Experimental Verification of a Walking Robot Self-Localization System with the Kinect Sensor

Submitted: 15th January 2013; accepted: 5th June 2013

Michał Nowicki, Piotr Skrzypczyński

DOI: 10.14313/JAMRIS_4-2013/43

Abstract:
In this paper we investigate methods for self-localization of a walking robot with the Kinect 3D active range sensor. The Iterative Closest Point (ICP) algorithm is considered as the basis for the computation of the robot rotation and translation between two viewpoints. As an alternative, a feature-based method for matching of 3D range data is considered, using the Normal Aligned Radial Feature (NARF) descriptors. Then, it is shown that NARFs can be used to compute a good initial estimate for the ICP algorithm, resulting in convergent estimation of the sensor egomotion. Experimental results are provided.

Keywords: *3D perception, salient features, iterative closest point, visual odometry, walking robot*

1. Introduction

In recent years the research on Simultaneous Localization and Mapping (SLAM) in robotics heads towards more challenging scenarios, mostly in three-dimensional (3D) environments. Mobile robots moving in 3D require three-dimensional positioning, with regard to six degrees of freedom (DOF). The 6-DOF pose contains the position in three dimensions, orientation, and pitch and roll angles: $\mathbf{x}_R = [x_r\ y_r\ z_r\ \theta_r\ \phi_r\ \psi_r]^T$. Self-localization with regard to 6-DOF is particularly important in the case of walking robots, because the discrete nature of their motion causes sudden and frequent changes in the robot's trunk roll, pitch and yaw angles with regard to the global reference frame [2].

Various sensing modalities can yield the data necessary for 6-DOF SLAM or self-localization. Recent SLAM algorithms focus on the use of passive vision sensors (cameras) [20]. However, passive vision requires specific photometric features to be present in the environment. Encountering areas with ill-textured surfaces and few visually salient features the systems relying on passive vision usually cannot determine the pose of the robot, and easily get lost. Therefore, active range sensors seem to be more practical solution for many mobile robots, particularly those, which have to work in highly unpredictable environments, such like disaster sites explored by robots during search and rescue missions.

Often point clouds yielded by 3D laser scanners are matched against each other in order to obtain the displacement between two poses of the sensor. This approach, known as scan matching, is characterized by high reliability and relative independence from the characteristics of the environment, because in scan matching dense point clouds are matched against each other, not against a particular type of features. The scan matching procedure is typically implemented with the Iterative Closest Points (ICP) algorithm. However, implementation of 6-DOF scan matching with data from laser scanners in most of the walking robots is hardly possible because of the tight size, mass, energy and computing resources limits in such robots – there are no 3D laser scanners that could be placed on board of a typical walking robot. A solution to this problem would be to use a compact 3D range sensor, which has no moving parts. In this context an interesting device is the integrated 3D sensor developed and patented by PrimeSense [21], then introduced to the market by Asus as Xtion and by Microsoft as Kinect. These sensors are compact and affordable, and as shown by the recent research [31], the range measurement accuracy up to the distance of about 3.5 m is comparable to that achieved by a 3D laser scanner.

This paper addresses the issues of reliable incremental self-localization for the six-legged walking robot Messor equipped with the Kinect sensor. Only the range data from Kinect are used, because we would like to have a procedure that is portable to other 3D range sensors, such as the VidereDesign STOC (STereo On a Chip) camera that yields a dense depth map in real-time using hardware-embedded image processing. This camera was already successful used on the Messor robot for terrain perception [15]. Another motivation for using only the range data from Kinect is the fact, that the Messor robot is designed for patrol and search missions in buildings [32], and it should not much rely on the natural lighting of the scene, which in such an application scenario may be insufficient.

We evaluated performance of the standard ICP algorithm on Kinect range data in tests that simulated a scenario typical to a walking robot traversing rough terrain. Because these tests revealed that matching consecutive point clouds obtained from Kinect by using the ICP algorithm requires a good initial estimate of the robot displacement we propose a two-stage point cloud matching procedure exploiting salient features in the range data. The feature-based matching procedure employs the NARF detectors/descriptors [29] and yields reliable initial alignment, which is then refined using the ICP.

This work is an extension of our previous conference papers [16, 17]. The major difference is evaluation of the proposed approach on more realistic in-

door datasets acquired with the Messor robot, and on a publicly available dataset. Moreover, presentation of the state of the art has been significantly extended. In the remainder of this paper we briefly review the state of the art (section 2), then we present the concept of the incremental self-localization system for the Messor robot with Kinect (section 3). Next, we examine theoretical foundations of the point cloud matching algorithms (section 4), and in section 5 we show how the sought rotation and translation between two viewpoints can be computed robustly. We demonstrate experimentally the performance of the proposed self-localization system, at first using an industrial manipulator to move the sensor, and then on the actual Messor robot (section 6). We conclude the paper in section 7 with some remarks as to the directions of further research.

2. State of the art

Nowadays passive vision is considered the most affordable exteroceptive sensing modality for mobile robots, and vision systems receive much attention in the SLAM research. In particular, the monocular EKF-based SLAM algorithm [6] with its numerous extensions is considered one of the most successful solutions to the 6-DOF self-localization problem. However, monocular vision provides only bearing information without a range to the detected features, which leads to data association problems. Although it was demonstrated that a real-time, monocular SLAM can be implemented on a walking robot [28], in a search and rescue scenario, which is considered for our Messor robot, the machine often explores new areas, and avoids returning to previously visited places. Therefore a self-localization system that is rather a form of visual odometry with some mapping capabilities, like in [30], seems to be more appropriate for this task than a SLAM algorithm maintaining a global map of the environment.

Active sensors, such like laser scanners and 3D range cameras do not depend so much on both the natural illumination of the scene, and the ubiquity of salient photometric features. As it was shown in our previous work [27] the pose of a walking robot can be precisely estimated by using 2D laser scan matching and data from an inertial measurements unit. However, this approach works only in man-made environments, where flat walls are available. In an unstructured environment, lacking such features like long flat walls, point clouds yielded by a 3D laser scanner can be used for self-localization with regard to 6-DOF [18], but a 3D laser scanner cannot be used on Messor due to the mass and size limits.

Recently, a number of research teams demonstrated the use of Kinect/Xtion sensors for self-localization or SLAM. The 3D range data stream obtained from a Kinect-like sensor is very intensive. Therefore, the ICP algorithm, which tries to establish correspondences between all the available points requires much time for each iteration. Such amount of data can be processed using a parallel implementation of ICP on GPU [19]. Also the approach presented in [11] heavily uses the GPU to run the ICP algorithm on the Kinect range data. It is demonstrated that this method is enough for real-time tracking of the sensor with unconstrained motion, but this solution is not available for on-board processing in Messor, which uses a single-core x86 processor.

The amount of data being processed can be reduced substantially if salient features are detected in the range or photometric data obtained from the Kinect sensor. Endres et al. [8] demonstrated recently a SLAM system using Kinect RGB and range information. This solution relies mostly on the photometric data for salient feature detection, and then uses the depth image to locate the keypoints in 3D space. While this system shows good performance on large data sets it is unclear how the solution based on photometric features (SIFT, SURF, ORB) will work on the walking robot, which has the sensor pointed partially towards the ground, where the number of photometric features can be much smaller. This system needs also small displacements between the consecutive frames. Therefore, the RGB-D images from Kinect have to be processed at the frame rate, which is impossible with the on-board computer of our robot. Also in [10] features extracted from PrimeSense camera's RGB data are used. Feature matching initializes combined feature and ICP optimization. The RGB-D ICP algorithm proposed in [10] uses photometric features and their associated range values to obtain an initial alignment, and then jointly optimizes over the sparse feature matches and dense 3D point correspondences.

Range-only features are used in [23] by an initial alignment algorithm that transforms the 3D point clouds to the convergence basin of an iterative registration algorithm, such as ICP. In this case persistent feature histograms are used. They are scale and pose invariant multi-value features, which generalize the mean surface curvature at a point. The approach is shown to work indoors and outdoors, but only with rather good quality data from mechanically scanning 3D laser sensors.

3. Concept of the Kinect-based self-localization system

3.1. Kinect as a sensor for the walking robot

The 3D ranging technology used in Kinect is patented by PrimeSense, and thus few details concerning the characteristics of the sensor are publicly available. Therefore, we present a short overview of the principle of operation and basic uncertainty characteristics of Kinect, which are relevant to our application.

Kinect is a device consisting of several components, and is intended to be used for natural interaction with computer games. The basic functionality of the Kinect sensor allows to obtain color images (RGB) and range images of 640×480 resolution, which gives 307200 points in 3D space. By combining information from the range image and the RGB image the RGB-D image can be obtained, in which RGB values

are allocated to the points in the 3D space. The range measurements are obtained by the principle of triangulation, using structured light. The laser projector operates in the infrared range, with the wavelength of about 830 nm. The laser beam passes through a diffraction grating to form a constant pattern of spots projected onto the observed surfaces. The image of this pattern is captured by the CMOS infrared camera, and compared with a stored reference pattern, obtained by projecting the spots onto a plane located at a known distance from the sensor (Fig. 1).

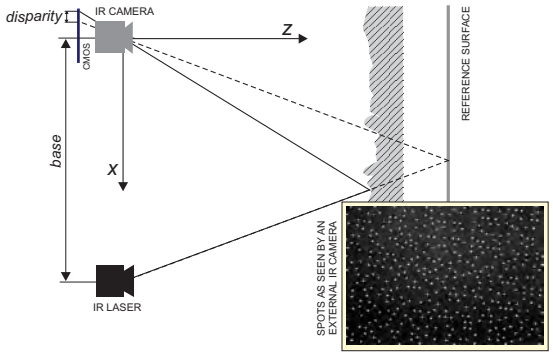

Fig. 1. Principle of operation of the structured light sensor. Inset image shows the actual light spots projected by Kinect on a flat surface

Spatial density of the measured points is defined by the number of projected spots per unit area. It decreases with the square of the measured distance. For the measured range of 2 m the distance between the dots is about 3 mm. Because the number of spots in the projected pattern is smaller than the number of pixels in the captured infrared image, some pixels have interpolated values. The depth resolution decreases with the increasing distance to the sensor. For the measured range of 2 m the depth resolution is about 1 cm, and falls exponentially for longer distances. Kinect is also very prone to errors due to environmental conditions, especially the ambient light. Too strong ambient light, and the sunlight in particular, reduces the contrast of the infrared spots in the captured image, often rendering the range measurements impossible.

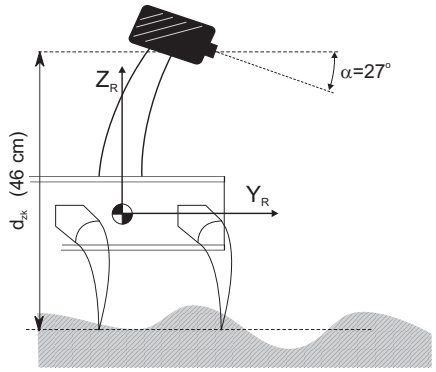

Fig. 2. Geometric configuration of the Kinect-based perception system for the Messor robot

On the walking robot Messor the Kinect sensor is mounted on a mast (Fig. 2). The sensor is located at the elevation of 46 cm above the ground, assuming the neutral (default) posture of the robot. Since the main task of Kinect in this configuration is to observe the area in front of the robot, it was tilted down by 27°. This configuration of the perception system allows both to measure the shape of the terrain in front of the robot, and to perceive larger objects being located farther from the sensor (Fig. 3). These objects are important for the self-localization task, because they often provide more salient features than the terrain itself.

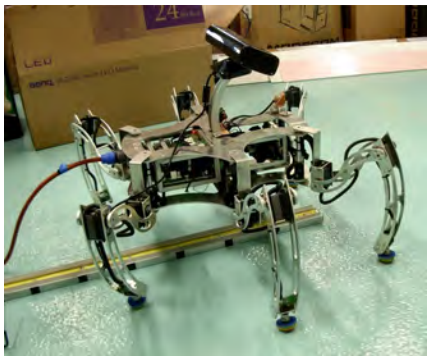

Fig. 3. Messor walking robot with the Kinect sensor

3.2. Implementation of the self-localization method

The most important limitation imposed on the self-localization method by the resources available on our walking robot is the processing speed. The single-board PC used by the robot cannot process the range data at full frame rate, no matter which algorithm is applied for matching of point clouds. Thus, 3D points can be acquired only at selected locations. On the other hand, the ICP with Kinect 3D data is effective only for small values of translation (a few centimeters) and rotation (up to about 5°) between the consecutive point clouds, as it was shown in our preliminary experiments (see Section 6.1). For bigger translations and/or rotations the ICP algorithm usually only finds a local minima. In ICP scan matching on wheeled robots this problem is usually alleviated by obtaining a fairly good initial estimate of the displacement from odometry, but in walking robots proprioceptive sensing is unreliable [9].

Therefore, we are looking for a self-localization system, which uses range data and can process isolated point clouds, tolerating larger displacements (both translations and rotations) between the points of data acquisition. Moreover, we would like to avoid use of any additional hardware (such like a GPU board), which is not available to the Messor robot. Finally, we decided to use feature-based point cloud matching to find a better initial guess for the ICP algorithm. The implementation uses the open source Point Cloud Library (PCL) [24] to manipulate the 3D points, and to detect the salient features. Also the implementation of ICP is taken from the PCL.

The whole procedure of self-localization is implemented in several steps. At first, the range data are trimmed to discard the measurements shorter than 50 cm and longer than 350 cm, that is unreliable or of low

depth resolution. Then, the VoxelGrid filter from the PCL library is used on both data sets with the voxel size of 2 cm. This filter reduces the number of points and decreases the influence of small range measurement errors. We do not use other filters available in the PCL, as their use did not improve the results, while it consumed additional time for processing. NARF keypoints are detected in both point clouds under consideration, and then their descriptors are computed. The next step is matching of the descriptors as described in Section 4.2. The kinematic constraints are imposed in this step. If a sufficient matching is found, the transformation between the two point clouds is estimated, and the data set $\mathcal{A}$ is transformed to the new coordinates. Then, the standard ICP procedure from the PCL library is applied to the two data sets in order to obtain a more precise transformation.

4. Methods for range data matching

4.1. The ICP algorithm

There are many variants of the ICP algorithm described in the literature. The basic version was proposed in [4], while the improvements introduced later involve speeding up the algorithm [22], improving its robustness to noise in range data, and improving robustness to wrong initial guess of the transformation [25]. In this paper, the standard ICP algorithm is used, in the implementation available in the PCL library. This choice makes it possible to asses how much the proposed feature-based matching method improves the self-localization with regard to the baseline method.

Generally, the ICP algorithm is aimed at matching of two sets of spatial data (usually represented by points) describing the same object or part of a scene, and then to represent these sets in a common coordinate system. In each iteration, the ICP algorithm selects the closest points (Euclidean distance) in both clouds as the matching points. For matching points, the rotation $\mathbf{R}$ and translation $\mathbf{t}$ are computed, which minimize the criteria:

$$(\mathbf{R},\ \mathbf{t}) = \arg\min_{\mathbf{R},\,\mathbf{t}} \left\{ \sum_{i=1}^{n_A} \sum_{j=1}^{n_B} w_{i,j} \| \mathbf{p}_{A_i} - (\mathbf{R}\,\mathbf{p}_{B_j} + \mathbf{t}) \|^2 \right\}, \tag{1}$$

where n_A and n_B are the numbers of points in the clouds $\mathcal{A}$ and $\mathcal{B}$, respectively, $w_{i,j}$ are weights defined as $w_{i,j} = 1$ when points $\mathbf{p}_{A_i}$ and $\mathbf{p}_{B_j}$ are closest neighbors or $w_{i,j} = 0$ otherwise. Assuming that points are matched correctly, the transformation $(\mathbf{R},\ \mathbf{t})$ is computed by using the SVD method.

When the transformation that minimizes (1) is computed, the $\mathcal{A}$ set of points is transformed to the new position and new correspondences of points are established. Due to the search for the closest neighbors the computational complexity of ICP is quadratic with regard to the number of points in the two clouds. A more effective way of searching neighbors can be achieved using *k-d* trees to represent the data.

It is necessary to introduce a limit for the maximum Euclidean distance between neighbors, because the clouds may not represent the same part of the scene, and unlimited search for correspondences could yield a completely wrong spatial transformation. This limit, expressed by the $d_{\max}$ distance, prevents the algorithm from matching points that are too far away from each other. The ICP algorithm is only guaranteed to converge to a local minima, and may not reach the global one [4]. Because of this, it is extremely important to have a good initial guess for the sought $(\mathbf{R},\ \mathbf{t})$ transformation.

4.2. Feature-based matching of range data

An alternative approach to matching 3D data exploits salient features of various types. This approach is similar to matching 2D photometric images [14]. It is based on finding some characteristic keypoints in both data sets, then describing them with unique descriptors, and finding the matching between the computed descriptors. These features are salient points, which do not represent pre-specified structures, and commonly appear in both man-made and natural environments.

Because we are interested in the range data from Kinect, which we consider to be a richer representation of the environment than the photometric (RGB) data, we choose the Normal Aligned Radial Feature (NARF) detector/descriptor for our implementation of the feature-based point cloud matching procedure. The NARF concept was introduced for object recognition from 3D range data [29]. The NARF detector searches for stable areas with significant changes in the neighborhood, that can be keypoints looking similar from various viewpoints. Detection is done by determining the plane minimizing the mean square error between the processed point, and other points contained in the predefined sphere around this point. When the mean square error does not exceed the given threshold, the tested point is considered to be a NARF keypoint. The most important parameter of the NARF detector is the diameter of the sphere around the processed point. This sphere contains points, which dominant directions are used to determine the "coefficient of interest", and then to detect the keypoints.

Then, descriptors are built upon the areas around the keypoints. The NARF descriptor characterizes the keypoint by determining depth changes in each direction, calculating an aligned range value patch, and identifying the dominant orientation of this patch. Thus, descriptors are the normalized values of gradient in n directions with regard to the keypoint. The number of directions is also the length of the NARF descriptor, here n=36 is used. The number of NARF descriptors in a dataset depends on the environment characteristics, but in some cases hundreds of descriptors in one point cloud can be established.

5. Transformation between two point clouds with salient features

Once the keypoints and NARF descriptors are determined in both point clouds under consideration it

is necessary to compute the rotation **R** and translation **t** between these two data sets.

In this step of the self-localization method the input data consists of a point set $\mathcal{A}$ (the first cloud) and a point set $\mathcal{B}$ (the second cloud). The set of all possible pairs of descriptors is defined as $\mathcal{Z}$. For each descriptor of the first point cloud all possible pairings with the descriptors of the second cloud are established. For the set $\mathcal{Z}$ the matchings are defined by the descriptor dissimilarity function, which in our application is given as:

$$f_k = \min_{j=0,1,\ldots,m_B-1} \sum_{i=1}^{36} |p_k[i] - q_j[i]|, \tag{2}$$

where f_k is the value of dissimilarity, which is the sum of absolute differences between 36 values describing the k-th descriptor of the cloud $\mathcal{A}$, and the corresponding values of the descriptor of the cloud $\mathcal{B}$. The descriptor of the cloud $\mathcal{B}$ is chosen to minimize the value of f_k. In equation (2) $p_k[i]$ is the i-th value of the k-th descriptor of the first point cloud, while $q_j[i]$ is the i-th value of the j-th descriptor of the second point cloud, and m_B is the number of descriptors of the second cloud. Then again, all pairings of the given descriptor of $\mathcal{A}$ are investigated, and the pairs for which the dissimilarity value is greater than 3 times f_k are discarded as incorrect.

Unfortunately, among the accepted pairing there could be incorrect correspondences, minimizing (2), but leading to a clearly incorrect estimate of the egomotion. To solve this problem we exploit the fact, that the walking robot is a physical system, having limited motion capabilities with regard to acceleration, velocity, and direction of motion. Taking this into account it is possible to introduce a set of constraints on the maximum translations and rotations with regard to the particular axes of the robot's coordinate frame. These constraints are used to discard pairs of the descriptors for which the distance between their keypoints is bigger than the maximum distance:

$$d_{\text{des}} = \sqrt{d_{\text{des}_X}^2 + d_{\text{des}_Y}^2 + d_{\text{des}_Z}^2}, \tag{3}$$

where d_{des_X} is the maximum distance in x_r axis, d_{des_Y} is the maximum distance in y_r axis, and d_{des_Z} is the maximum distance in y_r axis. Formula (3) is simplified in order to allow fast computations, and does not take explicitly into account rotation of the robot, but for the limited range of Kinect measurements it is a reasonable approximation, as we do not expect to have keypoints on objects located far away. This condition allows to discard most of the incorrectly matched pairs from the $\mathcal{Z}$ set.

However, the set of matched NARF pairs still can contain some outliers. Therefore, the point cloud transformation algorithm is applied within the robust estimation framework based on the RANSAC scheme. In each of its iterations RANSAC defines a subset $\mathcal{N}$ of the set $\mathcal{Z}$, which contains 3 unique descriptor pairs. The next step of the algorithm computes the transformation between those pairs:

$$\arg\min_{\mathbf{T}} \| \mathbf{T}\,\mathbf{P} - \mathbf{Q}\|, \tag{4}$$

where **T** means the transformation combined of rotation **R** and translation **t**, while **P** and **Q** are matrices built from the coordinates of keypoints as follows:

$$\mathbf{P} = \begin{bmatrix} x_{11} & y_{11} & z_{11} \\ x_{12} & y_{12} & z_{12} \\ x_{13} & y_{13} & z_{13} \end{bmatrix}, \tag{5}$$

$$\mathbf{Q} = \begin{bmatrix} x_{21} & y_{21} & z_{21} \\ x_{22} & y_{22} & z_{22} \\ x_{23} & y_{23} & z_{23} \end{bmatrix}, \tag{6}$$

where x_{1i}, y_{1i}, z_{1i} are the coordinates of the i-th keypoint from the cloud $\mathcal{A}$, belonging to the set $\mathcal{N}$, while x_{2j}, y_{2j}, z_{2j} are the coordinates of the j-th keypoint from the cloud $\mathcal{B}$, also belonging to the set $\mathcal{N}$.

The 3D rigid body transformation that aligns two sets of points for which correspondence is known can be computed in several ways. In [7] four algorithms are compared, each of which computes the rotation **R** and translation **t** in closed form, as the solution to a formulation of the problem stated by (4). The results of comparison presented in [7] show that for the algorithms under study there is no difference in the robustness of the final solutions, while computing efficiency mainly depends on the implementation. Therefore, to solve (4) we chosen the algorithm based on Singular Value Decomposition (SVD) computation using the standard (**R**, **t**) representation of the transformation. Although this method has been for the first time proposed by Kabsch [12] often a much later work on computer vision [1] is cited as the origin.

After computing the transformation **T** new coordinates for the points of the cloud $\mathcal{A}$ are calculated. The next step involves computing the matching error:

$$\epsilon = \sum_{i=1}^{n_A} (\mathbf{h}_i - \mathbf{d}_j)^2 \quad \text{for} \quad j = \arg\min_{j=0,1,\ldots,n_B-1} (\mathbf{h}_i - \mathbf{d}_j)^2, \tag{7}$$

where $\mathbf{h}_i$ is the vector of coordinates of i-th keypoint of the point cloud $\mathcal{A}$ transformed by **T**, $\mathbf{d}_j$ is the vector of coordinates of the j-th keypoint of the cloud $\mathcal{B}$, which is the closest to $\mathbf{h}_i$. When the value of $(\mathbf{h}_i - \mathbf{d}_j)^2$ is greater than a fixed parameter, the keypoint is treated as an outlier and the computed distance is not counted in (7).

The necessary condition of stopping RANSAC is achieving the error ϵ below a preset value, or not improving the best model for a fixed number of iterations, or exceeding the number of maximum iterations [17]. Alternatively, in the new version of our software the necessary number of RANSAC iterations can be estimated using a simple probabilistic model [5], which slightly improves on the computing time.

When RANSAC is finished, the transformation with the minimal error is performed on all points of the cloud $\mathcal{A}$. This method transforms the point clouds to the convergence basin of the ICP algorithm, thus improving the chance of finding the global minimum. The

points belonging to cloud $\mathcal{A}$ are transformed to the new coordinates by the formula:

$$\mathbf{p}'_A[i] = \mathbf{R}\,\mathbf{p}_A[i] + \mathbf{t} \quad \text{for} \quad i = 1, 2, ..., n_A \qquad (8)$$

where $\mathbf{p}_A[i]$ means the i-th point of the first point cloud. The newly computed set $\mathcal{A}'$ is then the point cloud for the ICP algorithm searching for a more precise transformation of $\mathcal{A}'$ into $\mathcal{B}$.

6. Experimental evaluation of the method

6.1. Preliminary tests

Testing Kinect as a sensor for walking robot self-localization we conducted several experiments to evaluate robustness of the basic ICP algorithm and our NARF-based method for initial alignment of the point clouds. These initial tests were performed on pairs of point clouds obtained for various data sampling rates along the path of the sensor motion. First tests were aimed at determining the approximate maximum translational and rotational distance for which two Kinect point clouds can be successfully aligned by the ICP algorithm. Then, we tested if these limits can be relaxed by applying the NARF-based method.

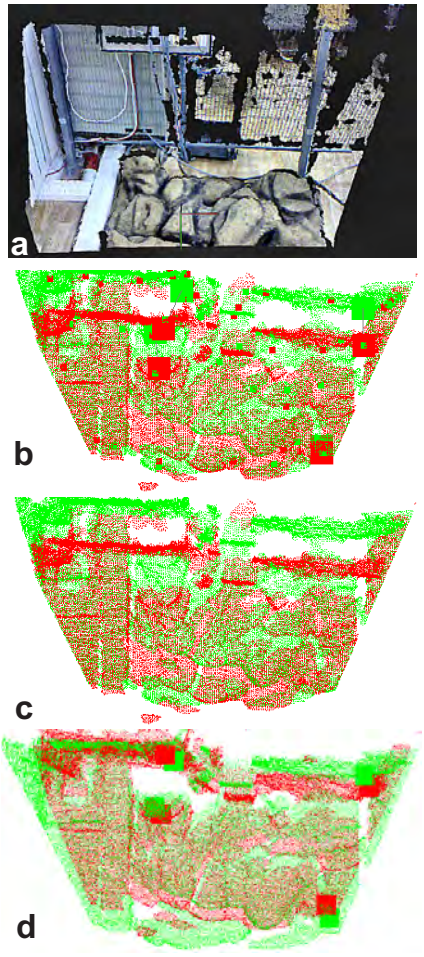

Fig. 4. Experiment with the industrial robot: terrain mockup as seen by Kinect (a), point clouds with NARFs prior to matching (b), matching result with ICP (c), matching result with NARF+ICP (d)

Experiments were carried out using the Kuka KR-200 industrial robot, which allowed us to obtain ground truth for the sensor's trajectory. The Kinect sensor was moved over a mockup of rocky terrain. The sensor was attached to the robot's wrist and tilted down at an angle of 27°, as in the configuration used on the walking robot. An RGB-D rendered image of the experimental setup is shown in Fig. 4a. The processing times mentioned further in the paper were measured under Linux on a low-end PC notebook with an Intel Core2duo T6500 2.1GHz processor and 4GB of RAM. Although the notebook had a dual-core processor the single-thread program didn't use this feature, effectively running in a single core configuration, similar to the one used on the actual Messor robot.

In the experiment shown in Fig. 4 the robot's end effector moved 20 cm along a straight line. Two datasets were obtained at the beginning, and at the end of the trajectory (Fig. 4b). The first dataset ($\mathcal{A}$) is depicted by darker points, while the second one ($\mathcal{B}$) is shown in lighter shade of grey. The small squares mark NARF keypoints for each of the point clouds, while descriptors that were used to determine the transformation between the clouds are marked with larger squares.

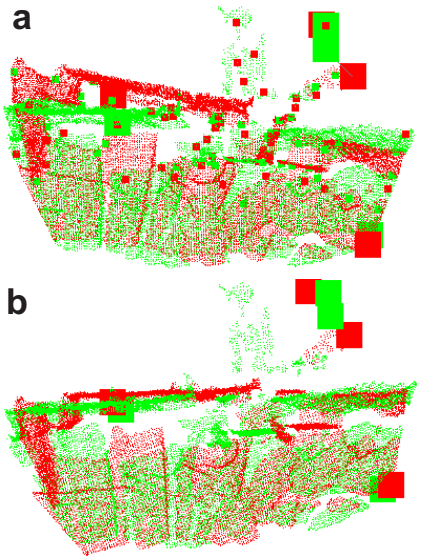

Fig. 5. Rotational motion of the sensor: point clouds with NARFs prior to matching (a), matching result with NARF+ICP (b)

Results of this simple test have shown that for the displacement of 20 cm the standard ICP algorithm was unable to find any satisfying estimate of the egomotion of the sensor (Fig. 4c). It was not possible despite the fact that we have tested various values of the $d_{\max}$ parameter and allowed for thousands of iterations of the algorithm. In our experiments for large initial displacements between the point clouds the ICP algorithm usually converged to a local minima, yielding a wrong egomotion estimate. However, the NARF-based matching algorithm was able to find a good initial transformation of the $\mathcal{A}$ dataset, located in the convergence basin of the ICP algorithm. Thus, the ICP used as the second step of the self-localization method was able to find an acceptable estimate of the displacement: 17.3 cm, and a good alignment of the two clouds

(Fig. 4d). The time required by particular parts of the registration procedure was: detection and description of the NARFs 0.84 s, matching with the RANSAC procedure 0.25 s, and then the ICP algorithm 0.27 s.

A similar experiment was performed with the rotational motion of the sensor. The robot wrist was rotated by 10° between two poses of point cloud acquisition (Fig. 5a). Again, the standard ICP could not find a satisfying transformation between the clouds. The initial displacement found by the NARF-based method was quite imprecise – a rotation by 4.7° and small translations along all axes. However, this initial transformation turned out to provide a good enough starting point for the ICP algorithm, transforming the clouds to its convergence basin. As the result of the two-step registration procedure a good alignment was achieved (Fig. 5b), with the rotation value close to the actual 10°.

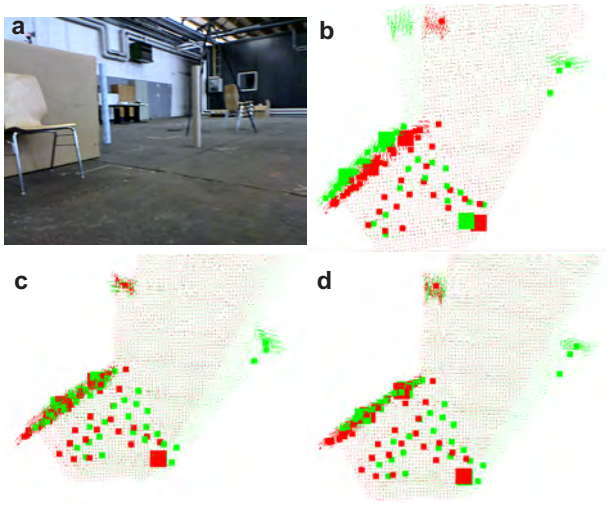

Fig. 6. Results for the publicly available dataset: a view of the environment from the Kinect camera (a), point clouds with NARFs prior to matching (b), NARF-based initial alignment (c), matching result with NARF+ICP (d)

Unfortunately, we do not have a motion capture system that can provide reliable ground truth for environments more extended than the relatively small mockup used with the industrial robot. Therefore, we evaluated the performance of our self-localization system also on a publicly available dataset[1]. This dataset, which contains RGB-D data from Kinect, and time-synchronized ground truth poses of the sensor obtained from a motion-capture system was recently used to evaluate a SLAM algorithm [8]. Figure 6a shows an example RBG image taken from this dataset, while Fig. 6b presents two point clouds taken from this sequence (the first one is synchronized with the shown RGB image) with detected NARF keypoints. As it can be seen in Fig. 6c the NARF-based matching procedure provides quite good initial alignment of the two clouds, which is however further refined by the ICP method (Fig. 6c). Quantitative results (in meters) for this experiment are presented in Tab. 1. For this dataset the detection and description of NARFs took 0.735 s, and the estimation of initial alignment using RANSAC required 0.291 s.

Displacement [m]	Δx_r	Δy_r	Δz_r
Ground truth	0.040	-0.028	-0.011
NARF-based alignment	0.069	-0.051	-0.076
NARF+ICP estimate	0.046	-0.048	-0.040

Tab. 1. Estimated egomotion for the publicly available data

6.2. Indoor tests on the Messor robot

Next, the proposed method was verified on the Messor robot. Because we do not have any external motion capture system to measure the pose of the robot during the experiments, we compare the results of the Kinect-based self-localization to the results obtained from another self-localization system – Parallel Tracking and Mapping (PTAM) software [13], which uses monocular vision data from another camera mounted on Messor. The PTAM system offers precise tracking of the camera pose, and was already evaluated on the Messor robot with positive results [3]. However, PTAM works well only in environments rich in photometric features, and is limited to rather small workspaces. Therefore, PTAM was unable to provide self-localization results for longer trajectories, and we have decided to show here only results of pair-wise matching of two point clouds, similarly to the the industrial robot experiments.

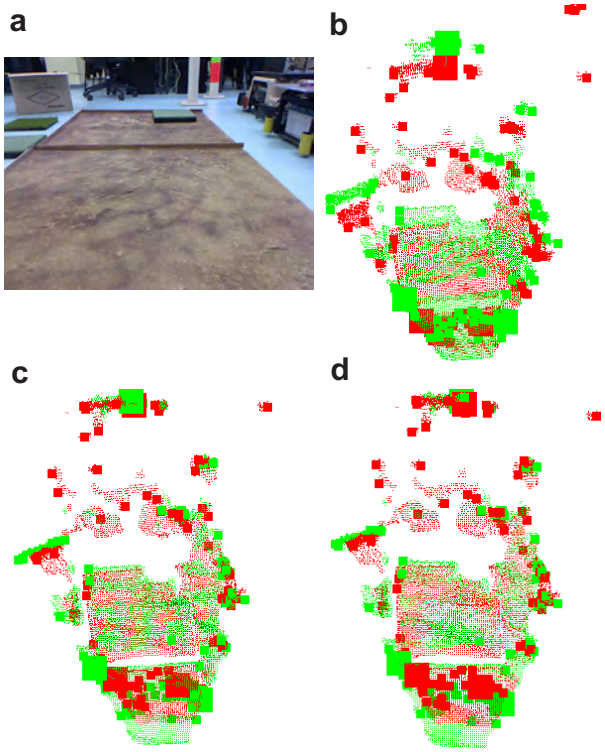

Fig. 7. Results for the Messor robot traversing a simple mockup: a view from the Kinect camera (a), point clouds with NARFs prior to matching (b), NARF-based initial alignment (c), matching result with NARF+ICP (d)

Although Messor does not have reliable odometry, the knowledge of the motion direction and the estimated length of the step was used to impose kinematic

constraints on the egomotion estimates computed by the self-localization algorithm. The constraints are not so tight: 50 cm translation along each axis, and 30° rotation around each axis are allowed. These values are not violated even by sudden and unpredictable motions related to slippages of the walking robot [2]. They simply reflect the fact, that the robot cannot be "teleported" during the relatively short time between the two moments of range data acquisition.

Displacement [m]	Δx_r	Δy_r	Δz_r
PTAM results	0.043	0.188	0.003
NARF-based alignment	0.047	0.186	0.069
NARF+ICP estimate	0.039	0.187	0.007

Tab. 2. Estimated egomotion for the simple mockup experiment

The first dataset was obtained for translational motion of the Messor robot on a simple terrain mockup, which is visible in Fig. 7a. This environment contained few larger geometric features, so the NARF descriptors were mostly located on the borders of the mockup and on more distant objects, outside the mockup (Fig. 7b). Table 2 provides the egomotion estimation results for two versions of the range-data-based point cloud registration procedure: the initial transformation with NARF keypoints (Fig. 7c), and the two-step procedure applying the ICP algorithm to the initially aligned point clouds (Fig. 7d). These results are compared to results provided by the PTAM system, which could be considered very accurate in this experiment, because the surface of the mockup is rich in photometric features. The NARF detection, description, and computation of the initial alignment required 1.055 s, while the whole self-localization procedure between two point clouds, with the NARF-based alignment and ICP took 1.543 s on a single-core PC.

A similar experiment was conducted in one of corridors in our building. This corridor is rather wide and contains some furniture, as seen in Fig. 8a. The camera yielding images for the PTAM algorithm is mounted on Messor similarly to the Kinect sensor – it is tilted down in order to see the surface in front of the robot. Therefore, PTAM in our application relies mostly on photometric features found on the ground. In the corridor the floor is very dark, and few natural photometric features can be found here, which makes PTAM quite unreliable in this environment. Therefore we put some small stones on the floor to increase the number of features for PTAM. Table 3 provides results for the initial NARF-based transformation (Fig. 8c), and the full NARF+ICP version of the proposed method (Fig. 8d). In this case the NARF detection, description, and computation of the initial transformation took 1.756 s, because more RANSAC iterations were needed to find an acceptable initial estimate of the egomotion. This was in turn caused by small number of NARFs that were found on the floor. The whole self-localization procedure including ICP required 2.315 s. However, these results are still acceptable for our application, as the

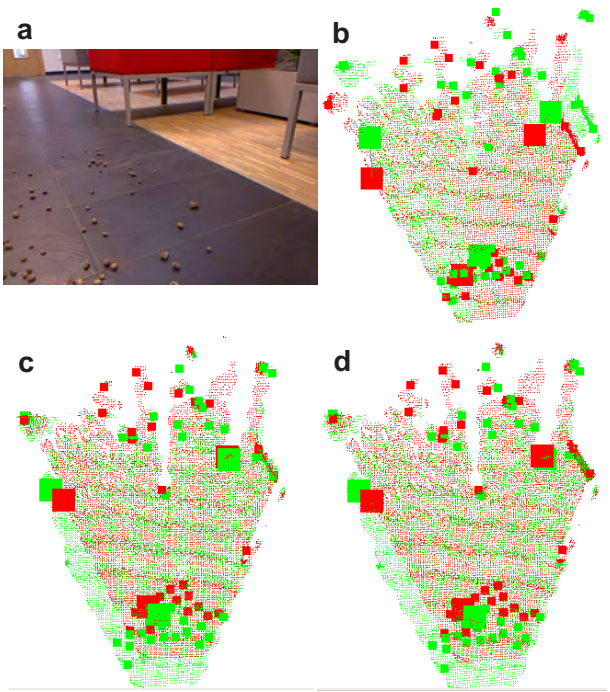

Fig. 8. Results for the Messor robot moving in a corridor: a view from the Kinect camera (a), point clouds with NARFs prior to matching (b), NARF-based initial alignment (c), matching result with NARF+ICP (d)

walking robot needed four seconds to cover the distance of about 20 cm, using a tripod gait and making steps of the length of 5 cm. Thus, our system was able to complete all self-localization steps between the two consecutive data acquisition events.

Displacement [m]	Δx_r	Δy_r	Δz_r
PTAM results	-0.027	0.217	0.016
NARF-based alignment	-0.468	0.246	0.021
NARF+ICP estimate	-0.042	0.244	0.018

Tab. 3. Estimated egomotion for the corridor experiment

6.3. Outdoor tests on the Messor robot

In order to further verify the Kinect applicability to walking robot self-localization, experiments were carried out also in outdoor environment (Fig. 9a). When the sensor was exposed to direct sunlight, it was not possible to get any usable range measurements. However, it was possible to use Kinect for self-localization in the shade of a building wall. Because of the influence of sunlight (even in a shaded place) the obtained point clouds had only about 30% of the number of points typically obtained indoors (Fig. 9b). During the experiment, the robot moved forward, roughly along its y axis, on slightly uneven ground. The reduced number of points turned out to be sufficient for the proposed algorithm. This is due to the fact that the visible (measured) areas are almost the same for both point clouds (Fig. 9c). Thus, the algorithm has found the correct displacement in spite of the environment conditions being difficult for the Kinect sensor (Fig. 9d). The time

required to find the NARFs and build the descriptors was only 0.15 s, obviously due to to the small number of points. The RANSAC procedure took 0.28 s – this value is quite stable across all our experiments. Finally, the ICP procedure required only few iterations to converge from a very good initial guess, thus the time was only 0.06 s.

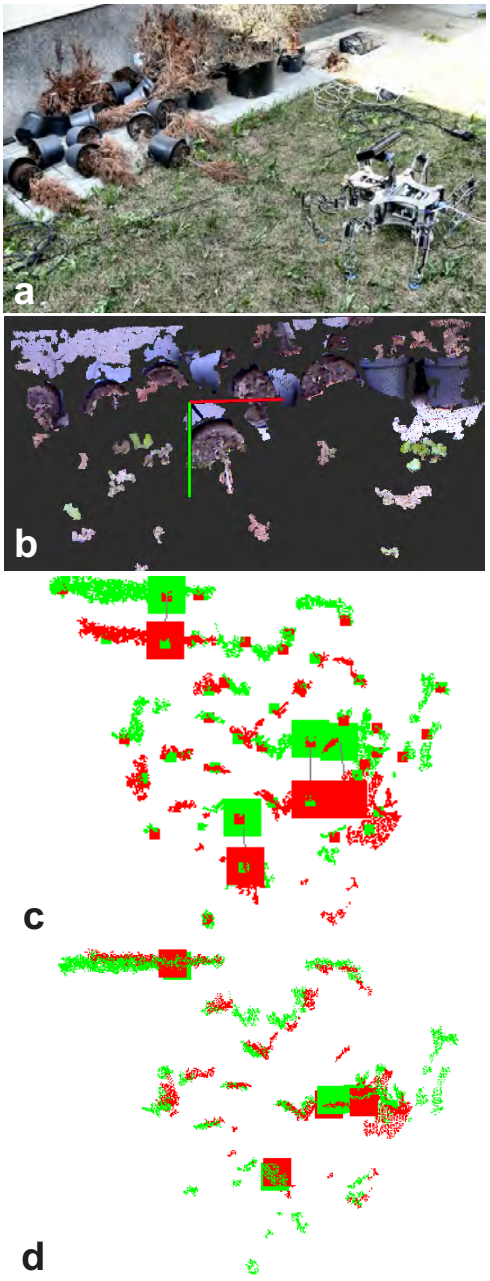

Fig. 9. Outdoor experimental setup (a), outdoor scene as seen by Kinect (b), two point clouds prior to matching with NARF correspondences (c), results of matching with NARF+ICP (d)

7. Conclusions

This paper presented a two-step range data registration method, which is suitable for the inexpensive Kinect 3D range sensor mounted on a walking robot.

The experiments have shown that the use of NARF descriptors matching brings the point clouds close to the correct position even for relatively large translations and rotations between the viewpoints. This procedure provides a good initial estimate of the egomotion, which is within the convergence basin of the ICP algorithm. In turn, the ICP algorithm allows precise alignment of the two point clouds, using all the range information. The combination of these two steps yields a robust estimate of the robot's egomotion in various environments, even under moderately varying lighting conditions.

Further studies will be carried out to accelerate the ICP implementation, and to fully exploit the combined RGB-D data. Using the range and photometric information the robot should be capable of self-localizing in both featureless areas (e.g. long corridors), and poorly illuminated areas. Implementation of a full scan-matching-based visual odometry system for the walking robot is also a matter of further research.

Notes

[1]Data downloaded from: http://vision.in.tum.de/data/datasets/rgbd-dataset

AUTHORS

Michał Nowicki – Poznań University of Technology, Institute of Control and Information Engineering, ul. Piotrowo 3A, 60-965 Poznań, Poland, e-mail: michalsminowicki@gmail.com.

Piotr Skrzypczyński* – Poznań University of Technology, Institute of Control and Information Engineering, ul. Piotrowo 3A, 60-965 Poznań, Poland, e-mail: piotr.skrzypczynski@put.poznan.pl.

*Corresponding author

REFERENCES

[1] K. S. Arun, T. S. Huang, S. Blostein, "Least-squares fitting of two 3-D point sets", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 9, no. 5, 1987, pp. 698–700.

[2] D. Belter, P. Skrzypczyński, "Rough terrain mapping and classification for foothold selection in a walking robot", *Journal of Field Robotics*, 28(4), 2011, pp. 497–528.

[3] D. Belter, P. Skrzypczyński, Precise self-localization of a walking robot on rough terrain using PTAM, in: *Adaptive Mobile Robotics* (N. Cowan *et al.*, eds.), Singapore, World Scientific 2012, pp. 89–96.

[4] P.J. Besl, N.D. McKay, "A method for registration of 3-D shapes", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 14, 2, 1992, pp. 239–256.

[5] S. Choi, T. Kim, W. Yu, "Performance evaluation of RANSAC family", In: *Proc. British Machine Vision Conference*, London, 2009.

[6] A. Davison, I. Reid, N. Molton, O. Stasse, "MonoSLAM: Real-time single camera SLAM",

IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 29, no. 6, 2007, pp. 1052–1067.

[7] D.W. Eggert, A. Lorusso, R.B. Fisher, "Estimating 3-D rigid body transformations: a comparison of four major algorithms", *Machine Vision and Applications*, vol. 9, no. 5–6, 1997, pp. 272-290.

[8] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, W. Burgard, "An evaluation of the RGB-D SLAM system", in: *Proc. IEEE Int. Conf. on Robot. and Automat.*, St. Paul, 2012, pp. 1691–1696.

[9] B. Gaßmann, J. M. Zöllner, R. Dillmann, "Navigation of walking robots: localisation by odometry". In: *Climbing and Walking Robots VIII*, Berlin, Springer 2005, pp. 953–960.

[10] P. Henry, M. Krainin, E. Herbst, X. Ren, D. Fox, "RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments", *Int. Journal of Robotics Research*, vol. 31, no.5, 2012, pp. 647–663.

[11] S. Izadi *et al.*, "KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera", *ACM Symp. on User Interface Software and Technology*, New York, 2011, pp. 559–568.

[12] W. Kabsch, "A solution of the best rotation to relate two sets of vectors", *Acta Crystallographica* 32:922, 1976.

[13] G. Klein, D. Murray, "Parallel tracking and mapping for small AR workspaces", In: *Proc. Int. Symp. on Mixed and Augmented Reality*, Nara, 2007, pp. 225–234.

[14] D.G. Lowe, "Distinctive image features from scale-invariant keypoints", *Int. Journal of Computer Vision*, vol. 60, 2, 2004, pp. 91–110.

[15] P. Łabęcki, P. Skrzypczyński, "Real-time visual perception for terrain mapping in a walking robot", In: *Adaptive Mobile Robotics* (N. Cowan *et al.*, eds.), Singapore, World Scientific 2012, pp. 754–761.

[16] M. Nowicki, P. Skrzypczyński, "Experimental verification of a walking robot self-localization system with the Kinect sensor", *Prace Naukowe Politechniki Warszawskiej – Elektronika. Postępy robotyki*, vol. 182, Warsaw, 2012, pp. 561–572 (in Polish).

[17] M. Nowicki, P. Skrzypczyński, Robust registration of Kinect range data for sensor motion estimation, in: *Computer Recognition Systems 5* (M. Kurzynski, M. Wozniak, eds.), Berlin, Springer-Verlag 2013. (in print)

[18] A. Nüchter, K. Lingemann, J. Hertzberg, H. Surmann, 6D SLAM – 3D mapping outdoor environments, *Journal of Field Robotics*, 24(8–9), 2007, pp. 699–722.

[19] A. Nüchter, S. Feyzabadi, D. Qiu, S. May, SLAM à la carte – GPGPU for globally consistent scan matching, in: *Proc. 5th European Conf. on Mobile Robots (ECMR)*, Örebro, 2011, pp. 271–276.

[20] L. Paz, P. Piniés, J. D. Tardós, J. Neira, Large-scale 6-DOF SLAM with stereo in hand, *IEEE Trans. on Robotics*, vol. 24, no. 5, 2008, pp. 946–957.

[21] PrimeSense, 2010, `http://www.primesense.com`

[22] S. Rusinkiewicz, M. Levoy, "Eficient variants of the ICP algorithm", in: *Proc. 3rd Int. Conf. on 3D Digital Imaging and Modeling*, Quebec, 2001, pp. 145–152.

[23] R. B. Rusu, N. Blodow, Z. Marton, M. Beetz, "Aligning point cloud views using persistent feature histograms". In: *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Nice, 2008, pp. 3384–3391.

[24] R. B. Rusu, S. Cousins, "3D is here: Point Cloud Library (PCL)". In: *Proc. IEEE Int. Conf. on Robot. and Automat.*, Shanghai, 2011, pp. 1–4.

[25] A. Segal, D. Haehnel, S. Thrun, "Generalized-ICP". In: *Proc. of Robotics: Science and Systems*, Seattle, 2009 (on-line).

[26] P. Skrzypczyński, "Simultaneous localization and mapping: a feature-based probabilistic approach", *Int. Journal of Applied Mathematics and Computer Science*, 19(4), 2009, pp. 575–588.

[27] P. Skrzypczyński, "Laser scan matching for self-localization of a walking robot in man-made environments", *Industrial Robot: An International Journal*, 39(3), 2012, pp. 242–250.

[28] O. Stasse, A. Davison, R. Sellaouti, K. Yokoi, "Real-time 3D SLAM for humanoid robot considering pattern generator information". "'n: *Proc. IEEE Int. Conf. on Intelligent Robots and Systems*, Beijing, 2006, pp. 348–355.

[29] B. Steder, R. B. Rusu, K. Konolige, W. Burgard, "Point feature extraction on 3D range scans taking into account object boundaries". In: *Proc. IEEE Int. Conf. on Robot. and Automat.*, Shanghai, 2011, pp. 2601–2608.

[30] A. Stelzer, H. Hirschmüller, M. Görner, "Stereo-vision-based navigation of a six-legged walking robot in unknown rough terrain", *Int. Journal of Robotics Research*, vol. 31, no. 4, 2012, pp. 381–402.

[31] T. Stoyanov, A. Louloudi, H. Andreasson, A. Lilienthal, Comparative evaluation of range sensor accuracy in indoor environments, in: *Proc. 5th European Conf. on Mobile Robots (ECMR)*, Örebro, 2011, pp. 19–24.

[32] K. Walas, D. Belter, Messor – Verstatile walking robot for search and rescue missions, *Journal of Automation, Mobile Robotics & Intelligent Systems*, vol. 5, no. 2, 2011, pp. 28–34.