Human Localization in Robotized Warehouses based on Stereo Odometry and Ground-Marker Fusion

Goran Popović[a], Igor Cvišić[a], Gaël Ecorchard[b], Ivan Marković[a], Libor Přeučil[b], Ivan Petrović[a]

[a] *University of Zagreb Faculty of Electrical Engineering and Computing,*
Department of Control and Computer Engineering,
Laboratory for Autonomous Systems and Mobile Robotics,
Unska 3, HR-10000, Zagreb, Croatia,
goran.popovic@fer.hr, igor.cvisic@fer.hr, ivan.markovic@fer.hr, ivan.petrovic@fer.hr

[b] *Czech Technical University in Prague,*
Czech Institute of Informatics, Robotics and Cybernetics,
160 00 Prague, Czech Republic
gael.ecorchard@cvut.cz, preucil@cvut.cz

Abstract

Modern logistic solutions for large warehouses consist of a fleet of robots that transfer goods, move racks, and perform other physically difficult and repetitive tasks. The shopfloor is usually enclosed with a safety fence and if a human needs to enter the warehouse all the robots are stopped, as opposed to only the ones in the most immediate vicinity of the human, thus significantly limiting the warehouse efficiency. To tackle this challenge, an integrated safety system is needed with human localization as one of its essential components. In this paper, we propose a novel human localization method for robotized warehouses that is based on a suite of wearable visual sensors installed on a vest worn by humans. The proposed method does not require any modifications of the warehouse environment and relies on the already existing infrastructure. Specifically, we estimate the human location by fusing stereo visual-intertial odometry data and distances to the known absolute poses of the detected ground-markers which robots use for their localization. Fusion is performed by building a pose graph, where we treat estimated human poses relative to markers as graph nodes and odometry estimates as graph edges. We conducted extensive laboratory and warehouse facility experiments, where we tested the reliability and accuracy of the proposed method and compared its performance to a state-of-the-art visual SLAM solution, namely ORB-SLAM2. The results indicate that our method can track absolute position in real-time and has competitive accuracy with respect to ORB-SLAM2, while ensuring higher localization reliability when faced with structural changes in the environment. Furthermore, we provide publicly the experimental datasets to the research community.

Keywords: visual odometry, human localization, warehouse automation, sensor fusion

1. Introduction

The scale of products available today is largely the result of intense development of production and logistic services. The pressure on the demand-supply chain has resulted in large warehouses that are fully automated and the product flow is supported by a fleet of autonomous robots. Such automated warehouse systems, where robots bring products to humans, are characterized by significantly improved productivity and flexibility [1], but they also require knowledge about the locations of both products and robots, as well as an optimized fleet management system in charge of the robot control. To avoid hazardous situations, the robot-working area, i.e. the shopfloor, is usually enclosed with a safety fence to prevent humans from entering it. In case such a scenario happens, all the robots are stopped and remain still until the human exits. Although it is not intended for the humans to enter the shopfloor frequently, in the case of unexpected events, such as robot hardware failure or the drop of a product, the human needs to intervene and fix the problem. In the case of large warehouses, this means that many robots are standing idle and causing high opportunity costs. Sectional lockouts can be applied in the scenarios where the worker's tasks are limited to maintenance of the robots which can autonomously or manually come to the maintenance section. However, when workers must perform a picking task or resolve an issue in the warehouse, sectional lockouts cannot be applied as the worker needs to be able to freely access the whole warehouse.

In [2] authors give an extensive overview of safety mechanisms for human-robot collaboration during a manufacturing process, and works [3] and [4] present two frameworks where humans and robots closely interact in an industrial environment. However, direct interaction in the warehouse is not necessary and a different approach should be applied. With human safety and warehouse efficiency in mind, it would be ideal if only the robots which are in the immediate vicinity of the human are stopped while others continue to operate. Having such a system would ensure human safety; however, to make the warehouse work at peak efficiency, we would also need to know the location of the human workers in the warehouse. In this paper, we assume that a relative ranging safety system is available in

the warehouse and, indeed, it is one of the technologies developed withing the scope of the Horizon 2020 project SafeLog[1]. Given that, we then concentrate on the human localization aspect that enables the fleet management system to account for human presence and thus replan the paths of the robots, not only to ensure efficiency, but also the comfort of human workers (note that each robot carrying a rack filled with goods weighs close to 900 kg).

Until recently, localization solutions for warehouse environments have been oriented towards automated robots and products, but to our knowledge, no solution for human localization in the automated warehouse environment has been developed. In [5] authors divide real-time localization technologies into the following categories: ultra-wideband (UWB), radio frequency identification (RFID) systems, vision systems, and Wi-Fi technology. Though UWB and RFID have been successfully used for forklift localization, we find the approach with visual sensors more appropriate for our case since visual features that already exist for the robot localization could also be used for human localization. Visual systems can be implemented by placing a set of cameras throughout the environment that can track people within their fields of view [6, 7]. In our case, such an approach has two major drawbacks: large warehouses would require numerous sensors and installation and calibration of such a system would be very time-consuming. With this in mind, we are focusing on a visual localization framework based on a suite of wearable sensors that are placed on the so-called Safety Vest, depicted in Fig. 1, worn by human workers that enter the warehouse shopfloor (note that the vest also ensures worker safety thanks to the previously mentioned safe relative ranging).

Since our approach relies on wearable sensors, the choice is limited to small, lightweight, and low-power consuming ones. In [8] authors used a backpack equipped with 2D laser scanners and inertial measurement units (IMUs) to localize in an indoor environment. Their solution was extended with cameras in [9] to improve the localization with the aim of reconstructing interiors in 3D. We find the solution based on a backpack equipped with laser scanners inappropriate for our use case, since humans have to use this system for prolonged periods of time, pick goods, and repair robots, thus would find the backpack heavy and cumbersome. On the other hand, a camera as a visual sensor fulfills the size and weight requirements and is also low-cost, informative, and highly available [10]. Furthermore, cameras are passive sensors, meaning they do not emit any signals in the environment and there is no limit on the number of cameras that can be used simultaneously in the same environment [11]. Given that, we opted for a solution with a set of cameras aided with an IMU. Several visual indoor localization solutions have been presented recently in the literature. For example, localization with a wearable omnidirectional camera and with a smartphone equipped with Google's Tango sensor was presented in [12] and [13], respectively. In [14] authors used wheel odometry and detection of ground-markers for forklift localization and the solution we propose is similar to this approach. The main difference is that we do not have wheel odometry at our disposal, since sensors are worn by humans, and thus we use visual odometry computed with an IMU aided stereo camera. Moreover, wheel odometry is susceptible to drift due to wheel slippage and is less accurate than visual odometry solutions [15].

Figure 1: Safety Vest with the sensor setup that consists of an IMU-aided stereo camera and a downward looking monocular camera. This placement of the sensors was chosen since it will not disturb the human when performing the usual tasks. Furthermore, cameras cannot get obstructed by hands and this part of the human body is the most stable and has the smallest chance of doing abrupt motion that could blur the images.

There exist numerous localization solutions based solely on IMU and camera sensors, such as [16], [17], and [18], which are well known simultaneous localization and mapping (SLAM) solutions, and [19], which uses convolutional neural network for indoor localization. A natural question arises: "Why not use some of the existing visual SLAM solutions since they do not require any modifications to the warehouse but build the map in which they localize the agent afterward?" For example, the state-of-the-art visual odometry and SLAM solutions, like ORB-SLAM2 [20], DSO [21] and SVO [22], showed impressive accuracy on the datasets like KITTI [23] and EuRoC [24]. Unfortunately, our problem requires a different approach from the standard visual SLAM for several reasons. First, the carry-on prerequisite constrains the size and the weight of the data processing equipment. With frequent loop-closings, which are expected in a typical warehouse, the constrained processing power could not be sufficient to execute some heavy-duty SLAM apporaches in real-time. Second, visual SLAM builds a map of distinctive features under the assumption that they are and will remain static. This cannot be guaranteed in a warehouse, where racks are frequently moving and changing their positions. Third, as visual aliasing is often present in warehouse environments the risk of wrong loop-closing arises. Therefore,

[1] http://safelog-project.eu/

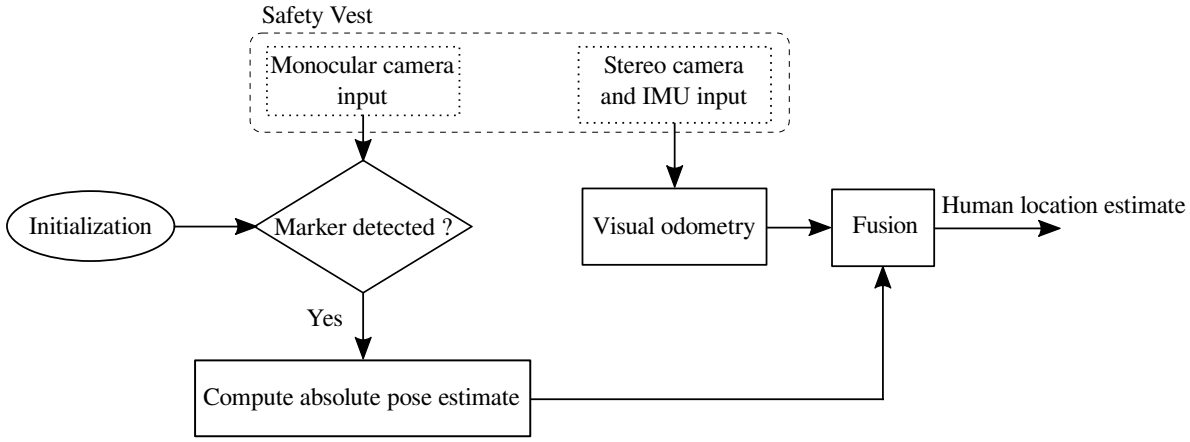


Figure 2: The concept of the proposed visual human localization system.

we are in a need for a solution that infers unambiguous human pose by detecting stable features in the environment. In an automated warehouse we have at our disposal markers that are placed on the ground and originally are intended to be used by robots for localization. The positions of these markers are static and known, and their map is lightweight for processing. We decided to use these markers to ensure unambiguous human pose estimation and to minimize the expensive and time-consuming modifications of the warehouse.

In this paper, we propose a new concept for human localization in robotized warehouses based on wearable sensors that estimates globally the pose of the human with high reliability. The system is based on the fusion of two complementary human pose estimators: (i) relative human pose estimator, i.e. odometry, based on a horizontally-looking stereo camera aided by an IMU, and (ii) an absolute human pose estimator based on the detection of ground-markers using a wearable down-looking monocular high-resolution camera. The former estimator regularly updates the relative human pose with respect to the initial pose as the human moves through the environment, while the latter one provides global pose corrections each time a ground-marker is detected by the algorithm presented in [25], thus preventing pose error accumulation with time which is inherent to the former estimator. To validate the proposed human localization system, we conducted extensive experiments in two settings: (i) a laboratory environment covered with a motion capture system providing localization ground truth, and (ii) a robotized warehouse testing facility that we partially covered with AprilTags and used TagSLAM [26] to provide localization ground truth. We also provide publicly the recorded datasets to the research community[2].

The rest of the paper is organized as follows. In Section 2 we present the concept of our visual human localization system in robotized warehouses, and give short overview of our previously developed visual odometry and marker detection algorithms for the completeness of the paper. The fusion of information from the visual odometry and the ground-marker detection algorithm is described in Section 3. A detailed description of the equipment used and dataset recording, followed by experimental analysis, is given in Section 4. Finally, the paper is concluded in Section 5.

[2]https://zenodo.org/communities/safelog/

2. The proposed visual human localization system

Research presented in this paper is part of the Horizon 2020 project SafeLog, whose goal is to develop a system for safe and interactive collaboration between robots and humans in robotized warehouses. In such warehouses, routing and control is carried out by the so-called fleet management system (FMS) that knows the positions and trajectories of all the robots all the time. However, when we introduce humans in the warehouse, we add agents that FMS cannot control directly, but only give them tasks and suggestions. Moreover, if locations of the humans are unknown to the FMS they can interfere with the robot routing, leading to decreased human comfort, safety issues, inferior performance or even complete halt of the warehouse. To tackle this challenge, a new concept needs to be introduced that would enable the FMS to change the tasks or reroute the robots to ensure continuous optimal performance and aid human comfort and safety. Such a concept requires estimating human trajectories, which is an interesting research topic in itself [27], but first we need to be able to localize the humans.

2.1. The concept of the system

The concept of the proposed visual human localization systems is illustrated in Fig. 2. Since one of our prerequisites is to have a wearable vest with on-board sensors, we found that cameras are the most suitable choice and can provide enough information for accurate localization. Cameras are also appropriate due to low power consumption, price, and also low weight since the whole Safety Vest should be kept light for ergonomic reasons. Additionally, the placement of the camera setup at the lower back proved to be optimal because during typical human motion, camera view does not get obstructed and, since it is the most stable part of the body, camera images do not get blurred often. The cameras also do not record the worker and the images are deleted once used for the pose estimation, so the privacy of the workers is not affected.

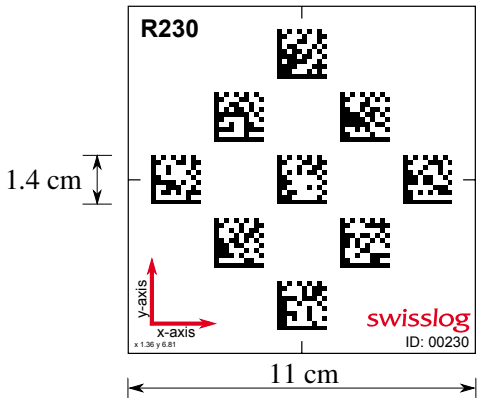


Figure 3: The marker used for localization in the warehouses. Each marker has a specific ID and a different combination of the 9 rectangular patterns, so-called DataMatrix.

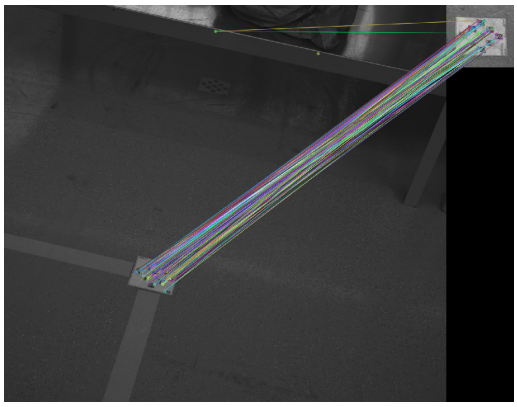


Figure 4: Matching ORB features in the downward-looking camera image to the reference marker image.

The sensor setup consists of two cameras mounted on the Safety Vest (cf. Fig. 1): a monocular camera, pointing to the floor, and an IMU-aided stereo camera, pointing in the horizontal direction. The stereo camera is in charge of the visual odometry that continuously estimates the pose of a human. However, it can only estimate the relative motion of the human and the localization error grows unbounded with time. Given that, it needs to be corrected from time to time and in this paper our idea is to reuse markers that already exists on the floor of the warehouse that robots use for localization. An example of such a marker is shown in Fig. 3. These markers are detected with the downward-looking monocular camera and when a marker is recognized the algorithm provides a global pose of the human, so we can consider it as an indoor GPS-like system. The markers are available at fixed regular intervals of 1.2 m throughout the environment. Whenever a ground-marker is recognized the accumulated error in the human pose estimated by the visual odometry is corrected by applying sensor fusion implemented within a graph optimization framework. Below, for completeness, we provide brief overviews of our stereo visual odometry (Section 2.2) and marker-based localization (Section 2.3) algorithms originally presented in [28] and [25], respectively, while the proposed fusion algorithm is presented in Section 3.

2.2. *Stereo visual odometry pose estimation*

The first source of pose estimation, the visual odometry (VO), computes the pose change between two consecutive image pairs and thus has a constant estimation frequency. We use the SOFT VO algorithm since it is an in-house developed solution and at the time of writing it is the highest-ranking stereo visual odometry solution on the KITTI odometry benchmark [23]. The key to SOFT's performance lays in the careful selection of features through the estimation process. Features are found on the gradient image with the corner and the blob masks. Extracted features are paired with the existing feature set in a matching process. A sum of absolute differences is used as a similarity measure, but since SAD is susceptible to outliers, matching is performed in a circular manner. If the feature is successfully matched through two subsequent stereo pairs, it is considered trustworthy. After the circular matching, the remained set of trustworthy features is further matched with normalized cross-correlation. The tracking step takes matched features but uses only a subset of them based on their age, position, feature strength, and class. It has been shown in [28] that stereo VO performs better when features with a variety in spatial, temporal, and class distribution are used. In [28] two stereo VO solutions were presented: one based solely on the stereo images and one aided with IMU measurements. Our implementation in this paper relies on the IMU aided stereo VO.

2.3. *Marker-based pose estimation*

The marker detection algorithm provides a global pose of the human by computing the transformation matrix between the camera and the ground-markers sparsely placed on the floor throughout the warehouse. Each ground-marker is unique and has a known global pose in the warehouse. As shown in Fig. 3 we can see that the marker detection algorithm has to detect and identify a ground-marker a bit larger than 10 cm with several DataMatrix codes sized 1.4 cm. For a robot this is not a problem, since its camera is placed several centimeters above the floor; however, as can be seen from Fig. 1, the camera suite is placed on the lower back of the human and the marker detection algorithm needs to achieve the same result from more than a meter height.

The marker detection algorithm consists of the following three main steps:

- detection of a ground-marker and finding the region of interest (ROI) around the ground-marker
- ground-marker identification
- computation of the relative camera pose.

The ground-marker detection and ROI search is performed by matching detected ORB features [29] in the input image with the ORB features from the reference image as illustrated in Fig. 4. The cropped image is forwarded to the marker identification step where the ground-markers are identified with the DataMatrix identification algorithm implemented in *libdmtx*

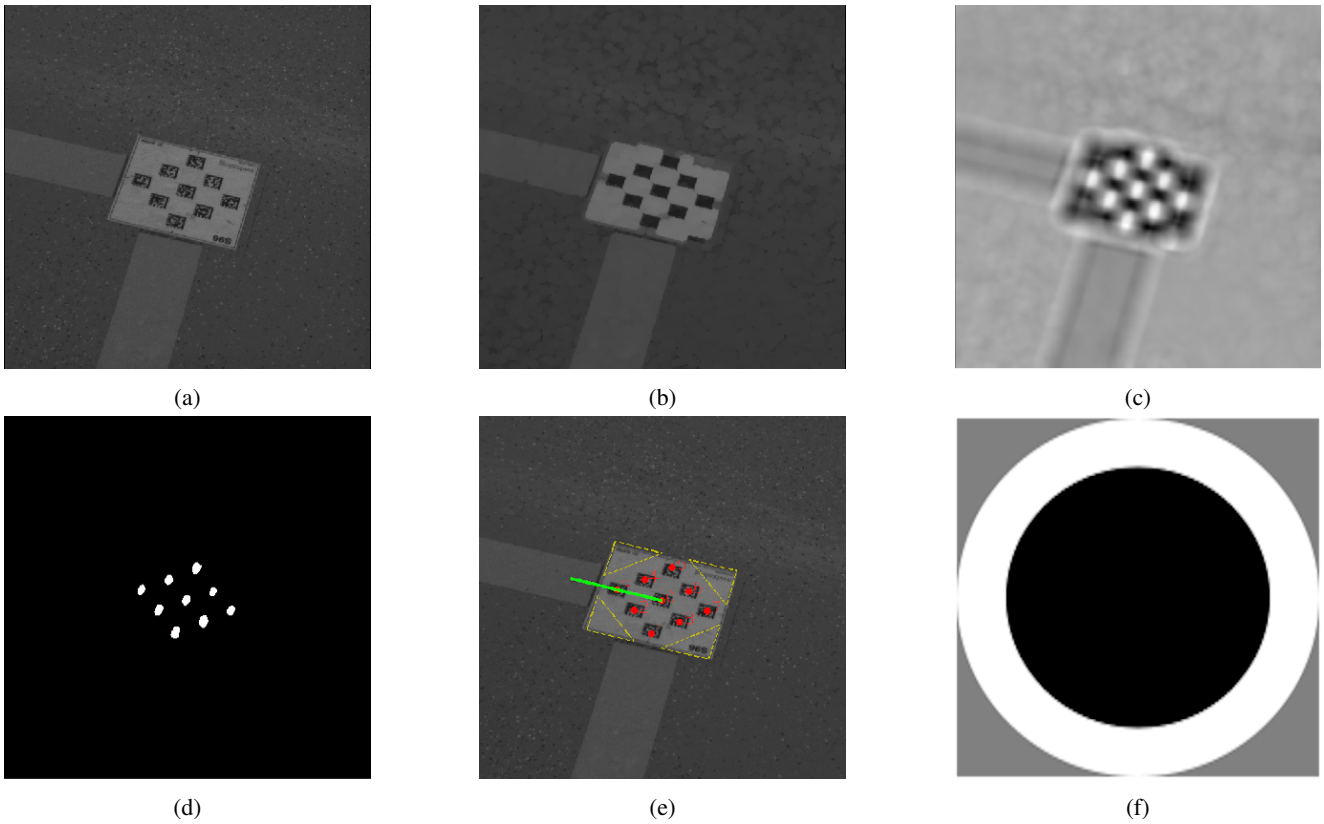


Figure 5: Steps of the marker-based pose estimation algorithm: a) ground-marker in the original image, b) result of the morphological opening, c) image correlated with a double kernel, d) thresholded correlation image, e) marker with the computed orientation, f) the double kernel.

[30]. After finding the ground-marker's ID we know the absolute pose of the ground-marker in the global coordinate frame. To compute the camera pose in the global coordinate frame, we need to determine the relative transformation between the ground-marker and the camera. The steps of computing the relative transformation are shown in Fig. 5. First, the cropped image of the ground-marker, depicted in Fig. 5a, is blurred through morphological opening resulting with image shown in Fig. 5b. This is followed by correlation with the double kernel, shown in Fig. 5f), resulting with the correlated image shown in Fig. 5c. The thresholding of the correlated image produces 9 distinct pixel clusters, as can be seen in Fig. 5d. The centers of those pixel clusters are treated as points that are forwarded to the Perspective-n-Points (PnP) method which then computes the relative transformation as illustrated in Fig. 5e. For more detailed description of the marker detection, please confer [25].

Even though ground-markers are placed on the floor in regular intervals of 1.2 m, due to erratic human motion the downward-looking monocular camera does not always necessarily see each ground-marker. Moreover, even if the ground-marker is seen, the image might be blurred, ground-marker might be partially covered, or the brightness might be inappropriate, which can cause the ground-marker not to be detected. All these issues cause an infrequent marker-based pose estimation that in our experiments can sometimes be absent for more than half a minute.

3. Stereo Odometry and Ground-Marker Fusion

As described in the previous section, a VO algorithm provides relative estimates of the human location which are locally accurate but for longer trajectories global inconsistencies occur. On the other hand, the marker detection algorithm provides a globally accurate location but rarely and unpredictably, i.e., only when a human passes over a ground-marker on the floor and the marker detection algorithm recognizes it. Given that, we can see that these two algorithms complement each other, and our idea is to fuse the information they both provide to get a reliable and globally consistent pose estimate. We have implemented this information fusion of the VO and marker detection algorithms within a graph optimization framework. In the following, we first introduce the graph optimization framework and then our sensor fusion implementation within that framework.

3.1. Graph optimization framework

Graph optimization is nowadays the most popular approach in mobile robotics and related fields for solving localization problems and over the past years most modern visual localization solutions are based on such a framework. [31]. One of the main reasons stems from the fact that, unlike typical filtering based approaches, it can constantly linearize the whole graph around the most recent estimates. In our graph optimization

approach we keep track of all the states, environment map features, and measurements. The goal of the optimization is then to minimize the discrepancy between the measurements and estimated positions of states and map features.

When an agent moves through the environment, as shown in Fig. 6, its trajectory $X_{0:k}$, from the moment it started its motion up to the moment t, consists of a set of states $X_{0:k} = \{X_0, X_1, ..., X_k\}$. The state X_k, at time instant k, consists of the position and orientation of the agent $X_k = \{x_k, y_k, z_k, \psi_k, \gamma_k, \theta_k\}$, while the map consists of a set of discrete features $M = \{M_1, M_2, ..., M_k\}$. Generally, both the agent's states and the observed features are represented in the graph as nodes. The nodes are connected with edges which define the relations between them. The relations arise from two models which are also used in *Bayesian filtering* methods: *process model* and *measurement model*. The process model describes the relation between the successive agent states in time, i.e., it describes the agent egomotion, while the measurement model describes the relation between the map features and the sensor readings for a given agent state. Under the assumption that both state and measurement variables are Gaussian, we can model these relations as

$$X_k = f(X_{k-1}, u_k) + w_k, \quad w_k \sim \mathcal{N}(0, Q) \tag{1}$$

$$z_{k,j} = h(X_k, M_j) + v_k, \quad v_k \sim \mathcal{N}(0, R) \tag{2}$$

where X_k is the state of the agent at the timestamp k, $z_{k,j}$ is the measurement of the feature M_j at the timestamp k, u_k is the traveled distance of the agent from the timestamp $k-1$ to the timestamp k, w_k and v_k are Gaussian process and measurement noise with zero mean and covariance matrix Q and R, respectively, while $f(X_{k-1}, u_k)$ and $h(X_k, M_j)$ are the process and measurement models, which generally can be nonlinear.

For the process and measurement models given in (1) and (2), we can define the graph optimization criterion for finding the optimal states of the agent X^* and features M^* as follows:

$$X^*, M^* = \underset{X_{1:k}, M}{\operatorname{argmin}} \sum_i \|X_i - f(X_{i-1}, u_t)\|^2_{Q_i} + \sum_{i,j} \|z_{i,j} - h(X_i, M_j)\|^2_{R_{i,j}} \tag{3}$$

Minimization of (3) is a complex problem due to optimization of both the agent states and the map features, large number of graph edges, and generally nonlinearity of the process $f(X_{k-1}, u_k)$ and the measurement model $h(X_k, M_j)$. In this paper we use the g2o nonlinear optimization framework [32].

3.2. Graph optimization for stereo odometry and ground-marker fusion

We have made the following adjustments to the graph optimization framework to handle our use case, i.e., to fuse human pose estimations in a robotized warehouse provided by stereo visual odometry and ground-marker detections. The poses of the human and ground-markers are formulated as the members of the SE(3) group

$$X_i = \begin{bmatrix} R_i & t_i \\ 0 & 1 \end{bmatrix}, \qquad G_j = \begin{bmatrix} R_j & t_j \\ 0 & 1 \end{bmatrix}. \tag{4}$$

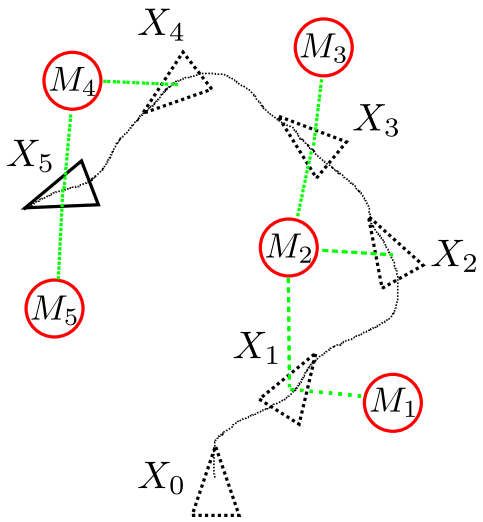


Figure 6: A two-dimensional visualization of an agent (black triangle) moving through an environment with features (red circles). Dashed poses are previous agent poses and the black curve connecting them represents the trajectory. When the agent is sufficiently close to a feature, sensors are able to measure the relative pose of the agent with respect to the feature (green dashed lines).

Once a ground-marker is detected, a marker node G_j and a current human pose node X_i get connected with an edge N_{ij} computed by the marker detection algorithm. Current pose estimate node X_i and the previous pose estimate node X_{i-1} get connected with an edge $U_{(i-1)i}$ computed with the stereo visual odometry. The transformations N_{ij} and $U_{(i-1)i}$ are also SE(3) group members and used to compute the current pose estimates as follows

$$X_i = U_{(i-1)i} X_{i-1}, \qquad X_i = N_{ij} G_j. \tag{5}$$

The poses of the ground-markers G_j are known, which significantly reduces the complexity of the graph optimization, meaning that minimization of criterion (3) is performed only for finding the optimal human poses X^*

$$X^* = \underset{X_{1:k}}{\operatorname{argmin}} \sum_i \|X_i - U_{(i-1)i} X_{i-1}\|^2_{Q_i} + \|X_i - N_{ij} G_j\|^2_{R_{i,j}}. \tag{6}$$

The workflow of the proposed approach is depicted in Fig. 7 and summarized in Algorithm 1. The initialization begins with the detection of the first ground-marker G_0 shown in Fig. 7a. This first ground-marker should be placed at the warehouse entrance to enable the fusion initialization. When the first ground-marker is detected, the marker detection algorithm computes the transformation between the marker pose and the current camera (human) pose X_0; the transformation is represented with a graph edge N_{00}. After the initialization, visual odometry starts to compute human pose change as they move through the warehouse. With the detection of the subsequent ground-marker G_1, Fig. 7b, a pair of two transformation estimates are created: one between the current human pose X_1 and detected ground-marker G_1, denoted as N_{11}, and one between the current pose X_1 and previous pose X_0 represented with the edge U_{01}. The transformations are temporally synchronized meaning than the visual odometry estimate is interpolated so

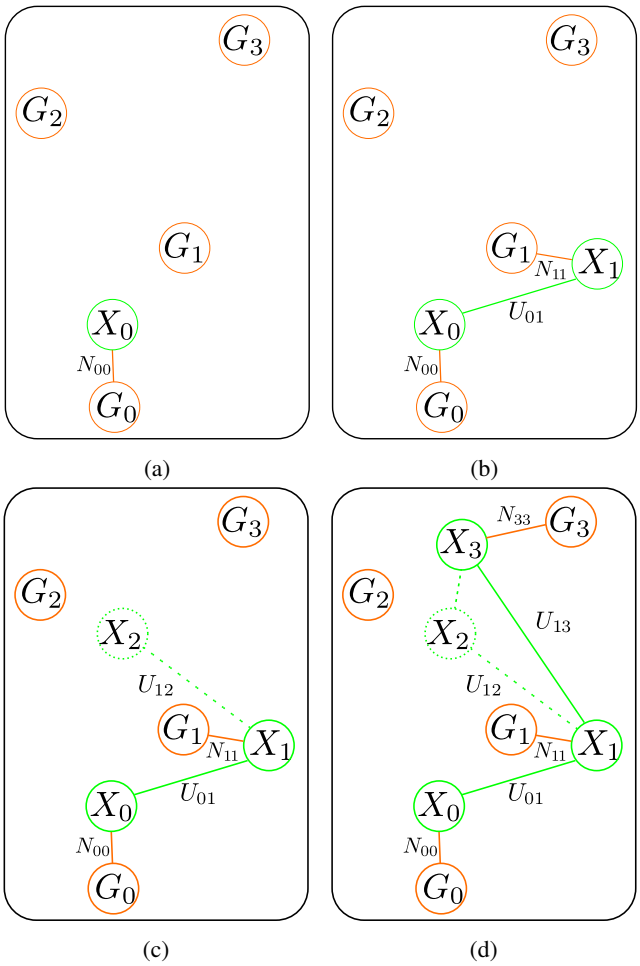


Figure 7: Construction of the pose graph with the pose nodes obtained from visual odometry (X_i) and the ground-marker nodes (G_i). Green nodes and edges represent the output of the visual odometry algorithm, while orange nodes and edges represent the output of the marker detection algorithm.

that the estimate's timestamp matches with the marker detection pose timestamp. As the human wearing the camera setup continues to move through the warehouse, the odometry is continuously updated, as illustrated with the green dashed line and circle in Fig. 7c, but the new information is added to the graph only when the ground-marker G_3 is detected (marker G_2 was not detected and therefore node X_2 was not added to the graph), as we can see in Fig. 7d. New pose nodes and edges, N_{33} and U_{13}, are added to the graph and the graph optimization is executed to correct both the newest pose estimate and the whole trajectory, i.e., poses of all the nodes are optimized. For the optimization implementation of the pose graph we used the g2o framework [32]. The graph optimization runs in parallel to the visual odometry and the ground-marker detection in a separate thread as described in Algorithm 1.

One could argue that for our use case the graph optimization is not necessary and that we could only correct the accumulated odometry drift by setting the pose estimate from the marker detection algorithm as the new pose estimate. This is partially true since we do not need the whole trajectory, but only the newest pose estimate. However, in that case the computed transformation matrix by the ground-marker detection algorithm can exhibit significant orientation errors resulting with unacceptably large location errors. Therefore, we optimize over the whole graph to force the pose consistency and yield a reliable location estimate.

Algorithm 1 Proposed fusion based human localization

main thread:
repeat
  if marker-based pose estimation input **then**
    Set initial pose
until pose initialized.
Initialize graph
repeat
  if VO pose estimate **then**
    if marker-based pose estimates in queue **then**
      Create marker-odometry pair
      Push pair to the pose graph
      Set optimization flag
  if marker-based pose estimation input **then**
    Add marker-based pose estimate to queue
until end of the recording.
optimization thread:
repeat
  if optimization flag set **then**
    Optimize graph
    Return optimized graph
until Killed from the main thread

4. Experiments

We tested the proposed visual human localization system on two datasets we collected in the warehouse testing areas of SafeLog project partners, namely Fraunhofer IML in Dortmund (Dataset 1) and Swisslog in Augsburg (Dataset 2). In the following, we first briefly describe the experimental sensor suite and its calibration, as well as common steps in the datasets collection and results evaluation. Then, we present and compare the results on both datasets obtained with the proposed method and ORB-SLAM2.

4.1. Experimental sensor suite

The experimental sensor suite that we used consisted of two cameras: a monocular downward-looking camera and an IMU-aided stereo camera, and both were placed at the lower-back part of the Safety Vest worn by humans. Figure 8 shows a zoomed-in photo of the sensor suite mounted on the Safety Vest. The monocular camera is a FLIR Chameleon3 CM3-U3-50S5M-CS with a Computar, 12mm, 2/3", 5 MP lens, while the IMU-aided stereo camera is PerceptIn Ironsides. Both cameras were installed on a thick aluminum metal plate which ensured rigid and fixed displacement between the cameras and enabled mounting of the cameras on the Safety Vest. The intrinsic and extrinsic parameters of the sensor setup were obtained through calibration using the Kalibr package [33] and OpenCV library.

Camera calibration was conducted before every experimental run. The camera setup recorded a calibration board with AprilTag patterns [34] from various poses to get the intrinsic and extrinsic parameters. The calibration of extrinsic parameters was more challenging since the stereo camera and monocular camera have very small overlapping field of view, as can be seen from the setup shown in Fig. 8. For that reason, to improve calibration accuracy, we placed an additional camera that shared sufficient view with both cameras in the setup. By adding the transformation from the stereo camera to the newly added camera, and the transformation from the newly added camera to the monocular camera, we could obtain the transformation between the stereo and the monocular camera.

In order to evaluate the localization performance a dataset needs ground truth data; however, the general problem of obtaining ground truth in a warehouse environment is the lack of a precise tracking sensor that could cover such an area. Given that, the first dataset (Dataset 1) that we recorded covers a smaller area which has ground truth available since it was covered with a motion capture system. The second dataset (Dataset 2) covers a larger area, but the ground truth is available only at specific parts of the warehouse and was obtained by elaborate placement of fiducial markers and the SLAM method proposed in [26]. Additionally, for comparison we used a state-of-the-art stereo visual SLAM solution, i.e., ORB-SLAM2. Note that we did not aim to outperform a state-of-the-art visual SLAM algorithm in a general case. In this paper we show that the proposed approach, that relies on stereo VO and detecting very small ground-markers for pose correction, yields comparative performance in warehouse environments, keeps the real-time constraints, but can outperform ORB-SLAM2 in scenarios when the environment is not static.

While collecting both datasets the human wearing the Safety Vest would be positioned at the starting point from which one ground-marker could be seen. This is part of the initialization procedure during which the ground-marker detection algorithm computed the transformation between the camera setup coordinate system and warehouse global coordinate system. After the initialization, the human walked through the testing area and performed usual motion such as fast and slow walk, crouching, bending, grabbing objects from the racks etc. At the end of the recording sequence, the human returned to the starting position.

For evaluation we present three trajectories: the *odometry* trajectory obtained from the stereo visual odometry algorithm without ground-marker corrections, the *fusion* trajectory obtained from the fusion of odometry and ground-marker detections, and the *orbslam2* trajectory obtained by the ORB-SLAM2 algorithm[3]. We distinguish two types of trajectories: *offline* and *online*. The offline trajectories are generated once the experiment is over and all the measurements are available. Although such a trajectory evaluation approach is standard in SLAM benchmarks [23, 24], we also measure localization quality of the online trajectory, which is a set of poses generated only with the measurements collected up to that moment,

[3]We used the following source code of the ORB-SLAM2 implementation for the evaluation `https://github.com/raulmur/ORB_SLAM2`.

Figure 8: The zoomed-in photo of the experimental sensor suite mounted on the Safety Vest.

i.e., no future information is used in the optimization process. For example, in the moments before and after the loop closure, the offline trajectory will have a smooth transition, whereas the online trajectory will have a discontinuity.

As the evaluation metric we used the *absolute trajectory error* (ATE) as is commonly done in SLAM and odometry evaluation [23]. Also, before the evaluation the trajectories were transformed with the Kabsch algorithm [35, 36] to get the best fit between the estimated and the ground truth trajectories. This transformation is also commonly performed and is an option in the package we used for evaluation [37]. A small modification of the ORB-SLAM2 implementation was necessary as it provides only the offline trajectory at the end of the sequence. The problem was solved by sending the newest pose estimate to the standard output. Since we were modifying the ORB-SLAM2 source code to get the online trajectory, all tests were performed with the offline trajectories unless indicated that they were performed with the online trajectories. All the evaluations were perfomed on the Lenovo P51 notebook with Intel Core i7-7700HQ CPU @ 2.80GHz×8.

4.2. Results on the Dataset 1

The testing area was approximately $5{\times}3\,\mathrm{m}^2$ large and equipped with the Optitrack motion capture system. The recording area contained one real rack while other racks and walls were imitated with plastic boxes, as can be seen in Fig. 9. The height of the boxes was set in a way to allow the motion capture sensor to record the ground truth pose of the camera setup and prevent the camera setup to record the outside of the arena. The floor of the testing arena had 6 ground-markers with known poses and the dataset contains nine different sequences of a human carrying out typical tasks while walking and crouching. This dataset was analyzed in two scenarios. The first is

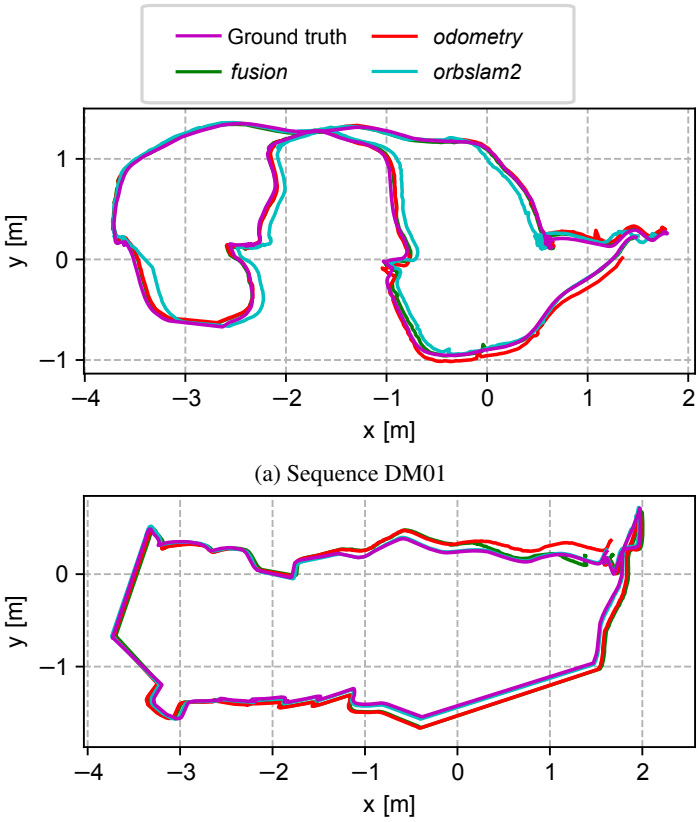
Figure 9: The experimental arena for collecting Dataset 1. Walls and racks were imitated with plastic boxes and the arena was covered with the Optitrack motion capture system. (By courtesy of Fraunhofer IML)

the *standard operating conditions* scenario that contains 9 sequences with the human walking in the warehouse and carrying out typical tasks. The second scenario is the *non-static environment* case where we simulated a varying warehouse rack distribution due to robots redistributing the racks around the warehouse and carrying them to the picking stations.

4.2.1. Standard operating conditions scenario

In Table 1 we show the absolute trajectory errors of the three offline trajectories for 9 sequences in total (DM01–DM09). Furthermore, we include the information about the total traveled distance and the number of detected ground-markers for each sequence. From the table we can see that *orbslam2* had the lowest error on 7 out of 9 sequences. It had no information about the environment, but by consecutive motion and map building it managed to generate a very accurate trajectory for all sequences. This result is expected since it is a state-of-the-art SLAM method and the recordings fulfill the assumption of a static environment with non-reflecting surfaces. On the other hand, the recordings had to be replayed with a significantly lower rate because of frequent track losses in real-time runs, while the *odometry* and *fusion* were able to provide the pose estimate at the frequency slightly over 30 Hz. We can see that in all but one sequence the *fusion* had lower error than the *odometry*, indicating that the challenging marker detection indeed improved the accuracy of the trajectory. Since the traveled distance was not very long, the *odometry* also produced an accurate trajectory, which is noticeable especially for the sequence DM03. The frequency of the ground-markers for this dataset was relatively high and we had a detection every 6–8 meters, during which an error, either accumulated through odometry drift or ground-marker pose detection error, did not reach high values. Compared to ORB-SLAM2, the *fusion* trajectory error is competitive; in the worst case it was 9.1 cm greater, indicating that for the case of the Dataset 1 the proposed method is capable of yielding performance close to the state-of-the-art visual SLAM.

To qualitatively assess the trajectories, we plot them in Fig. 10 for sequences DM01 and DM09. Figure 10a shows an almost perfect fit between the *fusion* and ground truth trajectories for DM01. The trajectory computed only with odometry had a slightly modified shape and the most noticeable difference between it and the *fusion* trajectory is at the lower end of the trajectory. The misalignment of *orbslam2* trajectory comes from the drift in the z-axis which cannot be seen from the currently shown perspective. However, in Fig. 10b, we can see that *orbslam2* was closer to ground truth than other approaches for DM09 sequence.



(a) Sequence DM01

(b) Sequence DM09

Figure 10: Trajectories for two sequences from Dataset 1

The results for the online trajectories are shown in Table 2. As expected, on all sequences the online trajectories showed poorer performances than the offline trajectories. The online *fusion* trajectory managed to keep the absolute trajectory error under 0.3 meters on all sequences. The online *orbslam2* trajectory had a more significant error on sequence DM01. In this sequence, the online *orbslam2* trajectory got deformed due to the error in the orientation estimation, which is not present in the offline *orbslam2* trajectory. The difference between the online and offline *orbslam2* trajectories, and the ground truth are shown in Fig. 11

4.2.2. Non-static environment scenario

The warehouses are not static environments because by definition the robots move racks during the operation. Since the safety system based on relative ranging is still under development, it was not possible to change the environment live while recording the dataset. Therefore, to simulate redistribution of

	DM01	DM02	DM03	DM04	DM05	DM06	DM07	DM08	DM09	DM12345
fusion	**0.044**	0.098	0.107	0.098	0.104	0.072	0.051	0.066	0.091	**0.185**
odometry	0.058	0.122	**0.053**	0.115	0.100	0.141	0.135	0.070	0.080	0.678
orbslam2	0.120	**0.057**	0.057	**0.029**	**0.022**	**0.038**	**0.032**	**0.025**	**0.020**	0.550
distance traveled	24.0	32.4	20.4	22.5	25.8	20.6	20.3	25.0	18.0	125.1
markers detected	4	5	3	5	5	5	8	3	3	22

Table 1: The offline trajectory results for Dataset 1. The first three rows show the absolute trajectory error in meters for each sequence, the fourth row shows the total distance traveled during the recording, and the last row contains the number of detected ground-markers (note that 2 markers are always detected at the start and end of sequence).

	DM01	DM02	DM03	DM04	DM05	DM06	DM07	DM08	DM09	DM12345
fusion	**0.073**	0.109	0.144	0.095	0.158	0.104	0.092	0.069	0.167	**0.268**
orbslam2	0.244	**0.108**	**0.101**	**0.047**	**0.031**	**0.049**	**0.026**	**0.036**	**0.030**	0.592

Table 2: The online trajectory results for Dataset 1. The rows show the absolute trajectory error in meters for each sequence.

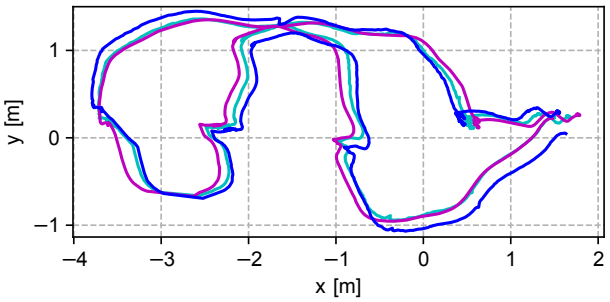


Figure 11: Comparison of the online (blue) and the offline (cyan) *orbslam2* trajectory with ground truth (purple) for sequence DM01.

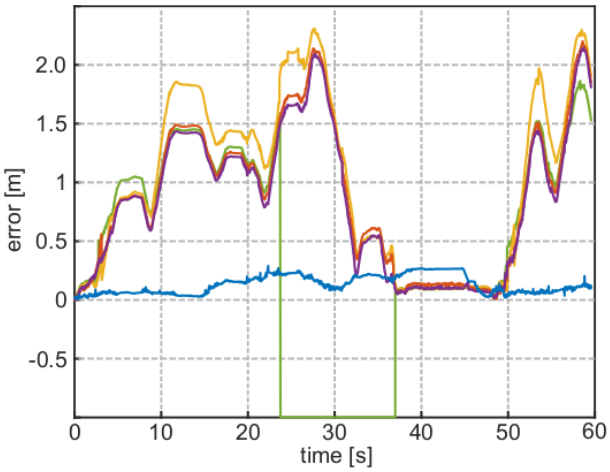


Figure 12: The absolute error of the proposed solution (blue) and multiple runs of ORB-SLAM2 on the sequence DM12345. The value -1 signifies that no pose estimates are produced by ORB-SLAM2 due to losing tracks of features.

racks during operation, we merged multiple sequences with different rack positions. All the sequences started and ended at the same location and we ensured that in all the cases both approaches managed to track features between the sequence ending and starting images without losing the location estimate. We merged the sequences DM01–DM05 into a single sequence and tested the localization quality of the proposed algorithm and ORB-SLAM2. In Table 1, the column DM12345 shows the results of the merged sequences, where we can see that the proposed approach achieved the lowest error. Furthermore, in Fig. 12 we show the absolute error of the proposed solution and ORB-SLAM2, from which we can see that for the majority of the experiments our solution was closer to the ground truth. In one of the runs, at about 30 s, ORB-SLAM2 lost track of the features and this period is marked with the value of -1. From the last row of Table 2 we can see that in the non-static scenario the online *fusion* trajectory had lower error than the online *orbslam2*, although both errors are slightly higher than their offline counterparts.

4.3. Results on the Dataset 2

Dataset 2 is a close approximation of a real use case scenario. The size of the warehouse testing arena was approximately 12×13 m^2 and was filled with metal racks as can be seen in Fig. 13. The main challenge of this dataset collection was acquiring the ground truth values. In contrast to the Dataset 1 arena, this testing facility was not equipped with a motion capture system. After considering all the constraints, we decided to use two approaches for localization evaluation. The first approach was to mark several checkpoints on the floor of the testing arena, whose position was known, and walk through those points during the experiment. The second approach was to accurately compute the location of the cameras with AprilTag markers [34] that we additionally installed on the racks (cf. Fig. 14) and we used TagSLAM [26] to obtain the map of the markers. To ensure the most accurate AprilTag map creation (containing marker location and orientation), we measured by hand the positions of all the markers which were used to initialize the SLAM algorithm. The mapping was done in a separate experiment, during which we focused solely on detecting the AprilTag markers. Once the map was obtained, we used it to compute the localization ground truth data for subsequent experiments. Covering the whole testing arena with AprilTag markers on the racks densely enough, so as to obtain smooth ground truth data for the whole trajectory with our Safety Vest, proved to be unfeasible as during the map building process the optimization would become unstable and crash. Because of that

Figure 13: The experimental arena for collecting Dataset 2. The arena was $13{\times}12\,\mathrm{m}^2$ large, populated with metal racks filled with goods and ground-markers on the floor. (By courtesy of Swisslog)

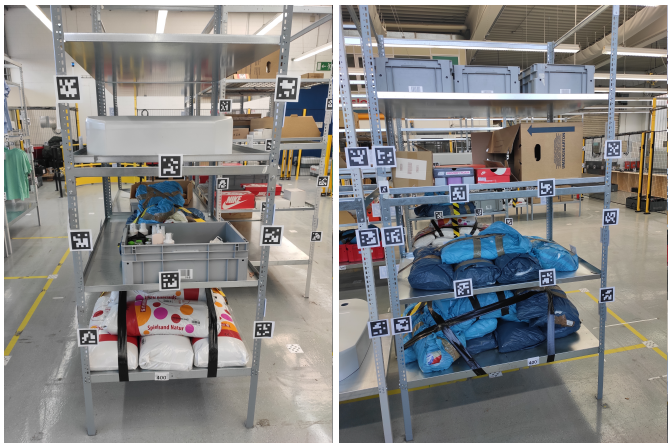


Figure 14: AprilTag markers placed around the warehouse used for localization with the TagSLAM algorithm. (By courtesy of Swisslog)

we focused on two sections of the arena for which we were able to get a reliable ground truth estimate. We assume that the accuracy of the ground truth positioning at those sections is under 20 centimeters. For this dataset we did not conduct comparison between offline and online trajectories, since results in Section 4.2 for Dataset 1 showed that the online error was not significantly larger than the offline error, and, furthermore, ground truth does not cover whole trajectories of Dataset 2, which might preclude such an accurate analysis.

We divided the dataset is divided in three scenarios. The first is the *standard operating conditions* scenario that contains 4 sequences with the human walking in the warehouse and carrying out typical tasks. The second scenario is the *kidnapped human* scenario in which cameras were briefly covered to simulate a situation in which the sensors field of view would get obstructed and localization would fail. Finally, the third scenario is the *non-static environment* case where the racks were redistributed between the sequences.

4.3.1. Standard operating conditions scenario

For this scenario we again evaluated the *orbslam2*, *fusion* and *odometry* trajectories, as we did for Dataset 1. All sequences started and ended at the same point, from which the starting

	AG01	AG02	AG03	AG04
fusion	0.328	**0.191**	**0.303**	0.719
odometry	0.548	0.327	**0.303**	0.692
orbslam2	**0.128**	0.514	0.661	**0.532**
distance traveled[4]	170.6	140.9	83.9	117.0
markers detected	7	4	3	5

Table 3: The results for the Dataset 2. The first three rows show the absolute trajectory error in meters, the fourth row is the total distance traveled for each of the recordings in meters, and the last row is the number of detected ground-markers for the sequence.

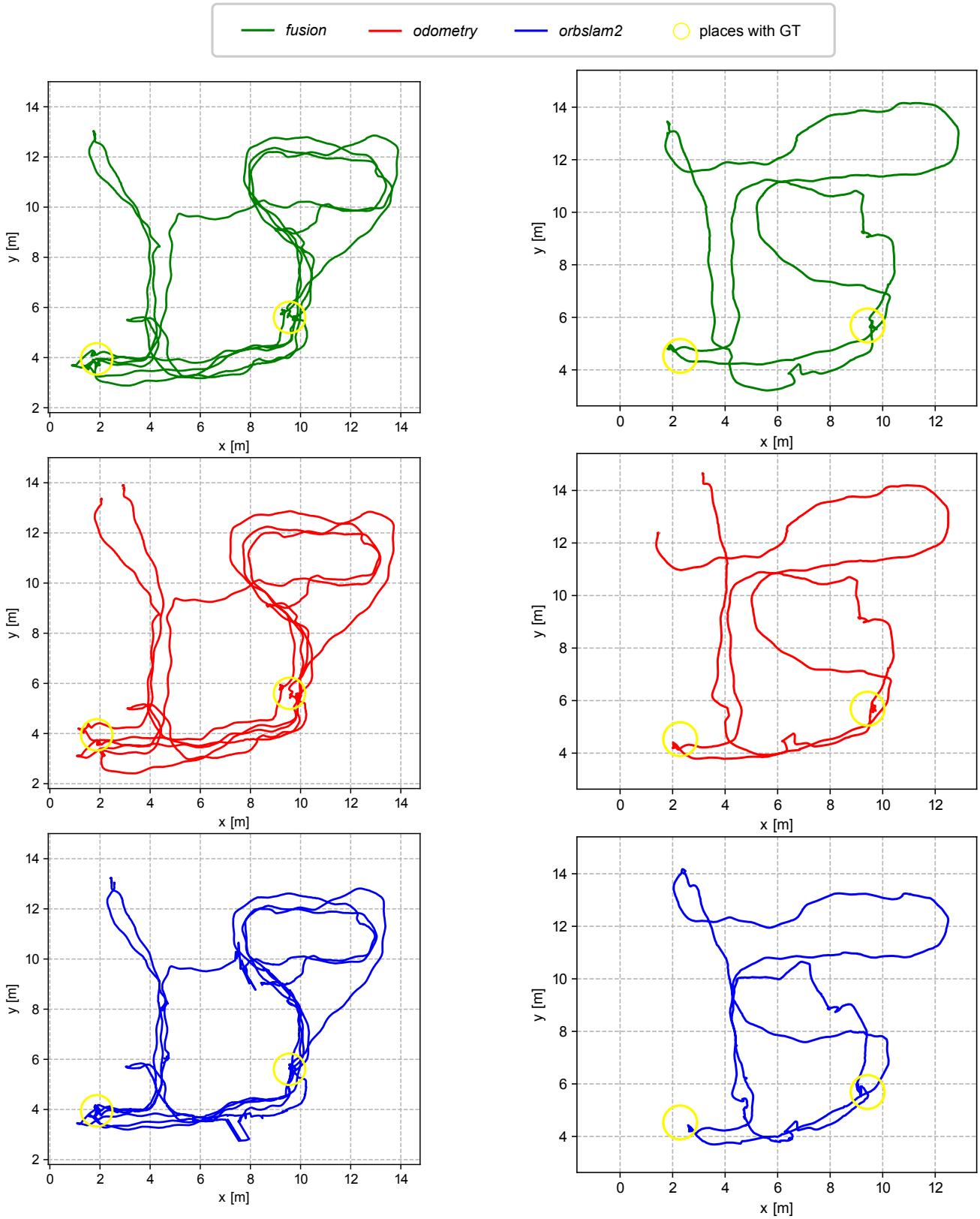
ground-marker could be detected. An example of two trajectories can be seen in Fig. 15 as well as the sections covered with ground truth obtained by AprilTags and TagSLAM. The absolute trajectory errors for all 4 sequences are shown in Table 3, where the error was computed only for those sections of the trajectory where the ground truth pose was available. Indeed, we cannot quantitatively assess the accuracy of the whole trajectory, but we assert that visual inspection coupled with quantitative evaluation at the two sections covered with AprilTags acts as a good indicator of the performance of the algorithms.

From Table 3 we can see that on some sequences ORB-SLAM2 had better accuracy, while on the others the proposed approach dominated. For example, on the AG01 sequence, shown in Fig. 15a, *orbslam2* had the lowest error, while *odometry* had the worst performance – this is also confirmed by the drift that can be seen in the top-left corner of the figure. On the AG02 sequence, even though the ratio of traveled distance and detected ground-markers was high, *fusion* still had the lowest error. The reason for this is the detection of a ground-marker in the vicinity of the evaluation zone. A trajectory with marker detections close to the evaluation zone has a lower impact of the odometry error on the result. Furthermore, not only the number of detected ground-markers is important, but also the accuracy of the marker pose estimate. If the marker detection algorithm returns a pose estimate with large orientation error, it will introduce translation error in further pose estimation with odometry. Another interesting example is the AG03 sequence, shown in Fig. 15b, where *odometry* and *fusion* had the same score that was better than *orbslam2*. Nevertheless, even though odometry was fairly accurate up to the trajectory parts covered by the AprilTags, at the end it still drifted as can be seen in the top left corner of the image. Finally, results for AG04 suggest similar relative performance of the algorithms as in the AG01 case.

In conclusion, the results for the Dataset 2 indicate that all the algorithms had poorer performance than on the Dataset 1, which is expected due to environment complexity and the low ratio of the number of detected ground-markers with respect to the traveled distance; however, even in this case the proposed approach can yield competitive performance with respect to ORB-SLAM2.

4.3.2. Kidnapped human scenario

One of the known localization problems in mobile robotics is the so-called *kidnapped robot* problem, where the robot is taken from its current location during localization and then placed



(a) AG01

(b) AG03

Figure 15: Dataset 2 - trajectory examples. The ground truth was not available along the whole trajectory, but only at the two sections marked with yellow circles. All sequences begin and end at the same position.

to an arbitrary location. The robot then needs to relocalize itself in the environment. Analogously, in our case, we have the *kidnapped human* problem. The problem could occur in situations where the input images of the cameras on the Safety Vest worn by a human are not usable, for example when the camera's field of view gets obstructed.

We simulated this scenario by temporally covering cameras with hands while walking through the warehouse testing area. Without images, neither the visual odometry nor the ground-marker detection could estimate the camera setup movement. After the cameras were uncovered both algorithms continued to estimate the pose, but with the assumption that human did not move while the cameras were covered. This scenario is depicted in Fig. 16, which shows *odometry* and *fusion* trajectories for four moments of the sequence. The trajectory computed with the proposed method is colored in blue, whereas the trajectory computed solely with visual odometry is green. Red obstacles represent the racks in the testing arena. Before we covered the cameras, as seen in Fig. 16a, both trajectories were pretty much aligned, and visible difference comes from the detection of a ground-marker (designated with a red arrow) which corrected the blue trajectory. In Fig. 16b we can see the moment where the cameras were covered (dashed orange circle). During this period, the human moved from the lower left side to the lower right side of the bottom three racks as seen in Fig. 16c. Afterwards, both trajectories continued their normal operation by concatenating newly estimated odometry trajectories on top of their last pose estimates. Figure 16d shows one correction of the blue trajectory when a ground-marker was detected. Even though the pose was not completely corrected with the first detection and the whole trajectory could not be corrected (at some points it still passes through racks), through consecutive corrections the current position of the blue trajectory was accurate and came back to the starting point as seen in Fig. 16e. This is taken to be the true pose since all the sequences started and ended at the same point.

4.3.3. Non-static environment scenario

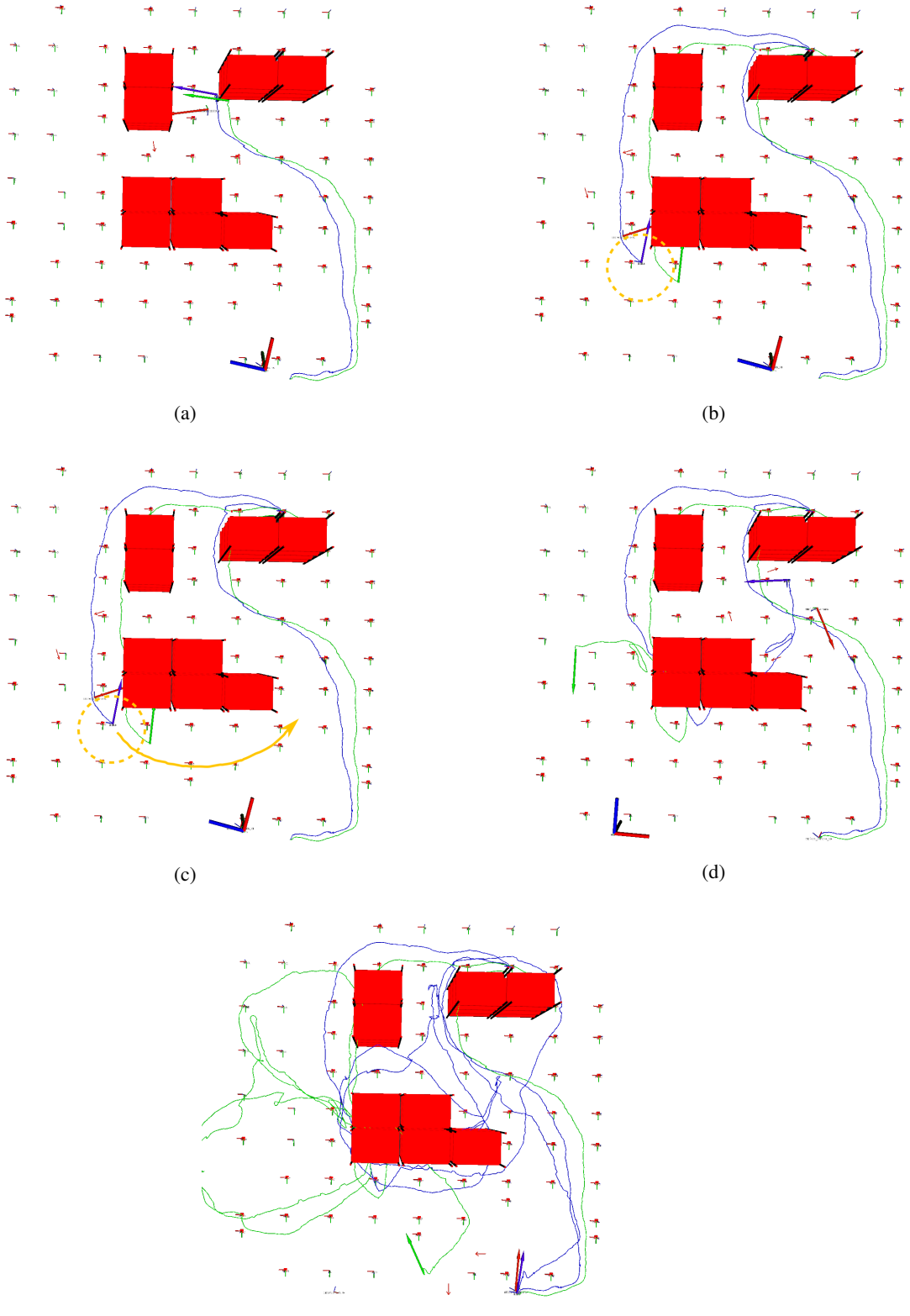
The scenario in which racks are not static, but some of them change their position during the localization process, was also tested on the Dataset 2. Again, the beginning and ending of all the sequences occurred at the same position with very similar appearance, thus we concatenated two recordings with different rack distributions to simulate the changing environment (since this was not possible to be done live during localization due to safety reasons). Again, the proposed approach uses the ground-markers map that always remain fixed, while ORB-SLAM2 builds a map during the first traversal and later the map features might change their position, thus we hypothesize that this could have detrimental effect on the trajectory estimation. Since sequences with different rack distributions were recorded during two separate visits to the testing facility, only one of them had ground truth with AprilTags; thus, for this experiment we could not reliably test the accuracy as for the standard operating conditions scenario. Therefore, to assess the performance in this case, we tracked the number of times the localization algorithm did not return a pose estimate. Namely, when localization fails, both the proposed approach and ORB-SLAM2 stop sending pose estimates. The proposed approach waits for another ground-marker to be detected, while ORB-SLAM2 tries to relocalize in the built map of the environment. Given that, missing pose estimates can act as an indicator of localization reliability and in these experiments we use this metric as a proxy for the evaluation of the algorithms performance.

The sequences we used in this experiment are AG02 and AG03, since we managed to merge them without any localization loss. The results are illustrated in Fig. 17, where we show the localization losses by the algorithms. Note that ORB-SLAM2 has variable performance, which is why we present 5 different runs. From the figure we can see that the proposed algorithm did not have any localization losses, while the mean value of ORB-SLAM2 losses for 5 runs was 16.1% and 11.9% for the merged AG02 and AG03, respectively. For reference, on the unmodified testing facility sequences only ORB-SLAM2 had some losses; specifically, 11.4% on AG02.

There are 4 repetitive localization gaps between 5 runs of the AG02 sequence and 2 repetitive gaps in 5 runs of the AG03 sequence. In Table 4 we can see how much the fusion pose estimate moved while the ORB-SLAM2 was unable to localize. We can see in Fig. 17 that localization gaps for multiple runs of ORB-SLAM2 are almost concurrent, thus Table 4 shows only the gap interval of one run. This observation shows that the localization gaps are not caused by, e.g., the stereo camera being kept still at locations where ORB-SLAM2 lacked features, while the proposed algorithm showed to be more robust. From this we can conclude that the proposed algorithm showed more reliable localization performance when faced with changing environment conditions that are expected in robotized warehouses.

5. Conclusion

In this paper we have proposed a novel approach for human localization in integrated robotized warehouses. The approach relies on a setup of wearable visual sensors, which consists of a downward-pointing camera for detecting ground-markers and a stereo camera for visual odometry. The human location is calculated by fusing the information about the global location inferred from a memory-light map of ground-markers and estimated ego-motion yielded by the stereo visual odometry. To evaluate the proposed approach, we recorded two datasets: the first in a laboratory environment covered with a motion capture system, and the second in a realistic testing facility that was partially covered with AprilTags to generate ground truth. Furthermore, we compared the performance of the proposed approach to a state-of-the-art visual SLAM solution, namely the ORB-SLAM2 algorithm. Results on both datasets showed that the proposed approach yields a robust and real-time localization with accuracy comparable to ORB-SLAM2, without requiring any modifications to the existing warehouses. Furthermore, comparing to ORB-SLAM2, our approach is computationally more lightweight and robust to changes in the environment that can occur frequently in a robotized warehouses due to robots redistributing racks and carrying them to the picking stations.

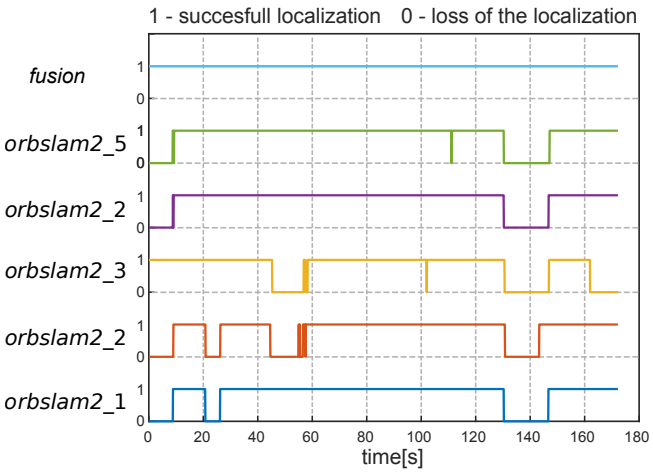


(a) (b) (c) (d) (e)

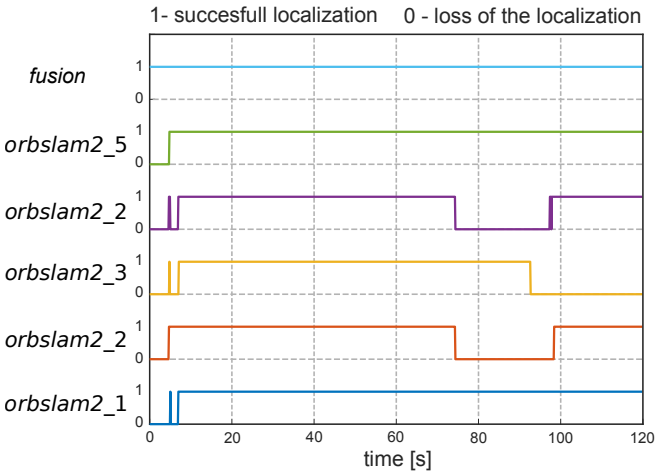
Figure 16: The kidnapped-human scenario captured in four timesteps. Red squares represent the racks present in the testing arena. Blue and green trajectories with the corresponding pose arrows represent the *fusion* and the *odometry* estimations respectively. The red arrow represents the last detected pose with the ground stickers detection. With help from the sticker detection, the blue trajectory manages to return to the starting position, whereas the green trajectory drifts away.

Sequence	Interval [s]	Distance [m]
AG02	0 - 8.2	1.8
	20.3-25.7	6.2
	44.1-56.9	6.6
	127.3-143.3	5.8
AG03	0 - 7.0	2.3
	74.3-97.4	7.2

Table 4: The localization gap intervals and the distances between poses of the *fusion* trajectory at the beginning and the end of each interval.



(a) AG02 merged



(b) AG03 merged

Figure 17: Loss of localization in time for 5 different runs of ORB-SLAM2 and the proposed algorithm on the merged sequences AG02 and AG03.

For future work we plan to integrate the proposed localization with the relative ranging safety system to yield an integrated solution capable of localizing humans and guaranteeing their safety in an environment teeming with robots that have limited perception capabilities.

Acknowledgment

This work has been supported from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688117 "Safe human-robot interaction in logistic applications for highly flexible warehouses (SafeLog)".

References

[1] P. R. Wurman, R. D'Andrea, M. Mountz, Coordinating hundreds of cooperative, autonomous vehicles in warehouses (2008).

[2] Z. M. Bi, M. Luo, Z. Miao, B. Zhang, W. J. Zhang, L. Wang, Safety assurance mechanisms of collaborative robotic systems in manufacturing, Robotics and Computer-Integrated Manufacturing 67 (January 2020).

[3] W. Kim, L. Peternel, M. Lorenzini, J. Babič, A. Ajoudani, A Human-Robot Collaboration Framework for Improving Ergonomics During Dexterous Operation of Power Tools, Robotics and Computer-Integrated Manufacturing 68 (October 2020).

[4] E. Magrini, F. Ferraguti, A. J. Ronga, F. Pini, A. De Luca, F. Leali, Human-robot coexistence and interaction in open industrial cells, Robotics and Computer-Integrated Manufacturing 61 (July 2019) (2020) 101846.

[5] F. Halawa, H. Dauod, I. G. Lee, Y. Li, S. W. Yoon, S. H. Chung, Introduction of a real time location system to enhance the warehouse safety and operational efficiency, International Journal of Production Economics.

[6] C. Losada, M. Mazo, S. Palazuelos, D. Pizarro, M. Marrón, Multi-camera sensor system for 3d segmentation and localization of multiple mobile robots, Sensors 10 (4) (2010) 3261–3279.

[7] J. M. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale, S. Shafer, Multi-camera multi-person tracking for easyliving, Proceedings Third IEEE International Workshop on Visual Surveillance (2000) 3–10.

[8] G. Chen, J. Kua, S. Shum, N. Naikal, M. Carlberg, A. Zakhor, Indoor localization algorithms for a human-operated backpack system, in: 3D Data Processing, Visualization, and Transmission, 2010, p. 3.

[9] T. Liu, M. Carlberg, G. Chen, J. Chen, J. Kua, A. Zakhor, Indoor localization and visualization using a human-operated backpack system, in: 2010 International Conference on Indoor Positioning and Indoor Navigation, 2010, pp. 1–10.

[10] K. Yousif, A. Bab-Hadiashar, R. Hoseinnezhad, An overview to visual odometry and visual slam: Applications to mobile robotics, Intelligent Industrial Systems 1 (2015) 289–311.

[11] M. Aladem, S. A. Rawashdeh, Lightweight visual odometry for autonomous mobile robots, Sensors 18 (9) (2018) 2837.

[12] A. C. Murillo, D. Gutiérrez-Gómez, A. Rituerto, L. Puig, J. J. Guerrero, Wearable omnidirectional vision system for personal localization and guidance, in: 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, IEEE, 2012, pp. 8–14.

[13] M. Li, R. Chen, X. Liao, B. Guo, L. Chen, J. Liu, T. Wu, L. Wang, Y. Pan, P. Zhang, A real-time indoor visual localization and navigation method based on tango smartphone, in: 2018 Ubiquitous Positioning, Indoor Navigation and Location-Based Services (UPINLBS), IEEE, 2018, pp. 1–6.

[14] L. Luyang, Y.-H. Liu, M. Fang, Z. Zheng, H. Tang, Vision-Based Intelligent Forklift Automatic Guided Vehicle (AGV), 2015.

[15] D. Scaramuzza, F. Fraundorfer, Visual odometry part ii, IEEE Robotics & Automation Magazine 18 (2011) 80–92.

[16] J. Engel, T. Sch, D. Cremers, Lsd-slam: Large-scale direct monocular slam (2014) 834–849.

[17] I. Cvišić, J. Ćesić, I. Marković, I. Petrović, Soft-slam : Computationally efficient stereo visual slam for autonomous uavs, Journal of Field Robotics.

[18] R. Mur-Artal, J. M. M. Montiel, J. D. Tardós, Orb-slam: a versatile and accurate monocular slam system, IEEE Transactions on Robotics 31 (2015) 1147–1163.

[19] Y. Wei, B. Akinci, A vision and learning-based indoor localization and semantic mapping framework for facility operations and management, Automation in Construction 107.

[20] R. Mur-Artal, J. D. Tardós, ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras, IEEE Transactions on Robotics 33 (5) (2017) 1255–1262.

[21] R. Wang, M. Schworer, D. Cremers, Stereo dso: Large-scale direct sparse visual odometry with stereo cameras, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 3903–3911.

[22] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, D. Scaramuzza, Svo: Semidirect visual odometry for monocular and multicamera systems, IEEE Transactions on Robotics 33 (2) (2017) 249–265.

[23] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the kitti vision benchmark suite.
URL `www.cvlibs.net/datasets/kitti`

[24] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, R. Siegwart, The euroc micro aerial vehicle datasets, The International Journal of Robotics Research.

[25] G. Écorchard, K. Košnar, L. Přeučil, Wearable camera-based human absolute localization in large warehouses, in: W. Osten, D. P. Nikolaev (Eds.), Twelfth International Conference on Machine Vision (ICMV 2019), Vol. 11433, International Society for Optics and Photonics, SPIE, 2020, pp. 754 – 761.

[26] B. Pfrommer, K. Daniilidis, Tagslam: Robust SLAM with fiducial markers, CoRR abs/1910.00679.

[27] T. Petković, D. Puljiz, I. Marković, B. Hein, Human intention estimation based on hidden markov model motion validation for safe flexible robotized warehouses, Robotics and Computer-Integrated Manufacturing 57 (2019) 182–196.

[28] I. Cvišić, I. Petrović, Stereo odometry based on careful feature selection and tracking, 2015 European Conference on Mobile Robots, ECMR 2015 - Proceedings (2015) 0–5.

[29] E. Rublee, V. Rabaud, K. Konolige, G. R. Bradski, Orb: An efficient alternative to sift or surf., in: D. N. Metaxas, L. Quan, A. Sanfeliu, L. V. Gool (Eds.), ICCV, IEEE Computer Society, 2011, pp. 2564–2571.

[30] M. Laughton, Open source data matrix software & library, https://github.com/dmtx/libdmtx.

[31] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, I. D. Reid, J. J. Leonard, Simultaneous Localization And Mapping : Present , Future , and the Robust-Perception Age 32 (6) (2016) 1–29.

[32] K. Rainer, G. Grisetti, K. Konolige, g 2 o : A General Framework for Graph Optimization (2011) 3607–3613.

[33] P. Furgale, J. Rehder, R. Siegwart, Unified temporal and spatial calibration for multi-sensor systems, in: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2013, pp. 1280–1286.

[34] J. Wang, E. Olson, AprilTag 2: Efficient and robust fiducial detection, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016.

[35] W. Kabsch, A solution for the best rotation to relate two sets of vectors, Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography 32 (5) (1976) 922–923.

[36] W. Kabsch, A discussion of the solution for the best rotation to relate two sets of vectors, Acta Crystallographica Section A 34 (5) (1978) 827–828.

[37] Z. Zhang, D. Scaramuzza, A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry, 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2018) 7244–7251.