Evaluation of a novel DSO-based Indoor Ceiling-Vision Odometry System

Abdelhak Bougouffa[1], Emmanuel Seignez[2], Samir Bouaziz[3], and Florian Gardes[4]

***Abstract*—Indoor localization for mobile industrial robots is a crucial step toward an autonomous system. A mobile robot needs a reliable and robust localization system to achieve its task autonomously. A reasonable estimate of the robot's state can be achieved through Visual Odometry (VO); however, with dynamic objects in the scene, classical VO approaches need to detect and filter these moving objects. Alternatively, we can use an up-facing camera to track the movement with respect to the ceiling, which represents a static and invariant space. This paper presents Ceiling-DSO: an indoor ceiling-vision (CV) system based on Direct Sparse Odometry (DSO). We take advantage of the generic formulation of DSO to avoid making assumptions about the observable shapes or landmarks on the ceiling, making the method generic and applicable to multiple ceiling types. We built a ceiling-vision dataset in a real-world scenario; we then used it to test our approach with different DSO parameters to identify the best fit for robot pose estimation. This paper provides a qualitative and quantitative analysis of the obtained results that showed an acceptable error rate compared to the ground truth.**

***Index Terms*—Indoor, ceiling-vision, visual odometry, state estimation, industrial environment, dynamic environment, DSO.**

I. Introduction

State estimation is still an open research problem. To accomplish an autonomous mobility task, a mobile robot must estimate its pose in space.

Estimating the robot's pose in an indoor environment requires using adequate sensors. Wheel encoders are commonly used for incremental pose estimation through *odometry*. However, due to mechanical coupling, wheel slippage, and the lack of external correction, the trajectory estimated from wheel encoders quickly drifts from the actual path [1], making it unreliable for long-term usage.

Alternatively, we can use cameras to calculate the mobile robot's position and orientation incrementally; this technique is known as *Visual Odometry (VO)* [2]. The term *"visual odometry"* was coined for the first time by Nister et al. in [3]; it consists of incrementally estimating the camera movement from a perceived image stream.

VO approaches can fall into two main categories: *indirect (or feature-based) methods* [4, 5, 6, 7] which requires preprocessing through feature detection techniques to extract a set of *points of interest (POI)*; these POIs are then used to minimize *geometric error*. And *direct methods* [8, 9] which do not require feature extraction; instead, it directly uses the raw pixel intensity values to minimize *photometric error*.

We expect several robots and humans to move around our mobile industrial robot. In such a highly dynamic environment, the VO estimation using a forward-facing camera can be challenging since we need to isolate and filter the moving objects from the scene [10, 11], which tends to be a complicated and resource-consuming task.

To solve the indoor dynamic environment issues, we can use an up-facing camera instead of a forward-facing one. The camera then observes the patterns in the ceiling, giving us a way to track the robot's movements with respect to the ceiling. WooYeon and Kyoung are known for being the first to propose such an approach [12]; they used the Harris corner detector to extract corners from a monocular up-facing camera. The corners are used as landmarks in an *Extended Kalman Filter (EKF)* based SLAM framework, with a multi-view description of landmarks to enhance data association.

In most ceiling vision-based odometry or SLAM approaches, the system makes strong assumptions about the shapes and patterns we can observe on the ceiling. In such approaches, known ceiling landmark classes are exploited (like *corners*, *lamps*, *speakers*, *fire alarms*, etc.); these landmarks are then used to track the pose and build the map using a filtering framework [13, 14].

Others make assumptions about the ceiling boundaries. In some use cases, the system can exploit the fact that the ceiling occupies most of the ceiling-view images while walls occupy the rest. The system then uses the ceiling images to build a feature map by detecting the boundaries between the ceiling and the walls. These features are then used to estimate the map and the pose in an EKF-SLAM framework [15, 16], or matched against a known blueprint using a *Monte Carlo Localization (MCL)* based framework [17].

In other approaches, visual markers are placed on the ceiling to facilitate the localization task. In [18], a set of easily detectable artificial markers are placed on the ceiling. The image stream from the up-facing camera is then processed to detect, identify and find the camera's position and orientation with

[1]Abdelhak Bougouffa is with Université Paris-Saclay, ENS Paris-Saclay, CNRS, SATIE, 91190 Gif-sur-Yvette, France. And with ez-Wheel, 16400 La Couronne, France. ORCID: 0000-0002-0323-1424. Corresponding author's mail: abdelhak.bougouffa@universite-paris-saclay.fr, a.bougouffa@ez-wheel.com.

[2]Emmanuel Seignez is with Université Paris-Saclay, ENS Paris-Saclay, CNRS, SATIE, 91190 Gif-sur-Yvette, France. emmanuel.seignez@universite-paris-saclay.fr.

[3]Samiz Bouaziz is with Université Paris-Saclay, ENS Paris-Saclay, CNRS, SATIE, 91190 Gif-sur-Yvette, France. samir.bouaziz@universite-paris-saclay.fr.

[4]Florian Gardes is with ez-Wheel, 16400 La Couronne, France. f.gardes@ez-wheel.com.

respect to the markers; the obtained information is then used in a graph-based optimization algorithm within a Bayesian estimation framework.

However, in an open warehouse or industrial space, most of the previously stated assumptions do not hold. The working space may be too wide to observe the ceiling and the walls in the same image, and the ceilings of such a space tend to be high and can contain inclined surfaces. To address these issues, we propose to apply a generic visual odometry approach to a ceiling-vision camera. We chose to use *Direct Sparse Odometry (DSO)*, which, being a direct approach, allows tracking movement even in regions with few distinct features. Our goal is to propose and evaluate a generic framework to estimate the movements with respect to the ceiling without making too many assumptions about the observable shapes or landmarks.

This paper presents and evaluates Ceiling-DSO, a *Direct Sparse Odometry (DSO)* based visual odometry with a ceiling vision camera. We conducted our experiments using a modular industrial mobile robot platform in a real-world environment. We tested varying DSO parameters while observing and analyzing their effect on the quality of the estimated trajectories. For comparison, we used a ground truth trajectory estimated with a LiDAR-based SLAM. We also provide qualitative and quantitative analysis of the obtained results.

The paper is organized as follows: Section II introduces the original DSO formulation and the assumptions we made in our Ceiling-DSO implantation; section III describes the experimental platform we used, the robot's sensors, the dataset we built and used to validate this work, and the experiment methodology; section IV presents and discusses the obtained results; and lastly, section V lists the conclusions and the perspectives of this work.

II. Ceiling Direct Sparse Odometry

The *Direct Sparse Odometry (DSO)* [8] is a monocular, direct visual odometry. In contrast to feature-based visual odometry methods, where we use only a set of distinct features to estimate camera motion, direct methods use information from all image pixels. We can divide direct methods into three categories: *dense* methods, which produce points cloud of high density using all pixels in the image, giving -in most cases- a heavy processing algorithm. Besides, *semi-dense* methods try to reduce the number of processed points to get a reasonable running time. Furthermore, *sparse* methods select only a small subset of points to process.

DSO falls in the third category; it uses a gradient-based strategy to uniformly select candidate points from high contrast regions. These candidate points are then used in a photometric error minimization process.

As a direct method, DSO needs all the pixels of a frame to be captured simultaneously; a camera that provides such acquisition is called a *global shutter camera*. For *rolling shutter cameras*, the image pixels are captured sequentially; therefore, DSO state estimation with such a camera will accumulate a systematic drift caused by this sequential acquisition. However, this effect can be considered by modeling or measuring it; for example, in [19], the authors proposed a DSO-based method that imposes the rolling shutter constraint to estimate the capture time.

We use the same formulation from the original DSO [8], we model the sensor as a pin-hole camera, with $\mathbf{K}$ its intrinsic matrix used for the geometric projection from 3D points into image plan Ω as: $\Pi_{\mathbf{K}} : \mathbb{R}^3 \rightarrow \Omega$ and back-projection from a 2D point (in image plan) and its depth information to 3D world $\Pi_{\mathbf{K}}^{-1} : \Omega \times \mathbb{R} \rightarrow \mathbb{R}^3$.

In DSO, a photometric calibration is considered from [20]. For a frame "i", the camera observes the raw intensity $I_i^{\text{RAW}} : \Omega \rightarrow [0, 255]$ of a pixel $\boldsymbol{x}$, which can be defined as a function of the irradiance B_i, the exposure time t_i, the non-linear response function $G : \mathbb{R} \rightarrow [0, 255]$, and the lens attenuation (vignetting) $V : \Omega \rightarrow [0, 1]$:

$$I_i^{\text{RAW}}(\boldsymbol{x}) = G(t_i V(\boldsymbol{x}) B(\boldsymbol{x})) \tag{1}$$

From (1), the photometrically corrected image I_i can be calculated by reverting the non-linear function, and removing the vignetting effect on the image, as in (2):

$$I_i(\boldsymbol{x}) \triangleq t_i B_i(\boldsymbol{x}) = \frac{G^{-1}(I_i^{\text{RAW}}(\boldsymbol{x}))}{V(\boldsymbol{x})} \tag{2}$$

The photometric error over all frames is defined as:

$$E_{\text{photo}} = \sum_{i \in N} \sum_{\mathbf{p} \in P_i} \sum_{j \in obs(\mathbf{p})} E_{i,\mathbf{p},j} \tag{3}$$

With N the total number of frames, P_i the set of points in the i-th frame, j iterates over $obs(\mathbf{p})$ which represents the set of all frames from which the point $\mathbf{p}$ is visible, and $E_{i,\mathbf{p},j}$ is the partial error term defined as a weighted sum of the Huber [21] norms calculated in a neighborhood pattern $\mathcal{N}_{\mathbf{p}}$ of the point $\mathbf{p}$:

$$E_{i,\mathbf{p},j} = \sum_{\mathbf{p} \in \mathcal{N}_{\mathbf{p}}} w_{\mathbf{p}} \left\| (I_j[\mathbf{p}'] - b_j) - \frac{t_j e^{a_j}}{t_i e^{a_i}} (I_i[\mathbf{p}] - b_i) \right\|_{\gamma} \tag{4}$$

$$\text{with Huber's norm} \quad \|\alpha\|_{\gamma} \triangleq \begin{cases} \frac{1}{2}\alpha^2 & \text{for} \quad |\alpha| < \gamma \\ \gamma \cdot (|\alpha| - \frac{1}{2}\gamma) & \text{otherwise} \end{cases}$$

$$\text{and} \quad w_{\mathbf{p}} \triangleq \frac{c^2}{c^2 + \|\nabla I_i(\mathbf{p})\|_2^2} \quad \text{with} \quad c \in \mathbb{R}$$

The Huber norm is a hybrid ℓ_1/ℓ_2-norm that is robust to outliers and differentiable everywhere; hence, it provides a suitable error measurement for gradient-based optimization.

The residual term in the sum (4) includes the difference between the intensity value of the point $\mathbf{p}$ in the current i-th frame and its intensity in all j-frames in which the point $\mathbf{p}$ is visible. The intensity is modeled as an affine brightness transfer function $e^{-a_i}(I_i - b_i)$ to be able to work with unknown exposure times. The $\mathbf{p}'$ in equation (4) represents the projection in the j-th frame, of the point $\mathbf{p}$ seen in

the i-th frame, with $d_{\mathbf{p}}$ its estimated depth. The projection matrix depends on the partial camera motion transform $\Delta\mathbf{T}_{j,i}$ between the camera poses $\mathbf{T}_i$ and $\mathbf{T}_j$.

$$\mathbf{p}' = \Pi_{\mathbf{K}}(\mathbf{R}\Pi_{\mathbf{K}}^{-1}(\mathbf{p}, d_{\mathbf{p}}) + \mathbf{t}) \tag{5}$$

$$\text{with} \quad \Delta\mathbf{T}_{j,i} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^{\mathsf{T}} & 1 \end{bmatrix} = \mathbf{T}_j\mathbf{T}_i^{-1}$$

The error term in (4) is then minimized through 6 steps of Gauss-Newton optimization. The optimization is done on the $\mathfrak{se}(3)$ Lie algebra, with the left-$\oplus$ operator defined as:

$$\oplus : \mathfrak{se}(3) \times \mathrm{SE}(3) \to \mathrm{SE}(3) \tag{6}$$

$$\text{with} \quad \begin{cases} \mathbf{x}_i \in \mathfrak{se}(3), \quad \mathbf{T}_i \in \mathrm{SE}(3) \\ \mathbf{x}_i \oplus \mathbf{T}_i \triangleq e^{\widehat{\mathbf{x}_i}} \cdot \mathbf{T}_i \end{cases}$$

More details about Lie groups in robotics and state estimation can be found in [22].

The optimized parameters $\zeta \in \mathrm{SE}(3)^n \times \mathbb{R}^m$ include the geometric parameters (poses, inverse depth values, and camera intrinsics), and the photometric parameters (the (a_i, b_i) affine brightness parameters). On the rigid motion manifold $\mathrm{SE}(3)$, lets ζ_0 denote the evaluation point of the manifold's tangent space, and $\mathbf{x} \in \mathfrak{se}(3)^n \times \mathbb{R}^m$ the accumulated delta updates. The current state estimate is then $\zeta = \mathbf{x} \oplus \zeta_0$ with the left-$\oplus$ operator extended beyond $\mathrm{SE}(3)$ elements as a regular addition.

The photometric error (3) is optimized in a Gauss-Newton system defined as:

$$\mathbf{H} = \mathbf{J}^{\mathsf{T}}\mathbf{W}\mathbf{J} \tag{7}$$

$$\mathbf{b} = -\mathbf{J}^{\mathsf{T}}\mathbf{W}\mathbf{r} \tag{8}$$

With $\mathbf{r} \in \mathbb{R}^n$ is a vector grouping all residuals, $\mathbf{W} \in \mathbb{R}^{n \times n}$ is the diagonal weights' matrix, and $\mathbf{J} \in \mathbb{R}^{n \times d}$ is the Jacobian of the residuals vector $\mathbf{r}$.

The let r_k a single residual from $\mathbf{r}$, and $\mathbf{J}_k$ its associated Jacobian row evaluated at $\mathbf{x}$ with a small additive increment $(\delta + \mathbf{x})$, the residual depends on the current state variables $(\mathbf{T}_i, \mathbf{T}_j, d_{\mathbf{p}}, a_i, a_j, b_i, b_j) = \mathbf{x} \oplus \zeta_0$, where: $\mathbf{T}_i$ and $\mathbf{T}_j$ the poses of the camera at frames "i" and "j", respectively, $d_{\mathbf{p}}$ the inverse depth, $\mathbf{K}$ the intrinsics matrix, and a_i, a_j, b_i, b_j the affine brightness parameters for the frames "i" and "j".

$$r_k = (I_j[\mathbf{p}'(\mathbf{T}_i, \mathbf{T}_j, d, \mathbf{K})] - b_j) - \frac{t_j e^{a_j}}{t_i e^{a_i}}(I_i - b_i) \tag{9}$$

$$\mathbf{J}_k = \frac{\partial r_k((\delta + \mathbf{x}) \oplus \zeta_0)}{\partial \delta} \tag{10}$$

The Gauss-Newton optimization is performed over a sliding window of N_f keyframes, points beyond this window get marginalized.

In our Ceiling-DSO implementation, we used a simplified version of the DSO formulation. We supposed a linear response function $\forall \mathbf{x} \in \Omega : G(\mathbf{x}) = \mathbf{x}$, and we used lenses with no vignetting, so $\forall \mathbf{x} \in \Omega : V(\mathbf{x}) = 1$.

III. Experiment

A. Experimental platform

In our experiments, we used a prototype of a mobile industrial robot (figure 1). The platform is a modular, differential-drive robot powered by two *ez-Wheel Safety Wheel Drive (SWD®)* self-contained motorized wheels. This prototype is an evolution of our previously validated platform named Smart-Trolley [23]. It targets moving high loads indoors, mainly for usage in industrial environments. We designed and dimensioned the robot to move a maximum of 2 tonnes (2000kg) of loads. For safety purposes, the robot operates only at low speeds ($5\mathrm{km\,h^{-1}} \approx 1.4\mathrm{m\,s^{-1}}$).

Fig. 1. The SWD Starter Kit.

The robot is equipped with two incremental encoders, two cameras, an *Intel® RealSense™ D435i* facing forward, and an *Intel® RealSense™ 455* facing upward. The robot also integrates a safety laser range finder of type *IDEC S2L*, with a maximum range of $30m$.

The platform includes a *Neousys Nuvo-7002LP* embedded industrial computer, based on an 8th Generation *Intel® Coffee lake Core™ i5* processor and a 16GB DDR4 2666/2400 SDRAM. It runs the *Ubuntu 20.04* operating system with the robotics middleware *ROS Noetic*. We used the embedded computer to implement the differential-drive kinematic model to control[1] the robot and to collect raw sensor data. We connected the embedded computer to a wireless transceiver, which we use to drive the robot remotely using a wireless joystick.

The platform uses the low-level obstacle detection provided by the safety LiDAR. We configured two detection zones, a large one mapped as a *Safety-Limited Speed (SLS)* zone triggered when the robot approaches an obstacle, and a smaller one mapped as a *Safe Direction Indication (SDI)* zone, used to prohibit moving toward a close obstacle. We implemented these detection and response behaviors at the lowest level. The SDI and SLS signals are sent from the LiDAR to the motorized wheels directly via safe *Output Signal Switching Device (OSSD)* outputs.

[1]The controllers are available under the LGPL-2.1 license at `github.com/ezWheelSAS/swd_ros_controllers` for ROS, and at `github.com/ezWheelSAS/swd_ros2_controllers` for ROS2.

B. Dataset

We performed our experiment in an indoor open space of $21 \times 15m$. We collected raw sensor data from the odometry, the images of the stereo up-facing camera, the forward-facing camera, and the LiDAR ranges.

The ceiling in our test environment is inclined, with heights between 4 and 6 meters. The shapes and landmarks on the ceiling can differ from region to region; figure 2 includes example images of the ceiling seen by the up-facing camera.

Fig. 2. Example images of the ceiling of the test environment as seen by the up-facing camera.

C. Methodology

We evaluated the Ceiling-DSO on a set of sequences from our collected dataset. We studied the influence of varying *input images size*, *frame rate*, and *maximum size of the optimization window* on the quality of the estimated trajectory and the execution time. The goal of such an evaluation is to identify good parameter combinations for real-time usage.

We used the data from the LiDAR to calculate a ground truth trajectory using the open-source LaMa SLAM [24]. The map of the test environment calculated using this SLAM algorithm is represented in figure 3.

We conducted our experiments on a planar surface with a camera, and a 2D LiDAR fixed on the robot. We denote the 2D ground truth trajectory $\mathcal{G}$, DSO uses a monocular camera; hence, its estimated trajectory $\mathcal{P}$ is only valid up to a scale $\lambda \in \mathbb{R}^+$. The DSO trajectory is 3-dimensional, while the ground truth is only 2-dimensional. Since the robot movement surface is planar, we can compare the two trajectories after alignment.

To align the estimated trajectories and prepare them for comparison, we use an approach similar to [25]. We first synchronize the two trajectories using ROS' global clock. Let's denote the n synchronized ground truth and visual odometry positions $\mathcal{G}'$ and $\mathcal{P}'$, respectively, defined as:

$$\begin{aligned} \mathcal{G}' &= \{\mathbf{g} \mid \quad \|t_{\mathbf{p}} - t_{\mathbf{g}}\| < \tau\} \\ \mathcal{P}' &= \{\mathbf{p} \mid \quad \|t_{\mathbf{p}} - t_{\mathbf{g}}\| < \tau\} \end{aligned} \quad \text{with} \quad \begin{cases} \forall \mathbf{g} \in \mathcal{G} \\ \forall \mathbf{p} \in \mathcal{P} \\ \tau \in \mathbb{R}^+ \end{cases} \tag{11}$$

With $t_{\mathbf{p}}$ and $t_{\mathbf{g}}$ are the timestamps of points $\mathbf{p}$ and $\mathbf{g}$, respectively, and τ is the synchronization threshold.

We estimate the global similarity transformation $\mathbf{S} \in \mathrm{Sim}(3)$ between the synchronized trajectories of the ground truth and the visual odometry ($\mathcal{G}'$ and $\mathcal{P}'$ respectively). We express the similarity transformation as:

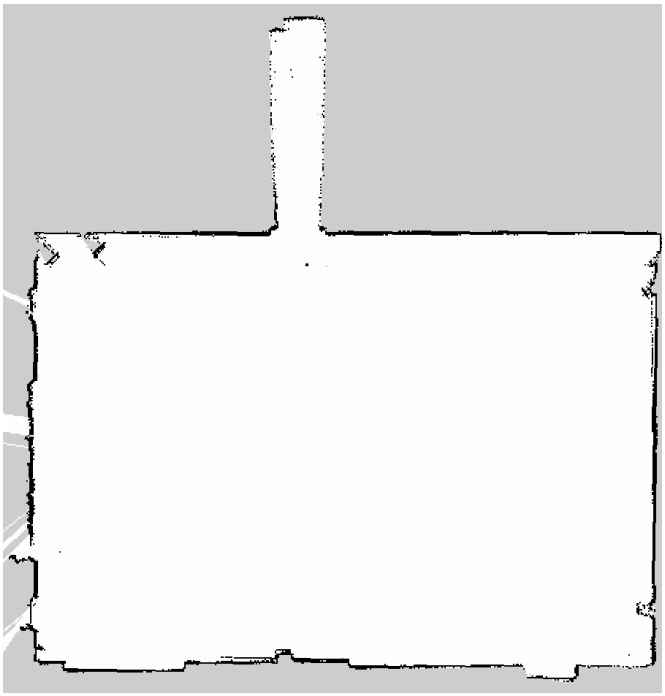

Fig. 3. The test environment's map.

$$\mathbf{S} \triangleq \begin{bmatrix} \lambda\mathbf{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix} \tag{12}$$

For convenience, we write $\mathbf{S} = (\mathbf{R}, \mathbf{t}, \lambda)$, where $\mathbf{R} \in \mathrm{SO}(3)$ is the 3D rotation, $\mathbf{t} \in \mathbb{R}^3$ is the translation and $\lambda \in \mathbb{R}^+$ is the scale.

We can express the alignment problem as a least-squares problem, applied to the minimization of the error between the synchronous pose pairs $\mathbf{p}_i \in \mathcal{P}'$ and $\mathbf{g}_i \in \mathcal{G}'$. The optimal global alignment transformation $\overset{\star}{\mathbf{S}} = (\overset{\star}{\mathbf{R}}, \overset{\star}{\mathbf{t}}, \overset{\star}{\lambda})$ can be expressed as:

$$\overset{\star}{\mathbf{S}} = \underset{\mathbf{R},\mathbf{t},\lambda}{\arg\min} \sum_{i=0}^{n} \|\mathbf{g}_i - (\lambda\mathbf{R}\mathbf{p}_i + \mathbf{t})\|^2 \tag{13}$$

The VO estimated trajectory $\mathcal{P}$ is then aligned using $\overset{\star}{\mathbf{S}}$, giving $\overset{\star}{\mathcal{P}}$ such as:

$$\forall \mathbf{p} \in \mathcal{P} : \ \overset{\star}{\mathcal{P}} = \{\overset{\star}{\lambda}\overset{\star}{\mathbf{R}}\mathbf{p} + \overset{\star}{\mathbf{t}}\} \tag{14}$$

We use metrics similar to [25], we evaluate the relative errors (REs) between the ground truth and the ceiling DSO trajectory. We then consider the euclidean norm of the position error to evaluate the obtained results.

IV. Results and Discussion

We tested Ceiling-DSO multiple times on multiple sequences from our collected dataset. In this paper we present results obtained on a square-like sequence.

First, we provide qualitative results where we align and trace the aligned trajectories compared to the ground truth. We

tested a total of 24 trajectories per sequence, iterating over all combinations of tested parameters: image size of 848×480 or 424×240, a frame rate of 3, 6, 15, or 30fps, and a maximum optimization window size of 5, 7, or 15.

Considering the trajectories estimated from the test sequence, figure 4 show the effect of changing frame rate and image size while fixing the maximum optimization window size at 7. We noticed that the trajectory error increases when using frame rates lower than 15fps. We need to emphasize that the choice of frame rate is also linked to the robot speed. For robots moving at high speeds, streams of higher frame rates will be needed. As DSO runs a coarse-to-fine matching, increasing the image size can help refine the estimation; however, our tests showed no significant effect when reducing the image size from 848×480 to 424×240.

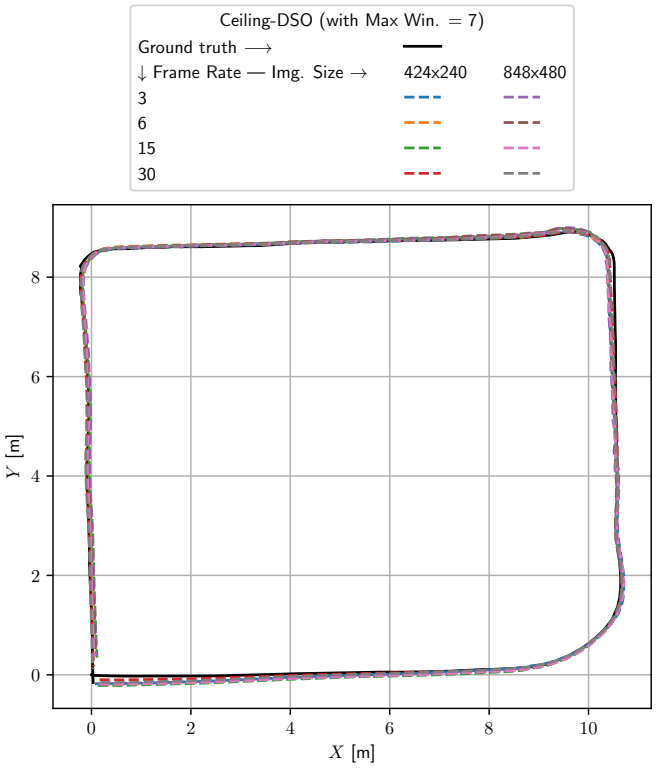

Fig. 4. Trajectories over multiple image sizes and input frame rates at a fixed optimization window size of 7.

Figure 5 show the effect of changing the image size and the maximum optimization window size on the trajectories while fixing the frame rate to 30fps. These trajectories show slight improvements when increasing the maximum window size. As the optimization step performs a local bundle adjustment, using a larger window size should help increase the accuracy in more complex trajectories; however, our tests showed an acceptable accuracy when using a maximum window size of 7.

For the quantitative analysis, we plot the euclidean norm of the relative position error $\epsilon_i = \sqrt{\Delta x_i^2 + \Delta y_i^2 + \Delta z_i^2}$

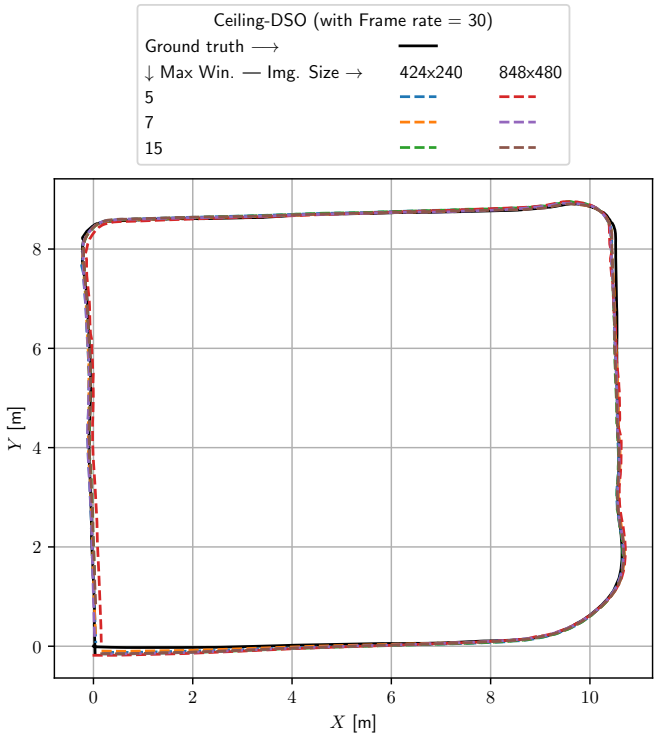

Fig. 5. Trajectories over multiple image sizes and maximum optimization window sizes at a fixed frame rate of 30fps.

as a function of traveled distance. Figure 6 summarize the relative error for the test sequence, varying one parameter at a time while fixing the two others to defaults (default value 848×420 for image size, 30 for frame rate, and 7 for the maximum window size). We can see that the most drastic errors occurred when decreasing the frame rate below 15fps. For other parameters, we have noticed no significant influence.

To better see the relative error distribution across the whole trajectory and for each combination of parameters, we provide figure 7, which shows box plots of relative error with respect to tested parameters, highlighting for each combination the estimated distribution characterized by the box which delimits the first and third quartiles (Q_1 and Q_3), the second quartile Q_2 *a.k.a.* the median, and the whiskers which extend from the box marking the variability outside the lower and upper quartiles within $1.5 \times \text{IQR} = 1.5 \times (Q_3 - Q_1)$, the IQR acronym stands for *interquartile range*.

The red dashed line in figure 7 bisects the sorted medians into two parts. We select the combinations of parameters of the first half, which have the lesser relative errors. Most of the second half, which contains combinations of high relative errors, correlates with the use of a frame rate less than 15fps. We can then define a frame rate regulation strategy based on the robot's movement; when moving on a straight line and at low speeds, we can lower the frame rate, while the frame rate should be increased when the robot is steering or accelerating.

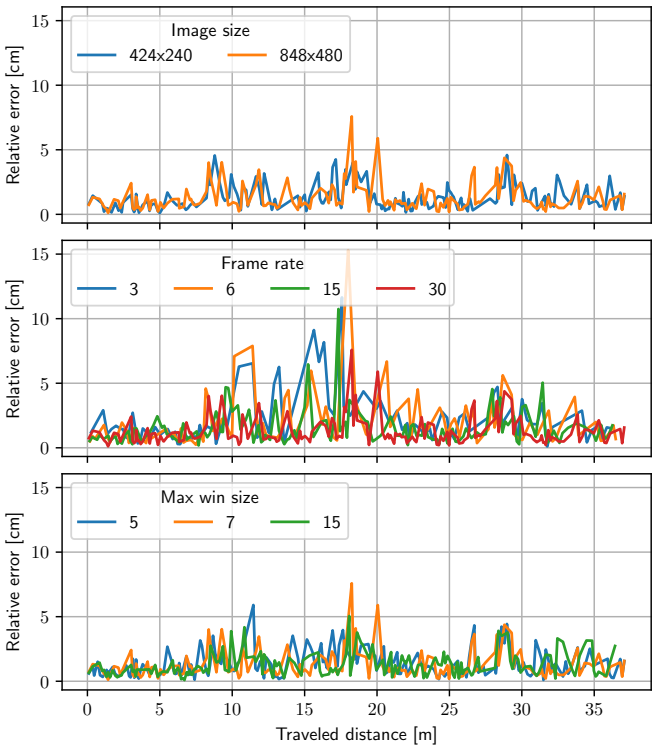

Fig. 6. The relative error between the ground truth and DSO trajectory with multiple frame rate, image size, and maximum optimization window size settings (Seq1).

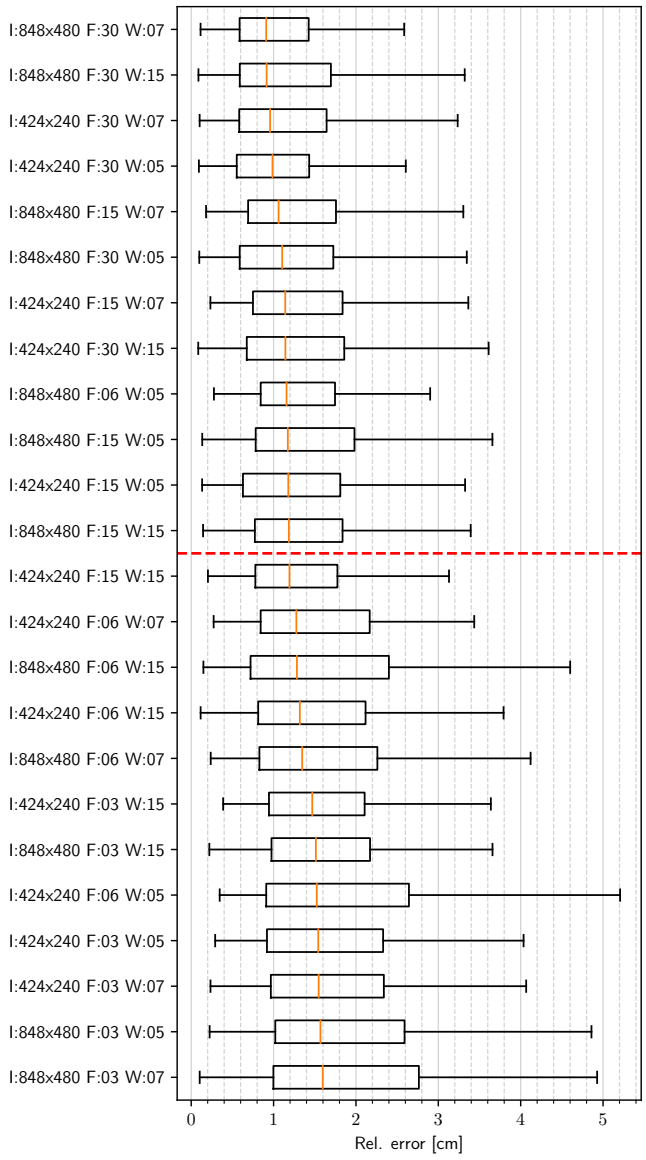

Fig. 7. Box plot of *relative errors* for each combination of tested parameters (I: *image size*, F: *frame rate*, W: *optimization window size*). The box delimits the first and third quartiles. The orange line marks the median. The lines (whiskers) extend from the box showing the variability within $1.5 \times$ IQR. Boxes are sorted by the medians of relative errors.

V. Conclusion

We presented in this paper Ceiling-DSO, an indoor ceiling-vision odometry system based on DSO, we designed our system for usage with indoor mobile industrial robots. The main advantage of Ceiling-DSO is its adequacy for indoor industrial dynamic environments; by observing the ceiling, the system is no more disturbed with the objects moving around the robot. By exploiting the generic DSO formulation, our system avoids making assumptions about the shape or the content of the ceiling, making it a generic solution which can be used in different ceiling types. We built a real-world ceiling-vision dataset with LiDAR sensor data for usage as ground truth. We used this dataset to validated our approach by comparing the Ceiling-DSO estimated trajectories to the ground truth. We also studied the effect of changing the *input image size*, the *input frame rate*, and the *optimization window size* to identify a suitable combination of parameters.

The results showed no significant influence in changing the input image size; furthermore, the estimated trajectory and the run time got slightly affected when changing the input frame rate. In our tests, a frame rate of 15fps gave a good tradeoff between accuracy and run time. Varying the optimization window size showed no significant influence on the accuracy of the estimated trajectory while heavily affecting the run time. Our tests showed that using 7 as the maximum window size gives reasonable results in both timing and accuracy.

This work defines a baseline for our future research; we plan to develop a sensor fusion method for online metric scale estimation to get the metric scale alongside with the DSO estimated pose. Furthermore, a map management and loop-closing strategies are planned to be added to our system to provide a complete SLAM solution. We plan also to publish the multi-sensor dataset we built, including the stereo image streams from the forward-facing camera and the up-facing camera, the LiDAR data, and the data from the inertial measurement units (IMUs) of the cameras. This will allow other scientists to use a dataset of a real-world scenario to test

ceiling-vision algorithms and compare them with a wide range of ordinary visual odometry/SLAM algorithms of monocular, stereo and/or inertial approaches.

References

[1] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*, 2nd ed. The MIT Press, 2011.

[2] D. Scaramuzza and F. Fraundorfer, "Visual Odometry [Tutorial]," *IEEE Robotics Automation Magazine*, vol. 18, no. 4, pp. 80–92, Dec. 2011.

[3] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 1, Jun. 2004, pp. I–I.

[4] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[5] D. N. S. D. Awang Salleh and E. Seignez, "Swift Path Planning: Vehicle Localization by Visual Odometry Trajectory Tracking and Mapping," *Unmanned Systems*, vol. 06, no. 04, pp. 221–230, Aug. 2018.

[6] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-Time Single Camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, Jun. 2007.

[7] D. N. S. D. Awang Salleh and E. Seignez, "Longitudinal error improvement by visual odometry trajectory trail and road segment matching," *IET Intelligent Transport Systems*, vol. 13, no. 2, pp. 313–322, 2019.

[8] J. Engel, V. Koltun, and D. Cremers, "Direct Sparse Odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 611–625, Mar. 2018.

[9] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 15–22.

[10] C. Sheng, S. Pan, W. Gao, Y. Tan, and T. Zhao, "Dynamic-DSO: Direct Sparse Odometry Using Objects Semantic Information for Dynamic Environments," *Applied Sciences*, vol. 10, no. 4, p. 1467, Jan. 2020.

[11] D.-H. Kim and J.-H. Kim, "Effective Background Model-Based RGB-D Dense Visual Odometry in a Dynamic Environment," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1565–1573, Dec. 2016.

[12] J. WooYeon and M. L. Kyoung, "CV-SLAM: A new ceiling vision-based SLAM technique," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Aug. 2005, pp. 3195–3200.

[13] D. Y. Kim, H. Choi, H. Lee, and E. Kim, "A new cvSLAM exploiting a partially known landmark association," *Advanced Robotics*, vol. 27, no. 14, pp. 1073–1086, Oct. 2013.

[14] S. Hwang and J. Song, "Monocular Vision-Based SLAM in Indoor Environment Using Corner, Lamp, and Door Features From Upward-Looking Camera," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 10, pp. 4804–4812, Oct. 2011.

[15] H. Choi, D. Y. Kim, J. P. Hwang, C.-W. Park, and E. Kim, "Efficient Simultaneous Localization and Mapping Based on Ceiling-View: Ceiling Boundary Feature Map Approach," *Advanced Robotics*, vol. 26, no. 5-6, pp. 653–671, Jan. 2012.

[16] H. Choi, R. Kim, and E. Kim, "An Efficient Ceiling-view SLAM Using Relational Constraints Between Landmarks:," *International Journal of Advanced Robotic Systems*, Jan. 2014.

[17] A. Ribacki, V. A. M. Jorge, M. Mantelli, R. Maffei, and E. Prestes, "Vision-Based Global Localization Using Ceiling Space Density," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 3502–3507.

[18] Y. Li, S. Zhu, Y. Yu, and Z. Wang, "An improved graph-based visual localization system for indoor mobile robot using newly designed markers," *International Journal of Advanced Robotic Systems*, vol. 15, no. 2, p. 1729881418769191, Mar. 2018.

[19] D. Schubert, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "Direct Sparse Odometry with Rolling Shutter," in *Computer Vision – ECCV 2018*, ser. Lecture Notes in Computer Science, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 699–714.

[20] J. Engel, V. Usenko, and D. Cremers, "A Photometrically Calibrated Benchmark For Monocular Visual Odometry," *arXiv:1607.02555 [cs]*, Oct. 2016.

[21] P. J. Huber, "Robust Estimation of a Location Parameter," *The Annals of Mathematical Statistics*, vol. 35, no. 1, pp. 73–101, Mar. 1964.

[22] J. Solà, J. Deray, and D. Atchuthan, "A micro Lie theory for state estimation in robotics," *arXiv:1812.01537 [cs]*, Nov. 2020.

[23] A. Bougouffa, E. Seignez, S. Bouaziz, and F. Gardes, "SmartTrolley: An Experimental Mobile Platform for Indoor Localization in Warehouses," in *2020 3rd International Conference on Robotics, Control and Automation Engineering (RCAE)*, Nov. 2020, pp. 108–115.

[24] E. Pedrosa, A. Pereira, and N. Lau, "Fast Grid SLAM Based on Particle Filter with Scan Matching and Multithreading," in *2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. Ponta Delgada, Portugal: IEEE, Apr. 2020, pp. 194–199.

[25] Z. Zhang and D. Scaramuzza, "A Tutorial on Quantitative Trajectory Evaluation for Visual(-Inertial) Odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2018, pp. 7244–7251.