTA, and BLACK lines represent the results obtained with 80×80 pixels, 120×120 pixels, 160×160 pixels template, MVO+IMU, and ground truth respectively. From the odometry results obtained for the outdoor path (Asphalt Road) which are shown in Figure 7 and Figure 8, it can be noted that the 120×120 pixels sized template performs better than other template sizes and MVO+IMU. The average error in the X and Y directions for the 120×120 template is 0.4148 meters and 1.1904 meters for the Path-1 dataset. Furthermore, the average distance error is 1.3286 meters for a total distance traveled of 470.31 meters. From Figure 7c, it can be seen that the distance error percentage over the total distance is the minimum (i.e., 0.44%) when compared with other template sizes. Next, for Path-2 (which is 636.14 meters long), with an 80×80 template size, the average error in X and Y directions is less (i.e., 0.4046 meters, 1.1103 meters) and the average distance error is 1.3062 meters, than with other template sizes which can be noted from Table-3. In both cases (Path-1 and Path-2), the use of a 120×120 template and 80×80 template results in a lesser error. In particular, the 80×80 template holds the second minimum error for Path-1 and shows similar performance to the 120×120 template for the Path-2 dataset. However, the 160×160 template shows a large maximum error in the X and Y direction, and also a maximum distance error. This is mainly due to the difficulty in precisely localizing the template in the actual image. Moreover, if the matched template position is having an error, the GSC output will also have an error since, it takes the center of the matched template (i.e., (u, v)).

From Figures 7a and 8a, it can be seen that the vehicle pose estimated with different-sized templates and the pose estimated using the orientation from the IMU, as in [15], matches the ground truth pose. However, if we see the error plots in Figures 7b and 8b, the magnitude of the errors in the

TABLE 4. Positioning errors observed for the outdoor dataset (Asphalt road). Lowest error highlighted in PURPLE and Second lowest error highlighted in MAROON.

Experimental scenes	Path 1				Path 2			
Traveling distance (m)	470.31				636.14			
Template size (pixels)	80x80	120x120	160x160	MVO + IMU	80x80	120x120	160x160	MVO + IMU
Average error in X-direction (m)	**0.4976**	**0.4148**	0.6892	3.9654	**0.4046**	**0.4146**	0.4462	2.3442
Maximum error in X-direction (m)	**1.9298**	**1.4398**	3.3745	8.7679	**2.3130**	**1.3303**	2.7695	7.9235
Average error in Y-direction (m)	1.3174	**1.1904**	**1.1994**	3.4326	**1.1103**	**1.1556**	1.8791	2.8113
Maximum error in Y-direction (m)	**2.5997**	**2.2872**	3.2485	16.745	**3.0630**	**2.0929**	4.4833	7.4324
Average distance error (m)	**1.5062**	**1.3286**	1.5343	6.1423	**1.3062**	**1.3350**	2.0400	4.1125
Maximum distance error (m)	**2.6447**	**2.3244**	3.9381	16.847	**3.1067**	**2.1247**	4.5889	8.6982

TABLE 5. Positioning errors observed for the indoor dataset (BLACK dotted granite, mixed cement & tile floor).

Experimental scenes	Path 3				Path 4				Path 5			
Traveling distance (m)	79.58				161.44				58.88			
Template size (pixels)	80x80	120x120	160x160	MVO + IMU	80x80	120x120	160x160	MVO + IMU	80x80	120x120	160x160	MVO + IMU
Average error in X-direction (m)	**0.2825**	**0.1707**	0.5438	0.4081	1.2430	**0.4390**	1.1821	**0.6382**	0.2268	**0.1186**	**0.0588**	0.1828
Maximum error in X-direction (m)	**0.6597**	**0.3829**	0.9425	0.8403	6.6320	**2.0584**	5.4018	**3.9754**	1.9781	0.9918	**0.3311**	**0.8714**
Average error in Y-direction (m)	0.0415	**0.0334**	0.0504	**0.0320**	0.4708	**0.2083**	**0.3678**	1.0156	0.9325	**0.7580**	**0.2422**	1.0625
Maximum error in Y-direction (m)	0.4910	**0.4408**	**0.4408**	**0.2812**	5.9880	**1.0909**	**2.2634**	4.1351	2.1134	**1.2754**	**0.5160**	1.8826
Average distance error (m)	**0.2941**	**0.1811**	0.5522	0.4128	1.5841	**0.5805**	**1.4151**	1.4941	1.0588	**0.8019**	**0.2651**	1.1675
Maximum distance error (m)	**0.6719**	**0.4408**	0.9441	0.8410	6.7504	**2.0594**	5.6716	**4.6324**	2.1163	**1.2760**	**0.5549**	1.8833

X and Y directions is less for the pose estimated with the 120×120 template. Furthermore, it can be noted that the pose estimated using the IMU orientation data shows maximum X, Y, and distance errors. This is due to IMU data inconsistency and drift error (i.e., error accumulating), and this error is predominant when the vehicle takes a turn (left or right). However, in the present work, as discussed in earlier sections (Sections III-B2 and III-B3), the orientation is computed only with MVO data (using subsequent image frames), thereby avoiding the drift problem. The positioning errors observed for the outdoor conditions are listed in Table 4. In the table, the values highlighted in PURPLE have the lowest error, and those highlighted in MAROON have the second lowest error.

Next, the MVO algorithm was tested on the indoor conditions with the image collected at different floor conditions. Figure 9 shows the results of straight maneuvering (i.e., Path-3). From Table 5 it can be noted that the position obtained with the use of a 120×120 pixel template shows the lesser average and maximum errors in the X direction and also the distance error. Whereas the MVO+IMU method shows the lesser average and maximum errors in the Y direction. Further, with the 120×120 pixel template, the Y error observed is the second lowest. Furthermore, the error in the Y direction is minimal and the X direction is higher with the MVO+IMU. The lower Y error can be attributed to the accurate estimation of the displacement with template matching, which is based on GSC. However, the X error is higher (shown in the MAGENTA line in Figure 9b), since the IMU sensor used in the present work shows a larger orientation error when the vehicle is static. Once the vehicle starts moving, the IMU captures the relative orientation of the vehicle with higher accuracy ($\pm 0.21°$). On the contrary, the relative orientation error estimated with the MVO is slightly larger ($\pm 0.52°$).

Subsequently, the use of a 160×160 template results in higher X, Y, and distance errors than with templates of

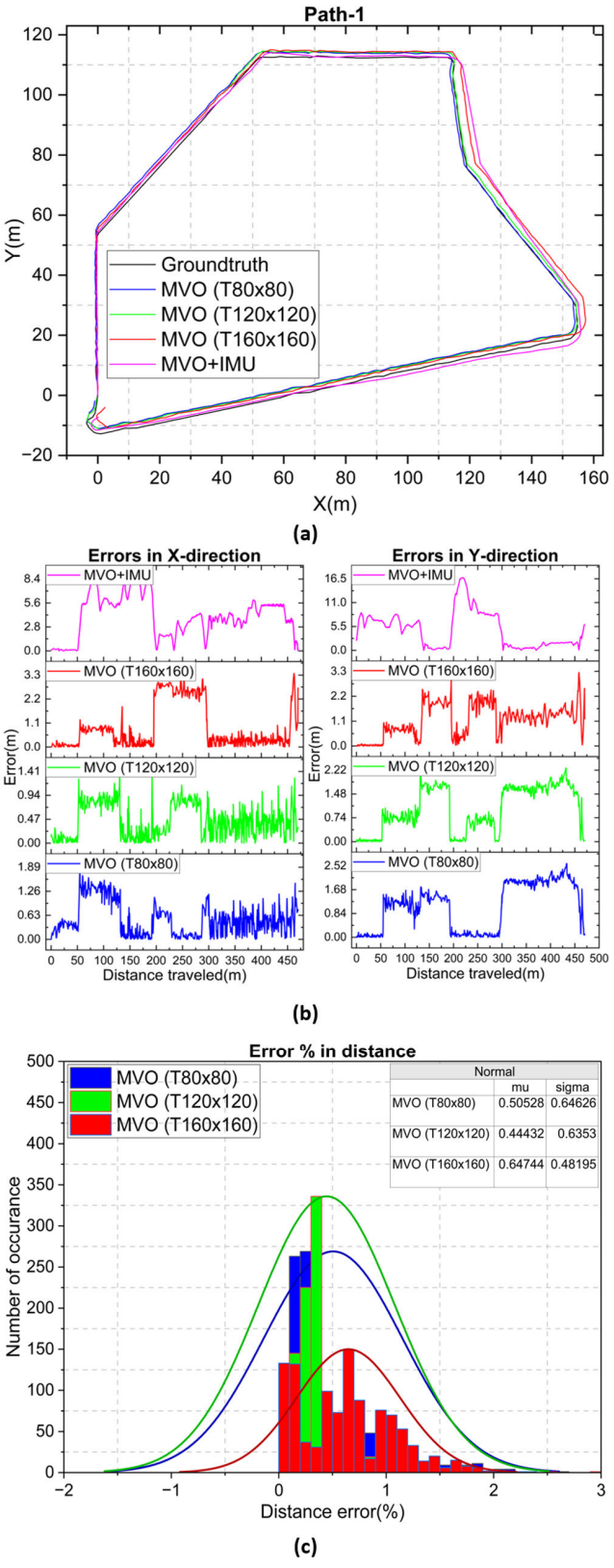

FIGURE 7. Estimated vehicle position and errors for outdoor Path-1.

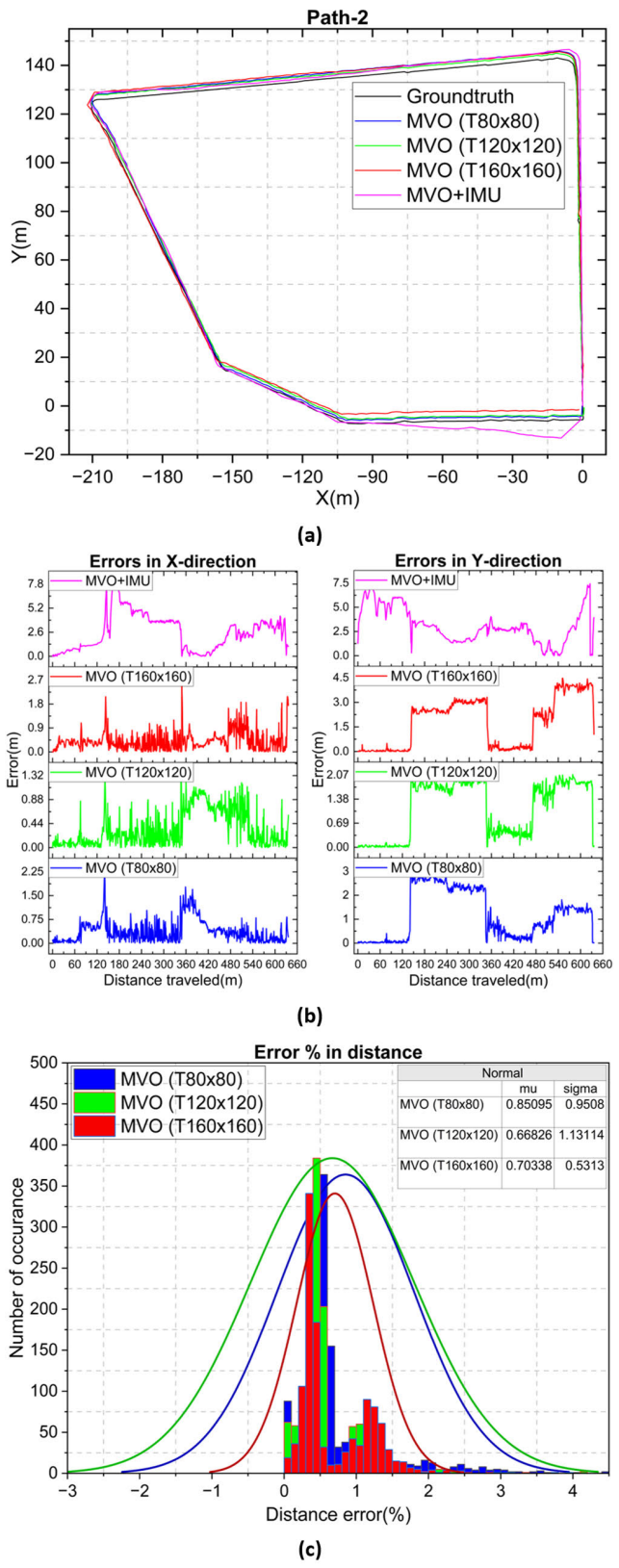

FIGURE 8. Estimated vehicle position and errors for outdoor Path-2.

lower size. This is mainly due to the inaccurate localization of the templates when the floor contains more generic features or indistinguishable textures. Hence, FNCC inaccurately fits the reference template, thus causing notable

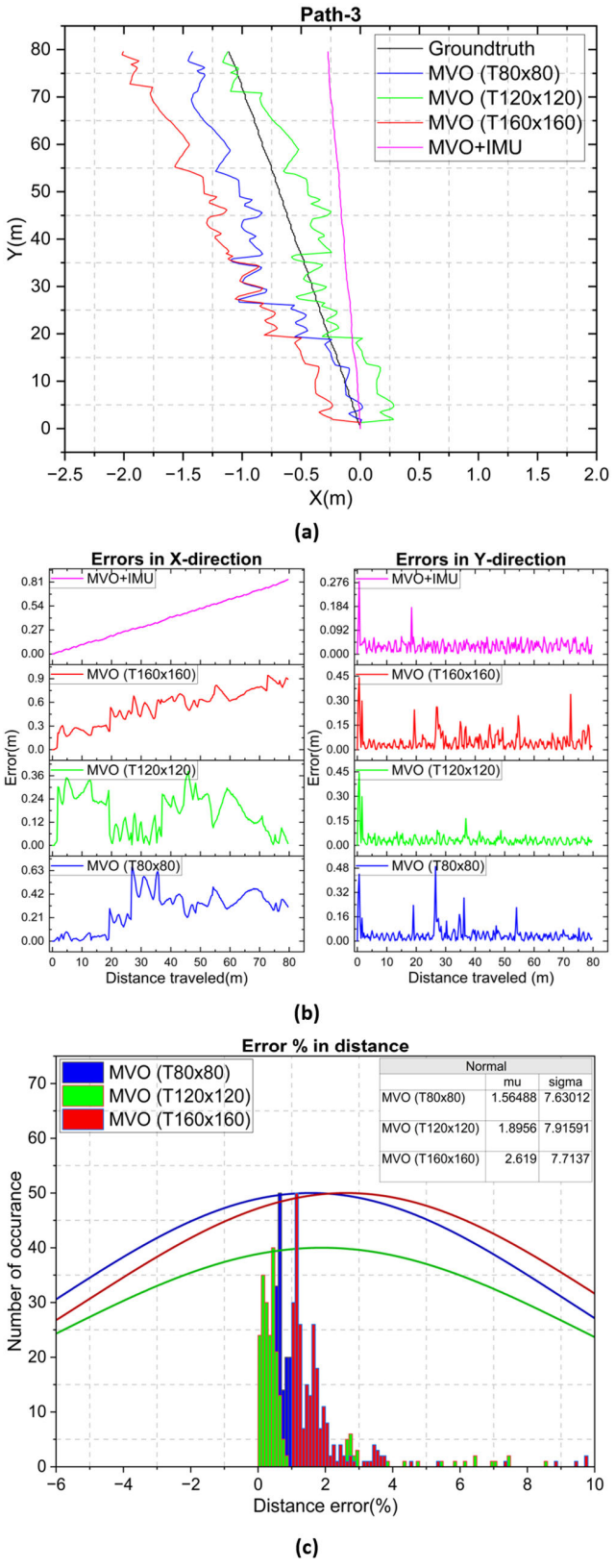

FIGURE 9. Estimated vehicle position and errors for indoor Path-3.

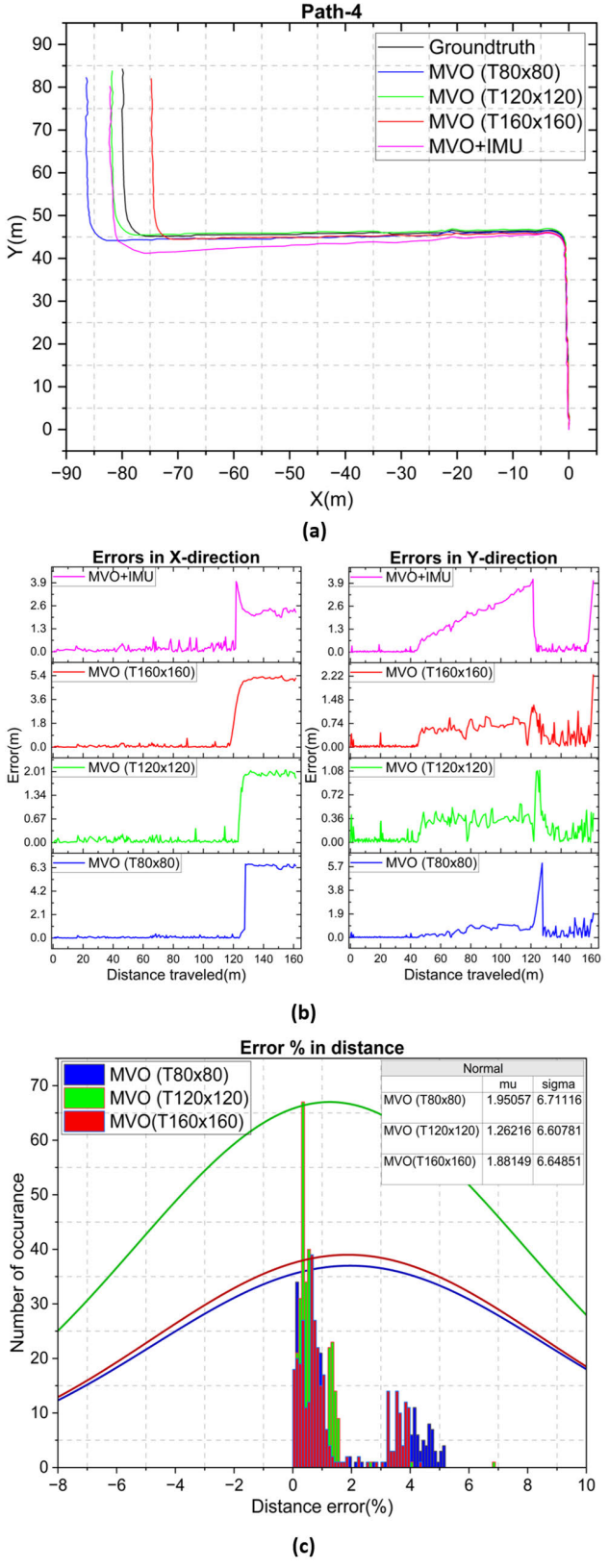

FIGURE 10. Estimated vehicle position and errors for indoor Path-4.

pixel errors that affect the GSC output. On the other hand, from Figure 9c, it can be seen that the distance error percentage distribution is minimal (1.56% with the 80×80 template).

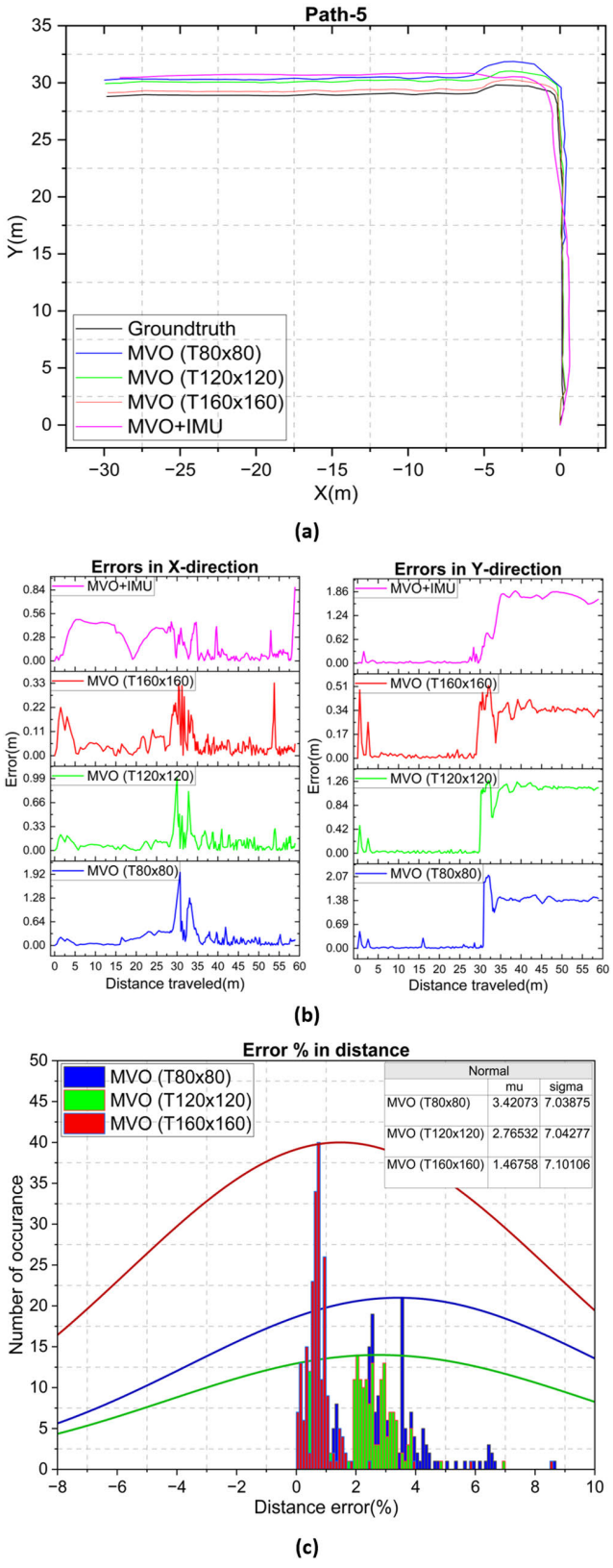

FIGURE 11. Estimated vehicle position and errors for indoor Path-5.

In the next experiment, the vehicle is made to follow Path-4 (shown in Figure 10a) and the vehicle pose is estimated. The use of a 120 × 120 template resulted in the lowest error with respect to the ground truth compared with the other templates, as well as with the MVO+IMU method. The highest error with the 80 × 80 template is due to the higher percentage of pixels being saturated (i.e., exhibiting glare) which results in notable displacement error. From Figure 10b, it is evident that after taking the second turn (i.e., nearly after 120 meters of distance traveled) the X error increases owing to the higher illumination at that particular section of the path. For the same path, when vehicle position is estimated with MVO+IMU, the estimated Y error starts increasing and this is due to IMU sensor drift. Overall, the 120 × 120 template shows a better estimation for Path-4 with the lowest average X and Y error (i.e., 0.4390 meters and 0.2083 meters respectively), and also the average distance error of 0.5805 meters. It can be noted that the percentage distance error for the 120 × 120 template is 1.26% which is the minimum among the other estimations.

In the last experiment, the vehicle was made to follow Path-5, which is made of different floor textures (i.e., with mixed floor conditions–cement floor for a few meters followed by tile floor and then BLACK dotted granite floor). The estimated path is plotted in Figure 11a. The experiment started with a fine cement floor with high reflectivity, as shown in Figure 6c. Therefore, it will be difficult for the smaller template (80 × 80) to perform the matching with high accuracy since there is a possibility of obtaining multiple similar FNCC scores. However, with larger template sizes, there is a greater possibility of capturing more surface features, which reduces false matching, thereby obtaining a higher FNCC score. With a high FNCC score, the displacement and orientation are estimated with higher accuracy and is evident from Figure 11b showing the error in the X and Y directions. Of the three template sizes, the 160 × 160 template shows the lesser average error in X and Y directions with the magnitude of 0.0588 meters and 0.2422 meters respectively. Also, the average distance error observed for this template is 0.2611 meters. When the vehicle moves from one type of floor to another type of floor, there is a sudden increase (i.e., spikes) in both the X and Y errors. This is due to the transition of the floor from cement to tile (at a distance of approximately 28 m) and the second impulse at a distance of approximately 33 m (from tile to granite floor). Even for different types of floor conditions, it can be noted that the error values are minimal with the 160 × 160 template. The percentage distance error is also low, with a value of 1.46%.

Since the developed GSC is a 2D calibration approach (i.e., it converts (u, v) pixel coordinates into (x, y) world coordinates), the elevation (height) estimation (i.e., z coordinate) is unviable. This is mainly due to the assumption of a flat ground plane. However, to realize the behavior of GSC-based MVO in the influence of uneven roads such as steep roads and speed breakers, we conducted two additional experiments. In the trailer ramp (steep road) experiment, the camera image frames are collected by making the test vehicle move on the 13° trailer ramp. From the results presented in Figure 12 (b) and (c), it can be seen that the developed MVO accurately

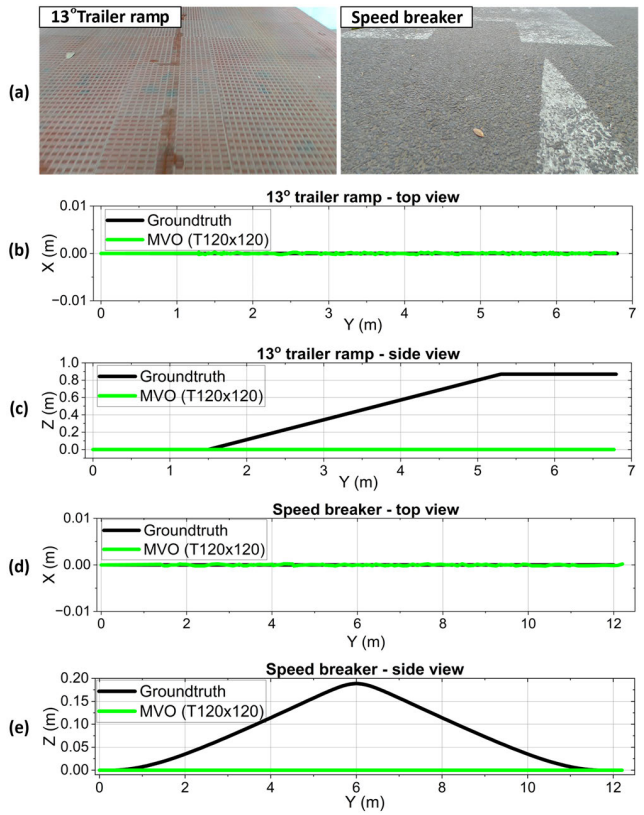

FIGURE 12. Estimated vehicle position in the trailer ramp and speed breaker. (a) 13° trailer ramp (first), and speed breaker (second) test image, (b) estimated pose on the trailer ramp-top view, (c) estimated pose on the trailer ramp-side view, (d) estimated pose on the speed breaker-top view, and (d) estimated pose on the speed breaker-side view. The BLACK line represents the ground truth and the GREEN line represents the estimated pose with a 120 × 120-sized template.

estimates the vehicle position with the average distance error in the xy-plane is 0.18 meters. However, due to the lack of elevation data, the average error in the z-plane is higher with a value of 0.57 meters. Similarly, in the speed breaker experiment, from Figure 12 (d) and (e), it is evident that the average distance error in the xy-plane is 0.23 meter, and also, in the z-plane, the average error is estimated as 0.14 meter. This increased z-plane error signifies the limitations of the developed GSC-based MVO presented in this work. Therefore, if the GSC is provided with the ability to estimate elevation, then the present 2D MVO can be extended to 3D MVO without the use of an additional IMU sensor.

A recently developed elevation estimation method called RoadBEV [32] estimates the elevation with an absolute error of 1.83 cm which can help extend the present GSC-based MVO. In RoadBEV, the authors presented a framework for estimating the elevation using monocular and stereovision cameras. Specifically, for monocular vision, the RoadBEV estimates the elevation by searching the candidate elevation voxels in the height direction (z-axis) rather than searching the depth. This makes RoadBEV more accurate than the existing elevation estimation methods. The same authors presented their dataset [33] for verifying elevation estimation methods. Additionally, it can be noted that, in the outdoor experiments (i.e., Path-1 and Path-2), owing to the rough surface of the road, the test vehicle is subjected to vibrations; therefore, the acquired images will be blurry. Moreover, on roads, tiny obstacles such as stones, leaves, broken twigs, and litter are present. However, FNCC-based template matching handles these blurry images and tiny obstacles efficiently, thus helping in accurate displacement and orientation estimation; however, owing to the lack of elevation data, 3D pose estimation becomes difficult and inaccurate.

TABLE 6. Maximum allowed velocity.

Template size (pixels)	Allowed matched template coordinates (pixels)	Maximum allowed velocity (m/s)	Maximum allowed displacement between image frames (m)
80x80	(640,597)	6.845	0.6854
120x120	(640,546)	6.411	0.6416
160x160	(640,501)	5.873	0.5989

C. MAXIMUM ALLOWED VELOCITY

The maximum allowed velocity of the Monocular Visual Odometry developed in this paper is directly proportional to the size of the template being used for the matching process. From Section III-A, it can be noted that the center point of the reference template is located at (640, 164) pixel coordinates. The center points of the matched template coordinates for each template size and the maximum allowed velocity are listed in Table 6.

From Table 6, it is evident that the developed MVO estimates the vehicle pose, even if the vehicle moves at a velocity greater than 5 m/s, irrespective of the template size. Furthermore, the algorithm estimates the odometry accurately for most of the cases by using a reference template size of 120×120 (after considering the results from all experiments). This template is selected as the optimum size, as the errors are minimal as shown in Table 7. Moreover, the proposed MVO performed better than the method presented in [26]. The visual odometry results from [26] have a distance error of 3.93 meters whereas with the proposed method the average distance error is 0.84 meters.

Considering the maximum allowed displacement presented in Table 6, the expected percentage distance error can be computed using equation (22) and is tabulated in Table 8.

$$Expected\ \%\ distance\ error = \frac{E}{Max.\ allowed\ displacement} \tag{22}$$

where $\boldsymbol{E}$ is the maximum allowed error (for a 120×120 pixel template it is 1.74 cm). It can be noted that the actual percentage distance error obtained with the 80×80 pixel template and 120×120 pixel template is lesser than the expected error (i.e., 2.54% and 2.72%) respectively. The average percentage

TABLE 7. Combined errors in the proposed MVO.

Template size	80x80	120x120	160x160	MVO+ IMU
Average error in X-direction (m)	**0.5309**	**0.3115**	0.5840	1.5077
Maximum error in X-direction (m)	**2.7025**	**1.2406**	2.5638	4.4757
Average error in Y-direction (m)	0.7745	**0.6691**	**0.7477**	1.6708
Maximum error in Y-direction (m)	2.8510	**1.4374**	**2.1904**	6.0952
Average distance error (m)	**1.1498**	**0.8454**	1.1613	2.6658
Maximum distance error (m)	**3.0580**	**1.6450**	3.1395	6.5803

TABLE 8. Comparison of percentage distance error (Actual vs Expected).

Percentage distance error (%)		Path 1	Path 2	Path 3	Path 4	Path 5	Average
80x80	Actual	0.50	0.85	1.56	1.95	3.42	1.65
	Expected			2.54			
120x120	Actual	0.44	0.66	1.89	1.26	2.76	1.41
	Expected			2.72			
160x160	Actual	0.64	0.70	2.61	1.88	1.46	1.46
	Expected			2.90			

distance error is lesser for the 120 × 120 pixel template at 1.41%. However, Path-5 exhibits a higher error, since the data is obtained on the cement floor where the 160 × 160 pixel template performs better. We compared our method with the LiDAR Odometry And Mapping (LOAM) algorithm [34], [35]. The LOAM algorithm is tested using the LiDAR point cloud obtained using the Velodyne vlp-16 LiDAR at 10Hz sampling frequency. The average percentage distance error obtained with LOAM in the xy-plane is 2.6% which is nearly 1.8 times higher than our MVO algorithm. Since LOAM is a 3D pose estimation algorithm the error in the z-plane is lesser than the MVO presented in this work.

D. COMPUTATIONAL PERFORMANCE

The real-time performance of the developed MVO algorithm is evaluated by measuring the time taken to process the image frames with the Central Processing Unit (CPU) and Graphical Processing Unit (GPU). As mentioned earlier (see Section IV-A), we use a 4 GB dedicated GPU-installed laptop to collect and process the acquired multi-sensor data. The processing time for template matching using the CPU and GPU is shown in Figure 13.

From Figure 13, it can be noted that the processing time required to execute the template-matching technique with the CPU is longer than with the GPU. Hence, there is a reduction of 42.43% execution time, when a GPU is used instead of a general-purpose CPU. In addition, by using the GPU for template matching, higher Frames of Image data can be processed per second (i.e., 9.75 FPS with GPU vs. 5.61 FPS with CPU). In the present work, using GPU-accelerated template matching, the proposed MVO-based vehicle distance measurement technique achieves real-time performance since the monocular camera sampling frequency is 10 Hz. In addition, this new template-matching method performs faster than the approach presented in [29], even with a CPU. Additionally, when compared with LOAM [34], the processing speed with GPU implementation is approximately 4 FPS, whereas the developed MVO on GPU is 2.5 times faster. In addition, we compared the computational complexity of our MVO with the LiDAR Inertial Odometry via Smoothing and Mapping (LIO-SAM) [36], [37]. The comparison showed that the LIO-SAM estimates the odometry at approximately 2 FPS (CPU) which is 2.8 times slower than the developed MVO implemented on a CPU.

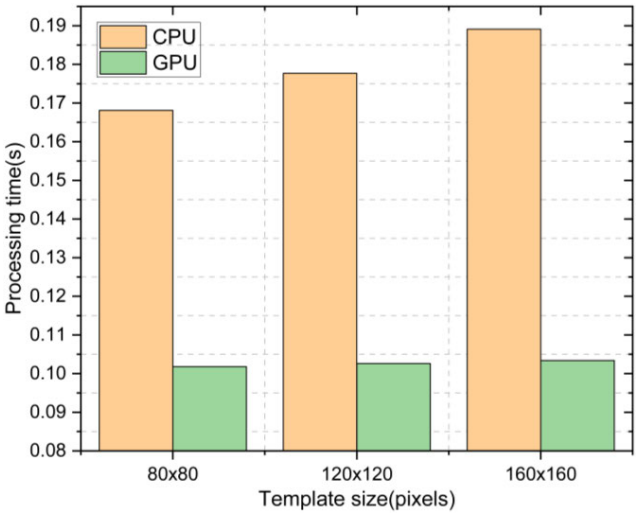

FIGURE 13. Relationship between processing time of template matching and the template size on CPU and GPU platforms.

E. FUTURE RESEARCH DIRECTIONS

Based on experimental findings and analysis, it is evident that enhancing the accuracy of the developed MVO is crucial. To achieve this, it is necessary to improve the displacement and orientation estimation processes. One notable limitation that should be addressed in future work is the potential failure of the template-matching process in scenarios involving sudden changes in lighting, contrast, partial occlusion, or image noise, leading to inaccuracies in the displacement and orientation estimation. A more robust approach involves extracting relevant features from both the reference template and the image to enhance the matching process rather than relying solely on direct pixel value matching. Introducing a 1D Kalman filter for displacement estimation can help mitigate sudden spikes and enhance estimation stability. Moreover, the orientation estimation accuracy can be enhanced by adopting a multiscale matching technique. Another critical aspect for future enhancement is to address the absence of elevation estimation, thereby limiting the applicability of the current system for 3D pose estimation. Integrating elevation estimation methods into GSC can provide 3D pose estimation.

By leveraging these improvements, the MVO can advance its capabilities and offer more precise and reliable pose estimation.

V. CONCLUSION

This paper proposes a Monocular Visual Odometry estimation algorithm that computes vehicle displacement and its orientation using a correlation-based template matching technique. The displacement is estimated using the Ground Spatial Calibration (GSC) technique, a lightweight direct method that converts camera pixel coordinates into world coordinates. Furthermore, the displacement estimation module uses Fast NCC algorithm-based template matching to compute displacement. In addition, instead of using an IMU to determine vehicle orientation, it is estimated from the acquired successive image frames. However, one of the challenges with image-based vehicle orientation estimation is that the orientation is limited to 135°. To solve this issue, an orientation fusion approach has been introduced to accurately transform the estimated orientation to a global coordinate frame. The accuracy of the proposed MVO is investigated for the different template sizes. Finally, the developed MVO algorithm is tested and validated through experiments conducted under both indoor and outdoor conditions. From the experiments, it is observed that the use of a 120 × 120 pixel template helps in reducing the displacement error estimation in both the X and Y directions. The proposed MVO method results in an average distance error of 1.41% (i.e., 0.8454 meters) for the driving distance of 1406.35 meters in the case of a 120 × 120 pixel template.

REFERENCES

[1] D. Nistor, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognition*, May 2004, pp. 1–23.

[2] M. Awsafur Rahman and S. Anowarul Fattah, "DwinFormer: Dual window transformers for end-to-end monocular depth estimation," *IEEE Sensors J.*, vol. 23, no. 18, pp. 21443–21451, Dec. 2023, doi: 10.1109/JSEN.2023.3299782.

[3] A. N. Tabata, A. Zimmer, L. dos Santos Coelho, and V. C. Mariani, "Analyzing CARLA 's performance for 2D object detection and monocular depth estimation based on deep learning approaches," *Expert Syst. Appl.*, vol. 227, Oct. 2023, Art. no. 120200, doi: 10.1016/j.eswa.2023.120200.

[4] Y. Liu and Z. Zhou, "Optical flow-based stereo visual odometry with dynamic object detection," *IEEE Trans. Computat. Social Syst.*, vol. 1, no. 1, pp. 1–13, Aug. 2022, doi: 10.1109/TCSS.2022.3205015.

[5] A. Howard, "Real-time stereo visual odometry for autonomous ground vehicles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2008, pp. 3946–3952.

[6] C. Duan, S. Junginger, K. Thurow, and H. Liu, "StereoVO: Learning stereo visual odometry approach based on optical flow and depth information," *Appl. Sci.*, vol. 13, no. 10, p. 5842, May 2023, doi: 10.3390/app13105842.

[7] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1052–1067, Jun. 2007, doi: 10.1109/TPAMI.2007.1049.

[8] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. 6th IEEE ACM Int. Symp. Mixed Augmented Reality*, Nov. 2007, pp. 225–234.

[9] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015, doi: 10.1109/TRO.2015.2463671.

[10] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," 2016, *arXiv:1607.02565*.

[11] F. Steinbrucker, J. Sturm, and D. Cremers, "Real-time visual odometry from dense RGB-D images," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Nov. 2011, pp. 719–722.

[12] J. Engel, J. Sturm, and D. Cremers, "Semi-dense visual odometry for a monocular camera," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1449–1456, doi: 10.1109/ICCV.2013.183.

[13] Q. Naixin, Y. Xiaogang, L. Chuanxiang, L. Xiaofeng, Z. Shengxiu, and C. Lijia, "Monocular semidirect visual odometry for large-scale outdoor localization," *IEEE Access*, vol. 7, pp. 57927–57942, 2019, doi: 10.1109/ACCESS.2019.2914033.

[14] S. Zaman, L. Comba, A. Biglia, D. R. Aimonino, P. Barge, and P. Gay, "Cost-effective visual odometry system for vehicle motion control in agricultural environments," *Comput. Electron. Agricult.*, vol. 162, pp. 82–94, Jul. 2019, doi: 10.1016/j.compag.2019.03.037.

[15] Q. Zeng, B. Ou, C. Lv, S. Scherer, and Y. Kan, "Monocular visual odometry using template matching and IMU," *IEEE Sensors J.*, vol. 21, no. 15, pp. 17207–17218, Aug. 2021, doi: 10.1109/JSEN.2021.3078847.

[16] M. Quan, S. Piao, M. Tan, and S.-S. Huang, "Accurate monocular visual-inertial SLAM using a map-assisted EKF approach," *IEEE Access*, vol. 7, pp. 34289–34300, 2019, doi: 10.1109/ACCESS.2019.2904512.

[17] L. Feng, X. Zhang, X. Peng, and M. Zhuang, "Improved monocular visual-inertial odometry with point and line features using adaptive line feature extraction," *Multimedia Tools Appl.*, vol. 83, no. 1, pp. 1481–1512, Jan. 2024, doi: 10.1007/s11042-023-15597-2.

[18] J. Zhang, J. Yang, and J. Ma, "Monocular visual-inertial odometry leveraging point-line features with structural constraints," *Vis. Comput.*, vol. 40, no. 2, pp. 647–661, Feb. 2024, doi: 10.1007/s00371-023-02807-z.

[19] M. He, C. Zhu, Q. Huang, B. Ren, and J. Liu, "A review of monocular visual odometry," *Vis. Comput.*, vol. 36, no. 5, pp. 1053–1065, May 2020, doi: 10.1007/s00371-019-01714-6.

[20] K. Sharma, "Improved visual SLAM: A novel approach to mapping and localization using visual landmarks in consecutive frames," *Multimedia Tools Appl.*, vol. 77, no. 7, pp. 7955–7976, Apr. 2018, doi: 10.1007/s11042-017-4694-x.

[21] X. Dong, L. Ai, and R. Jiang, "Motion estimation of indoor robot based on image sequences and improved particle filter," *Multimedia Tools Appl.*, vol. 78, no. 21, pp. 29747–29763, Nov. 2019, doi: 10.1007/s11042-018-6383-9.

[22] C. Zhao, L. Sun, Z. Yan, G. Neumann, T. Duckett, and R. Stolkin, "Learning Kalman network: A deep monocular visual odometry for on-road driving," *Robot. Auto. Syst.*, vol. 121, Nov. 2019, Art. no. 103234, doi: 10.1016/j.robot.2019.07.004.

[23] C.-H. Kim, J.-S. Kim, and D.-I. Dan Cho, "Harsh-environment visual odometry for field robots using data fusion of gyroscope & magnetometer," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 9566–9570, 2020, doi: 10.1016/j.ifacol.2020.12.2440.

[24] K. Song, J. Li, R. Qiu, and G. Yang, "Monocular visual-inertial odometry for agricultural environments," *IEEE Access*, vol. 10, pp. 103975–103986, 2022, doi: 10.1109/ACCESS.2022.3209186.

[25] S. Hwang, M. Cho, Y. Ban, and K. Lee, "Frame-to-frame visual odometry estimation network with error relaxation method," *IEEE Access*, vol. 10, pp. 109994–110002, 2022, doi: 10.1109/ACCESS.2022.3214823.

[26] N. Nourani-Vatani, J. Roberts, and M. V. Srinivasan, "Practical visual odometry for car-like vehicles," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2009, pp. 3551–3557.

[27] N. Nourani-Vatani and P. V. K. Borges, "Correlation-based visual odometry for ground vehicles," *J. Field Robot.*, vol. 28, no. 5, pp. 742–768, Sep. 2011, doi: 10.1002/rob.20407.

[28] R. Gonzalez, F. Rodriguez, J. L. Guzman, C. Pradalier, and R. Siegwart, "Combined visual odometry and visual compass for off-road mobile robots localization," *Robotica*, vol. 30, no. 6, pp. 865–878, Oct. 2012, doi: 10.1017/s026357471100110x.

[29] M. O. A. Aqel, M. H. Marhaban, M. I. Saripan, and N. B. Ismail, "Adaptive-search template matching technique based on vehicle acceleration for monocular visual odometry system," *IEEJ Trans. Electr. Electron. Eng.*, vol. 11, no. 6, pp. 739–752, Nov. 2016, doi: 10.1002/tee.22299.

[30] M. Birem, R. Kleihorst, and N. El-Ghouti, "Visual odometry based on the Fourier transform using a monocular ground-facing camera," *J. Real-Time Image Process.*, vol. 14, no. 3, pp. 637–646, Mar. 2018, doi: 10.1007/s11554-017-0706-3.

[31] R. E. Deakin, M. N. Hunter, and C. F. F. Karney, "The Gauss-Krueger projection: Karney-Krueger equations," in *Proc. 25th Int. Cartographic Conf.*, 2011, pp. 1–22.

[32] T. Zhao, L. Yang, Y. Xie, M. Ding, M. Tomizuka, and Y. Wei, "RoadBEV: Road surface reconstruction in bird's eye view," 2024, *arXiv:2404.06605*.
[33] T. Zhao, C. Xu, M. Ding, M. Tomizuka, W. Zhan, and Y. Wei, "RSRD: A road surface reconstruction dataset and benchmark for safe and comfortable autonomous driving," 2023, *arXiv:2310.02262*.
[34] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," *Robot., Sci. Syst.*, vol. 2, no. 9, pp. 1–9, 2014.
[35] L. Laboshin. *GitHub—Laboshinl/loam_velodyne: Laser Odometry and Mapping (Loam) is a Real-Time Method for State Estimation and Mapping Using a 3D LiDAR*. Accessed: May 10, 2024. [Online]. Available: https://github.com/laboshinl/loam_velodyne
[36] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled LiDAR inertial odometry via smoothing and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 5135–5142, doi: 10.1109/IROS45743.2020.9341176.
[37] T. Shan. *GitHub—TixiaoShan/LIO-SAM: LIO-SAM: Tightly-Coupled LiDAR Inertial Odometry via Smoothing and Mapping*. Accessed: May 12, 2024. [Online]. Available: https://github.com/TixiaoShan/LIO-SAM

R. RAJESH was born in Kumbakonam, Tamil Nadu, India, in 1996. He received the bachelor's degree in production engineering and the master's degree in mechatronics from Anna University, Chennai, India, in 2017 and 2019, respectively. He is currently pursuing the Ph.D. degree with the Department of Mechanical Engineering, Indian Institute of Technology, Madras, Chennai.

He has published conferences and journal articles. His research interests include computer vision and multisensor data fusion algorithm development for autonomous vehicle navigation. He was selected as the Prestigious Prime Minister's Research Fellow. He was a recipient of the Bronze Medal and Gold Medal for his academic excellence during the B.E. and M.E. degrees.

P. V. MANIVANNAN received the bachelor's degree in electronics and telecommunications engineering, the master's degree in applied electronics from the College of Engineering, Anna University, Chennai, India, and the Ph.D. degree in control systems for SI engines from Indian Institute of Technology Madras, Chennai, India.

From 2007 to 2016, he was an Assistant Professor with the Department of Mechanical Engineering, Indian Institute of Technology Madras, where he is currently an Associate Professor. He is the author of more than 50 research articles and three patents. He has also taught the summer term course "Mechatronic Systems" at the University of Nebraska, Lincoln, USA, and the University of Kaiserslautern, Germany, as a Visiting Faculty. His research interests include mechatronics, robotics, automotive control systems, embedded system designs, and sensor networks. He was a recipient of the Prestigious DAAD Fellowship and ERASMUS MUNDUS Teaching Fellowship.