A Set of Depth Sensor Processing ROS Tools for Wheeled Mobile Robot Navigation

Submitted: 8th January 2017; accepted: 18th April 2017

Michał Drwięga, Janusz Jakubiak

DOI: 10.14313/JAMRIS_2-2017/16

Abstract:
The paper presents a set of software tools dedicated to support mobile robot navigation. The tools are used to process an image from a depth sensor. They are implemented in ROS framework and they are compatible with standard ROS navigation packages. The software is released with an open source licence. First of the tools converts a 3D depth image to a 2D scan in polar coordinates. It provides projection of the obstacles, removes the ground plane from the image and compensates sensor tilt angle. The node is faster than the standard node within ROS and it has additional functions increasing range of possible applications. The second tool allows detection of negative obstacles i.e. located below the ground plane level. The third tool estimates height and orientation of the sensor with RANSAC algorithm applied to the depth image. The paper presents also the results of usage of the tools with mobile platforms equipped with Microsoft Kinect sensors. The platforms are elements of the ReMeDi project within which the software was developed.

Keywords: *RGB-D, Kinect, mobile robot, ROS, depth sensor, navigation tools*

1. Motivation

Autonomous navigation of mobile robots is a rapidly developing field. Every system used for this purpose requires information about obstacles in robot environment to plan a collision-free movement. The information about the obstacles may be collected from a variety of sensors. The most commonly used are laser scanners, due to their precision, reliability, range and measurement angles. There are however two factors that limit the usage of the laser scanners. One is their cost, the other – that the popular scanners from SICK and Hokuyo measure distance in a single plane. In real world applications, planar scanning is often insufficient as some obstacles do not manifest their presence in the measurement plane of the scanner. It happens when the objects are located above or below the scanning plane, but at the same time they are still at the height at which a robot may collide with them. In such case, a navigation based solely on the laser scanners cannot ensure a collision-free motion.

An important change in mobile robotics appeared in 2010 when Microsoft released its Kinect Xbox 360 with a PrimeSense depth sensor. With low cost and high availability, the device quickly gathered attention of numerous mobile robotics research groups. It has found many applications in robotics, like: object tracking and recognition, hand gesture processing, human action recognition, 3D mapping [15, 18]. The sensor found also its use in applications related to robot navigation: mapping, localization, obstacle detection [5, 11, 12, 20, 25, 34]. One sign of its popularity was IEEE/RSJ IROS conference in 2014, a part of which was "Kinect Robot Navigation Contest". Following the first Kinect, next depth sensors were introduced to the market and then adapted to robotic research. To these belong the second version of Kinect: Xbox One [13,22], ASUS Xtion, recently – DJI Guidance, ZED and RealSense [2].

A feature which makes the depth sensor particularly suitable for navigation is providing rich data about distances in the sensor view pyramid which allow 3D reconstruction of the scene. While in some works the authors assume using only depth sensors for navigation [5, 13, 25], some limitations of the sensors: measurement range, horizontal angle, parallax problem, cause that they cannot fully replace laser scanners [35]. However, with suitable combination of the data from both types of sensors, the depth sensors may be an excellent support to the laser scanners in obstacle detection for the purpose of robot navigation.

A factor that is important in robot sensory system design is that the volume of data produced by depth sensors is much bigger than that generated by laser scanners (e.g. Microsoft Kinect 360 may send over 9 million points per second, versus 6800 points per second from Hokuyo URG-04 laser scanner [5]). It results with much higher demands for computing resources: processor time and memory. However, a significant class of indoor mobile platforms are robots which are dedicated to motion on flat surfaces. In the case of such platforms, a full spatial image of their surroundings is not necessary. A common approach in that case is to use a planar, 2D map and a projection of data received from distance sensors to that map.

This paper presents a set of software tools developed within a ReMeDi project (*Remote Medical Diagnostician*) [1]. The tools are used to transform spatial data from the depth sensors to the planar form suitable for indoor robot navigation. They are implemented within ROS (Robot Operating System) [31] framework, which is currently frequently used in robotic community. A motivation to development of the tools was that available ROS packages were not fulfilling the requirements of the ReMeDi mobile platform. The main issue was the processing time which was too long for adequate reaction of the platform to obstacles met in environment. Moreover, a standard package demands horizontal orientation of the sensor, therefore not allowing optimization of range and dead zones by tilting the sensor. It also interprets visible ground as an obstacle. These three factors constraint a usage of the standard package in mobile robot applications. To overcome them, the first tool was designed: it provides fast processing of a depth image, providing information about distances to obstacles without ground interference and for any sensor tilt angle. The goal of the second tool is to detect in the depth image descents below the ground level, holes and ditches, which are called negative obstacles. The third of the tool set provides an estimate of the sensor location with respect to the ground which is necessary for the first two tools. The tool set is available as depth_nav_tools package [10] and it can be easily integrated with ROS navigation stack. The package has been released under open source BSD licence.

The paper contributes an approach to the depth image processing to detect both positive and negative obstacles. The presented methods of obstacle detection use mostly simple formulas from planar geometry. The simplicity of the used dependencies has allowed a development of a package with significantly shorter processing time than standard ROS packages used for that purpose. Processing time is shortened even though the package is enhanced with additional features like allowing sensor tilt and floor detection and removal from the image.

The paper is organized as follows: section 2 elaborates operation principles of each software tool in the depth_nav_tool package, section 3 presents mobile platforms which were used for evaluation and test results. The paper is summarized with conclusions.

2. Tool Description

2.1. Introduction

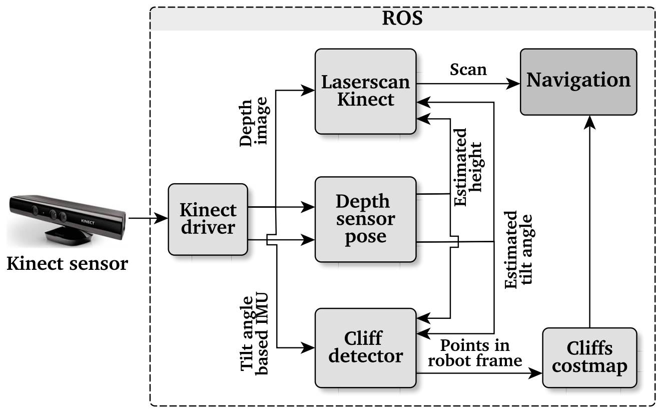

Fig. 1. A structure of connections between selected nodes of the control system

The implemented tools were build to be used for indoor navigation of mobile platforms. The first package, *laserscan_kinect*, converts spatial data of the depth image to a planar image, while preserving information which is important for the navigation system. The second tool, *cliff_detector*, allows detecting negative obstacles, like cliffs and descending stairs or holes and ditches in the floor. The last of the packages, *depth_sensor_pose*, estimates parameters of the sensor mount: elevation and inclination with respect to the ground. The parameters are determined on a basis of the ground plane detected in the depth image.

Fig. 1 presents a structure of connections between the presented packages running in a default configuration with a Kinect sensor. In this case the initial, rough tilt angle is based on the value measured by an accelerometer included in the Kinect sensor.

2.2. Depth Image to Planar Image Converter

The standard navigation package in ROS requires that the input data have a form of a laser scan. To allow usage of depth sensors for that purpose, the data must be transformed to a suitable format. There have been already ROS packages providing such conversion: *depthimage_to_laserscan* [30] and *pointcloud_to_laserscan* [8], however they were not meeting requirements of the ReMeDi project. In the first case, the depth map is converted to the planar form, but the package does not allow placement of the sensor close to ground. It is due to the fact that low mount of a sensor causes that the ground plane is detected as an obstacle and a distance to the floor is measured instead of to the real obstacles. It is also not possible to mount the sensor in tilted, non-horizontal position. In the case of point cloud based package the operation on point clouds demands too much computational and memory resources and its processing is too slow for platform navigation.

Those reasons were a motivation to develop a solution presented in this paper. It eliminates the drawbacks of the abovementioned packages: the ground is removed from the analyzed depth image and the sensor tilt angle is included in computations. Data conversion uses the method presented in [19] to obtain a transformation from the depth image to the ROS 2D laser scan format (*LaserScan* message). This message format contains a scan as a list of distances for consecutive scanning angles. The choice of the output format was determined by an assumption that the new package may be used as a replacement of *depthimage_to_laserscan*, enhancing its functionality and that it can be easily integrated with ROS 2D navigation module which uses LaserScan data format.

The input of all presented tools is a depth image with n rows and m columns and the image center located in (c_x, c_y) (see Fig. 2). A complete reading in a

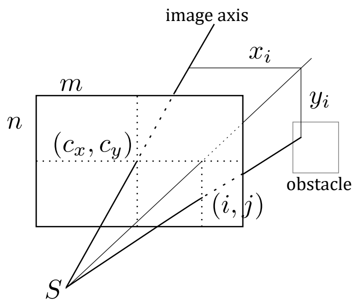

Fig. 2. Scene and depth image coordinates

given plane presents a projection of objects represen-

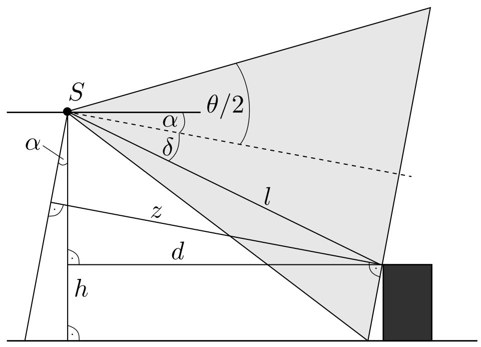

Fig. 3. Geometrical relationships in obstacle detection

ted as a distance to the closest object z_i and an offset x_i from the sensor optical center S. The conversion method is based on the assumption that the key information for the collision avoidance system is the location of the nearest obstacles. Therefore, while building the output image the lowest distance values are selected from each column of the image depth

$$z_i = min(z_{0,i}, z_{1,i}, \ldots, z_{n,i}), \tag{1}$$

where $z_{j,i}$ denotes a distance for ith column and jth row of the depth image. The solution assumes no lateral inclination of the sensor, which is a reasonable assumption in the case of robots navigating indoors, while reducing computation time. Field of view of the sensor can be tailored to application requirements by longitudinal tilt. To determine the offset relative to the sensor optical center the pinhole camera model was used

$$x_i = (i - c_x)\frac{1}{f_x}z_i, \tag{2}$$

where f_x is the focal length measured in pixels horizontally. Further considerations are made for the vertical plane containing the ith column of the image and the obstacle, as presented in Fig. 3. To simplify the notation we will skip in the equations the index denoting the column number. The angle to the obstacle in a vertical plane δ depends on number of the depth image row in which the obstacle is detected (j_{min}) according to

$$\delta = \theta\frac{j_{min} - c_y - \frac{1}{2}}{n-1}. \tag{3}$$

Taking into account the tilt angle of the sensor α, the distance to the obstacle is given by

$$d = l\sin(\frac{\pi}{2} - \alpha - \delta) = z\frac{\sin(\frac{\pi}{2} - \alpha - \delta)}{\sin(\frac{\pi}{2} - \delta)}. \tag{4}$$

If the sensor is heading below the horizontal line, there appears an additional problem of interpretation of the ground as an obstacle. The effect is related to the assumption of determining a distance to an obstacle as a lowest value in a given column of the depth image. To neutralize this effect, an algorithm of ground removal from the output data was implemented.

Detection and removal of the ground plane may base solely on the depth image. Such an aproach was

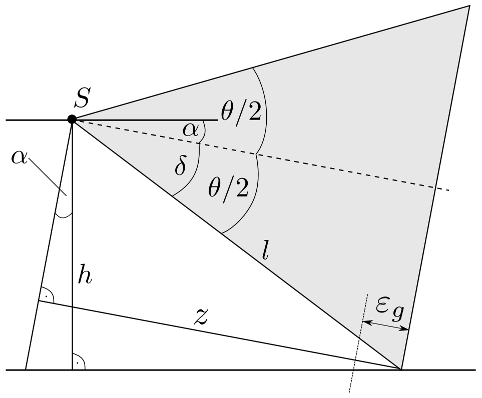

Fig. 4. Geometrical relationships in ground removal algorithm

used in combination with various techniques: direct fitting of depth readings to an exponential curve [21], Hough transformation [24] or clustering points with respect to directions of their difference vectors [26] or RANSAC [23]. With no additional information needed, the methods are general and may be used in various applications. However, a common problem of all methods is a correct discrimination of the ground plane from other flat surfaces appearing in the depth image. Usually it is solved with the aid of a user, by proper placement of the robot [23] or manual selection of points [27]. In [6] the authors propose a method where two of three points used for plane estimation are taken from fixed spots in the image, which are assumed to belong to the ground.

The ground removal method implemented in the *laserscan_kinect* package relies on a definition of a threshold value ε_g for each row of the image. The value corresponds to a theoretical distance to the ground for that row. Measured values higher than that are interpreted as ground points and they are ignored in further analysis of minimal distances in image columns. All points which are detected below given height are removed from the image. The computations require information on the tilt of the sensor α and the height h of the optical center of the sensor above the ground. Fig. 4 presents the geometrical relationships used in ground removal with $\delta = \theta/2$. The set of rejected points is defined as

$$P_g = \left\{(x, z) \mid z \geq h\frac{\sin\left(\frac{\pi}{2} - \delta\right)}{\cos\left(\frac{\pi}{2} - \delta - \alpha\right)} - \varepsilon_g\right\}. \tag{5}$$

Due to the accuracy of determining the angle of the sensor, its height and distance measurement errors, the measured values are classified as ground with a tolerance ε_g.

2.3. Detector of Negative Obstacles

The detector can detect negative obstacles, such as stairs down, faults or holes in ground (Fig. 5). The implemented detector package maps the obstacles to a dedicated layer in a costmap of the ROS navigation.

The detector requires for proper operation the information about the height at which the depth sensor is placed and the angle of longitudinal elevation to the ground. The detection of the obstacles is based on searching for the depth image points, for which the distance to the ground plane is exceeded with a tolerance of ε_g. The initial set of negative obstacles points is given by

$$P_c = \left\{ (x,z) \mid z \geq h \frac{\sin\left(\frac{\pi}{2} - \delta\right)}{\cos\left(\frac{\pi}{2} - \delta - \alpha\right)} + \varepsilon_g \right\}. \quad (6)$$

Due to various disturbances and to reduce the computational complexity of the algorithm, the depth image is divided into square blocks $b_{I,J}$ of the size b_s. For each of these blocks a number of points $n_{I,J}$ belonging to the negative obstacles is calculated,

$$n_{I,J} = \#\{P_c | i,j \in b_{I,J}\}. \quad (7)$$

If in a given block the number of points exceeds the threshold level n_{th}, then all points within that block are treated as negative obstacles. Hence the final set of negative obstacles points is

$$P_{obs} = \{b_{I,J} \mid n_{I,J} \geq n_{th}\}. \quad (8)$$

2.4. Sensor Elevation and Tilt Estimator

The Point Cloud Library (PCL) [27] provides a tool to estimate Kinect depth sensor elevation and tilt angle. However that tool relies on the points manually selected by user, who is required to select points from a floor in the image. In order to eliminate the need for manual selection of ground points, a third of the tools was created. Its aim is to automate the process of parameters estimation. The proposed method can estimate the tilt angle α and the elevation h over the ground. It is assumed that the sensor placement during the process is fixed. The method utilizes the RANSAC (RANdom SAmple Consensus) algorithm [9, 14]. The algorithm estimates parameters of the ground model on a basis of coordinates of points selected from the environment [7, 33].

To detect the ground plane a preliminary selection of points is made. To increase a chances that the selected points indeed belong to ground, they should be chosen from the lower part of the image and fitting between certain thresholds which indicate higher probability of the points being the ground. The thresholds are calculated from a given range of possible sensor mounting elevation $[h_{min}, h_{max}]$ and possible tilt angle range $[\alpha_{min}, \alpha_{max}]$. Based on those values the limits for distance measurements are determined. The

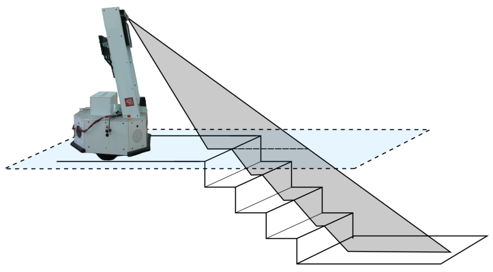

Fig. 5. A scheme of negative obstacles detection

method used is a modified method from section 2.2. The set of points which is transferred to the RANSAC algorithm to estimate the ground plane model parameters is given by

$$r_{min} = h_{min} \frac{\sin\left(\frac{\pi}{2} - \delta\right)}{\cos\left(\frac{\pi}{2} - \delta - \alpha_{max}\right)} \quad (9)$$

$$r_{max} = h_{max} \frac{\sin\left(\frac{\pi}{2} - \delta\right)}{\cos\left(\frac{\pi}{2} - \delta - \alpha_{min}\right)} \quad (10)$$

$$P = \{(x,y,z) \mid z \in [r_{min},\, r_{max}]\} \quad (11)$$

Coordinates of a point in space based on the $z_{i,j}$ value of the (i,j) point of the depth image are calculated as follows [19]

$$(x_{i,j}, y_{i,j}) = \left((i - c_x)\frac{1}{f_x} z_{i,j},\; (j - c_y)\frac{1}{f_y} z_{i,j} \right). \quad (12)$$

The ground model parameters estimated with the RANSAC algorithm has a form of a general plane equation

$$\widehat{\alpha} = \arccos\left(\frac{\vec{n} \circ \vec{m}}{|\vec{n}| \cdot |\vec{m}|} \right), \quad (13)$$

$$\widehat{h} = \frac{|D|}{\sqrt{A^2 + B^2 + C^2}}, \quad (14)$$

where: $\vec{n} = [A, B, C]$ i $\vec{m} = [A, B, 0]$. Based on the plane equation, the tilt angle and the height of the sensor placement are determined. The resulting parameter values are meant to be used by the tools from the previous sections to compensate sensor tilt angle and ground removal.

3. Verification

The ROS nodes included in the depth_nav_tools package were tested with mobile platforms of the ReMeDi project. The goal of the project [3, 17, 28] is to build a robotic system for remote medical examination. A mobile platform is the element of the system used to carry medical diagnostic equipment. To simplify operation of the platform by the medical personnel in the hospital, the mobile platform is equipped with an autonomous navigation system. A part of the system is an obstacle detection module, which employs several sensors, including Kinect sensors.

3.1. Sensor Placement

During a selection of mount place for the Kinect sensor on the robot it is necessary to take into account sensor dead zones. One can distinguish central, lower and upper dead zones, related respectively to: sensor minimal detection range, located below and above sensor field of view. Objects located in these areas are not visible to the sensor.

Fig. 6 illustrates an exemplary length of the lower dead zone for Kinect sensor with respect to the value of the tilt angle. In the general case, the size of the lower dead zone d_m is determined by

$$d_m = (h - h_1) \tan\left(\frac{\pi}{2} - \frac{\theta}{2} - \alpha \right), \quad (15)$$

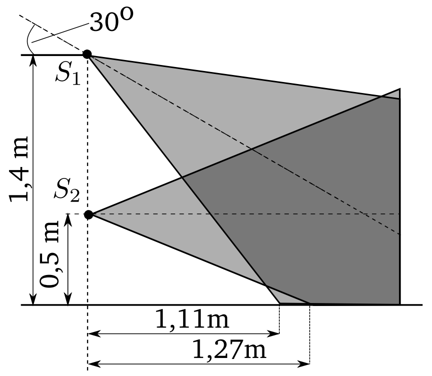

Fig. 6. Influence of Kinect sensor elevation and tilt angle on the lower dead zone length

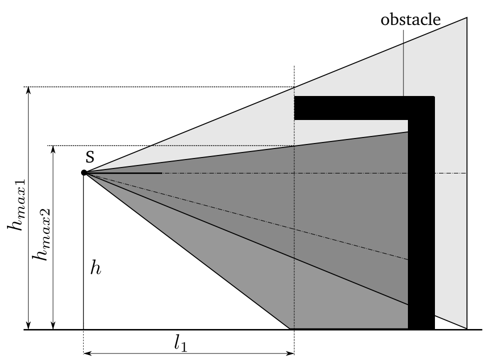

Fig. 7. Upper dead zone of the Kinect sensor

where h_1 is a distance from the ground for which the size of the dead-zone is determined ($0 \leq h_1 \leq h$). To reduce the length of the lower dead zone in front of the robot, the sensor should be raised high and tilted down.

The sensor placement should also take into consideration the upper dead zone located above the field of view of the sensor. This is particularly important in the case of such obstacles as tables, chairs or hospital beds. When the sensor is inclined downwards or the sensor is placed too low, there is a risk that such objects are not properly detected (Fig. 7). The maximum height at which the sensor still detects objects located in given distance is calculated as

$$h_{max} = h + sgn\left(\frac{\theta}{2} - \alpha\right) l_1 \tan\left(\left|\frac{\theta}{2} - \alpha\right|\right), \quad (16)$$

where l_1 is a distance from the object to a vertical line passing through the sensor optical center. Ideally the sensor should be tilted to such an angle that the maximum height of detected obstacles is bigger that the height of the robot.

For some robots it is possible to completely eliminate influence of the sensor dead zones by using carefully selected configuration of the sensor. However it will be usually associated with reduction of the maximum distance of obstacle detection. A solution which allows to ensure detection of obstacles in wider range of angles is to employ higher number of sensors so that their fields of view do not overlap and do not cause interference with one another. Some results of research on interference of multiple Kinect sensors were presented in [4, 29].

3.2. Processing Time Comparison

The first step of verification of the *laserscan_kinect* package was comparison of processing time with *depthimage_to_laserscan* package. The two packages convert depth images to 2D laser scanner format. The Table 1 shows results of an experiment for Kinect 360 sensor and depth image of 640×480 pixels. It presents time measured from publication of the depth image to reception of the processed scan (t_1) and time of processing only (denoted t_2). In case of *laserscan_kinect* package processing lengths were measured both with additional features (ground removal, tilt compensation) enabled and without them. The computer used in tests was equipped with *Intel i7-5500U* and 16 GB RAM.

Tab. 1. Depth images conversion time comparison

Package	Threads	Features	t_1 [ms]	t_2 [ms]
laserscan kinect	1	off	3.10	0.53
		on	3.77	0.72
	2	off	2.64	0.36
		on	2.96	0.55
depthimage to laserscan	1	N/A	18.91	N/A

On the same machine the package was tested with a newer version of sensor *Kinect v2 Xbox One*. In that case *libfreenect2* driver and software bridge *iai_kinect2* [32] between driver and ROS framework were used. The conversion time of depth image (1920×1080) was about 85 ms, but it should be noted that the depth image was generated with CPU (no GPU present), what significantly affects the depth image conversion time.

3.3. Tests with Mobile Platforms

The package was tested with two mobile platforms developed in the ReMeDi project: a testbed platform Carol and the ReMeDi platform prototype (Fig. 13).

The platforms were equipped with Kinect 360 depth sensor and Hokuyo URG-04LX laser scanner. Robot control systems were operating in ROS Indigo under Ubuntu 14.04 LTS. The details of the platform setup and configuration of the navigation systems for the two platforms were presented in [16, 17].

The first functionality that has been verified is the removal of the ground from the depth image. The result obtained from the package is shown in Fig. 9. One can observe that without ground removal feature enabled a floor in front of the robot is interpreted as an obstacle. After enabling the ground removal function. the obstacles are localized correctly in similar locations as detected by the laser scanner.

The next step was verification whether the *laserscan_kinect* detects obstacles invisible to the laser scanner and causes modification of the robot path. Fig. 10

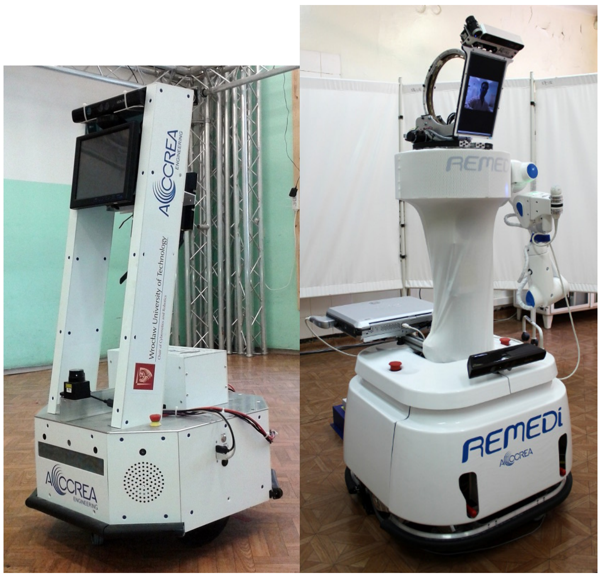

Fig. 8. Testbed platform Carol and ReMeDi robot prototype

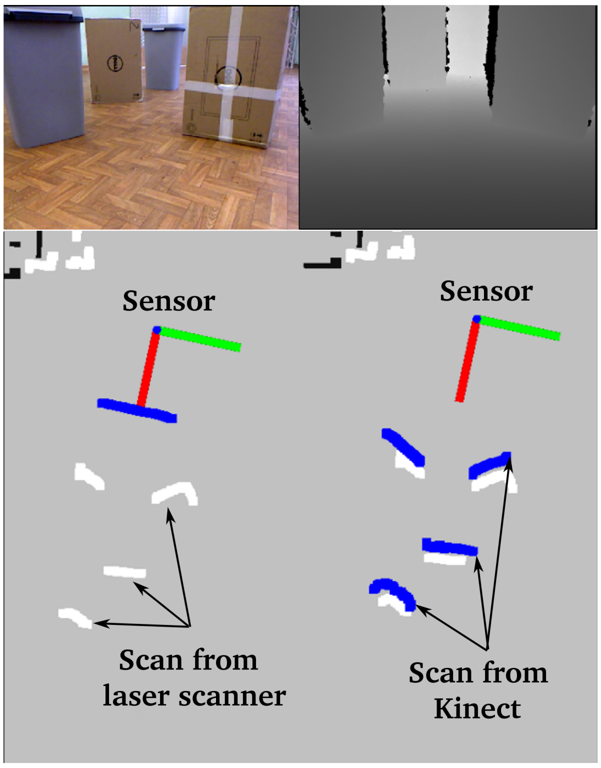

Fig. 9. Obstacle detection example: RGB and depth image, without (bottom left) and with (bottom right) ground removal enabled. Blue lines indicate where the obstacles were detected. For reference, white lines are readings from the laser scanner

shows the results from the autonomous navigation system. A setup included a table, the top of which was located above the laser scanner plane. The destination point was selected in such a way that initially planned path was passing close to a table. Without the data from the depth sensor, the planned path would cause a collision of the robot with the table (bottom-left of Fig. 10). The bottom-right part shows that after inclu-

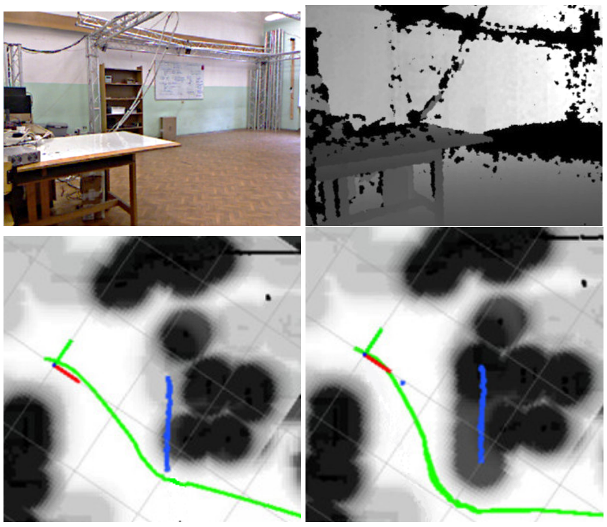

Fig. 10. Laserscan_kinect in robot navigation: RGB and depth images of a scene, navigation costmap without and with depth sensor layer

ding the data from the *laserscan_kinect*, the table top becomes visible to the navigation system and the robot avoids a collision with it.

Separate tests were conducted to verify the performance of the *laserscan_kinect* for various configurations and threshold value ε_g. The results are presented in Figs. 11 and 12. In the scene there were placed four boxes of various heights: 40 cm, 15 cm, 3 cm, 1 cm. The sensor was placed at 4 poses, differing with elevation and tilt angle: $(0.45m, 15^o)$, $(0.8m, 40^o)$, $(1.2m, 50^o)$, $(1.2m, 60^o)$. The value of ground threshold ε_g was from 0cm to 20cm for each of the sensor pose. It can be noticed that the increase of the ε_g value affects on range of obstacles height which are detected and boxes consequently disappears from the output scan. It is may also be observed that at the higher tilt angle less obstacles are visible. Partially the observed effect is expected, as according to (5), higher values of ε_g and δ cause increase the margin where objects are treated as the floor. The practical application would require to tune the ε_g value with respect to the placement of the sensor at the robot and expected heights of the obstacles to be avoided.

Further experiments were related to the negative obstacles detector. Fig. 13 presents an image of a scene with descending stairs. The image blocks recognized as negative obstacles are indicated in the depth image with yellow dots. The same detected blocks are also marked as an obstacle on a layer of a costmap. In the course of tests Kinect sensor was mounted on a platform at a height of approximately 0.5m and it was tilted $15°$ down from horizontal.

4. Conclusions

The paper presents a set of tools developed to extend possible usage of depth sensors in autonomous navigation system of mobile platforms. The *laserscan_kinect* package allows to use a depth sensor with the ROS navigation packages which require

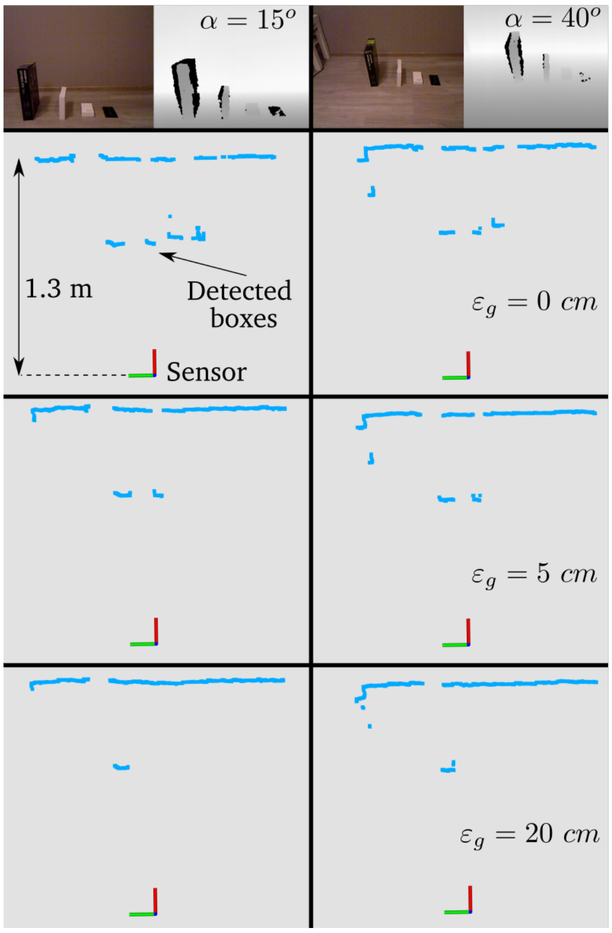

Fig. 11. Obstacle detection for various ground margin values ε_g and sensor poses: left column $h = 0.45\ m$, $\alpha = 15^o$, right column $h = 0.8\ m$, $\alpha = 40^o$

the *LaserScan* message. The presented package is a replacement in such applications for the *depthimage_to_laserscan*. It extends its functionality by allowing tuning sensor view area by modification of its tilt angle and elimination of the ground from processing. Additionally, the package significantly reduces the processing time (5 to 6 times for Kinect 360). The *cliff_detector* allows introduction of the elements below the ground level to the costmap used in robot path planning. This reduces vulnerability of the navigation systems to such elements like descending stairs or holes and ditches in a floor. The last of the tools is *depth_sensor_pose* which plays a supplementary role for the first two. It provides automatic estimation of the parameters necessary for *laserscan_kinect* and *cliff_detector*. The software was published with BSD open source licence and it is available to ROS community.

The tools were verified within the ReMeDi project with two platforms used in the project. The system was able to detect obstacles correctly, taking into account the constraints related mainly to angles and sensors dead zones. The experiments confirmed that the detection of negative obstacles worked correctly. Information about the obstacles obtained from the detector can be efficiently used in robot path planning to avoid dangerous areas or to decelerate near

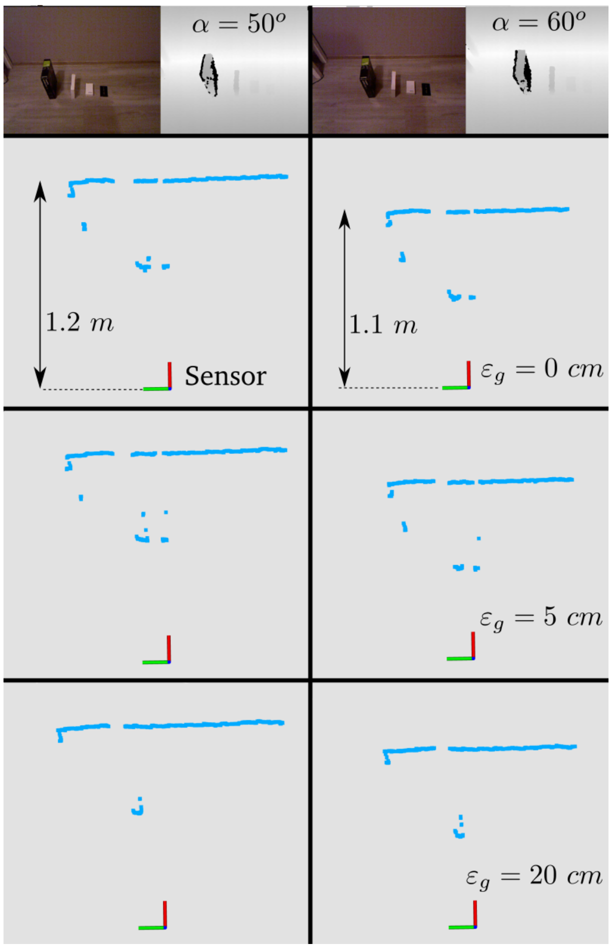

Fig. 12. Obstacle detection for various ground margin values ε_g and sensor poses: left column $h = 1.2\ m$, $\alpha = 50^o$, right column $h = 1.2\ m$, $\alpha = 60^o$

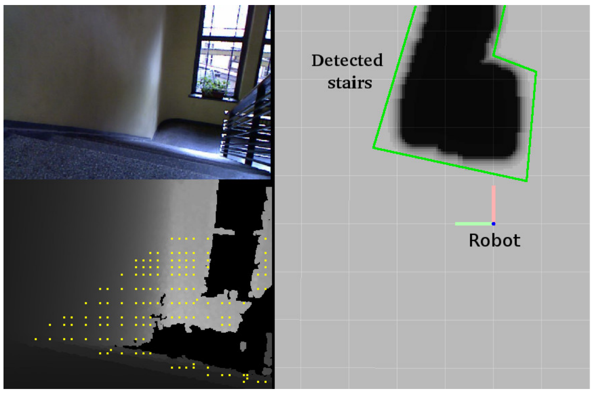

Fig. 13. Example of descending stairs detection: RGB image, depth image with negative obstacles regions marked with yellow dots, a costmap layer

them. It must however be mentioned, that the Kinect sensor itself is not hard real-time unit, therefore the *cliff_detector* tool should not be the used as a safety critical system. The motion safety should be provided by additional sensors operating in real-time mode.

ACKNOWLEDGEMENTS

This work was supported by the European Commission FP7 ICT-610902 project ReMeDi (Remote Medical Diagnostician) and co-financed by Polish Ministry of Science and Higher Education

AUTHORS

Michał Drwięga – Department of Cybernetics and Robotics, Faculty of Electronics, Wrocław University of Technology, ul. Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland, e-mail: michal.drwiega@pwr.edu.pl.
Janusz Jakubiak[*] – Department of Cybernetics and Robotics, Faculty of Electronics, Wrocław University of Technology, ul. Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland, e-mail: janusz.jakubiak@pwr.edu.pl.

[*]Corresponding author

REFERENCES

[1] "ReMeDi EU Project". http://www.remedi-project.eu.

[2] Y. ai Bi, J. Li, H. Qin, M. Lan, M. Shan, F. Lin, B. M. Chen, "An MAV localization and mapping system based on dual Realsense cameras". In: P. Z. PENG, D. F. LIN, eds., *International Micro Air Vechicle Competition and Conference 2016*, Beijing, PR of China, 2016, 50–55.

[3] K. Arent, J. Jakubiak, M. Drwięga, M. Cholewiński, G. Stollnberger, M. Giuliani, M. Tscheligi, D. Szczęśniak-Stańczyk, M. Janowski, W. Brzozowski, A. Wysokiński, "Control of mobile robot for remote medical examination: Design concepts and users' feedback from experimental studies". In: *2016 9th International Conference on Human System Interactions (HSI)*, 2016, 76–82. DOI: 10.1109/HSI.2016.7529612.

[4] K. Berger. *A State of the Art Report on Multiple RGB-D Sensor Research and on Publicly Available RGB-D Datasets*, 27–44. Springer International Publishing, Cham, 2014.

[5] J. Biswas, M. Veloso, "Depth camera based indoor mobile robot localization and navigation". In: *Proceedings of IEEE International Conference on Robotics and Automation*, 2012, 1697–1702.

[6] K. Bohlmann, A. Beck-Greinwald, S. Buck, H. Marks, A. Zell, "Autonomous person following with 3D LIDAR in outdoor environment", *Journal of Automation, Mobile Robotics and Intelligent Systems*, vol. 7, no. 2, 2013, 24–29.

[7] D. Borrmann, J. Elseberg, K. Lingemann, A. Nüchter, "The 3D Hough transform for plane detection in point clouds: A review and a new accumulator design", *3D Res.*, 2011, 32:1–32:13. DOI: 10.1007/3DRes.02(2011)3.

[8] P. Bovbel, T. Foote. "ROS package pointcloud_to_laserscan". http://wiki.ros.org/pointcloud_to_laserscan.

[9] S. Choi, T. Kim, W. Yu, "Performance Evaluation of RANSAC Family". In: *Proceedings of the British Machine Vision Conference*, 2009, 81.1–81.12. DOI: 10.5244/C.23.81.

[10] M. Drwięga. "Navigation tools". http://wiki.ros.org/depth_nav_tools.

[11] F. Endres, J. Hess, J. Sturm, D. Cremers, W. Burgard, "3-d mapping with an rgb-d camera", *IEEE Transactions on Robotics*, vol. 30, no. 1, 2014, 177–187. DOI: 10.1109/TRO.2013.2279412.

[12] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, W. Burgard, "An evaluation of the RGB-D SLAM system". In: *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 2012, 1691–1696.

[13] P. Fankhauser, M. Bloesch, D. Rodriguez, R. Kaestner, M. Hutter, R. Siegwart, "Kinect v2 for mobile robot navigation: Evaluation and modeling". In: *2015 International Conference on Advanced Robotics (ICAR)*, 2015, 388–394. DOI: 10.1109/ICAR.2015.7251485.

[14] M. Fischler, R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography", *Commun. ACM*, vol. 24, no. 6, 1981, 381–395. DOI: 10.1145/358669.358692.

[15] J. Han, L. Shao, D. Xu, J. Shotton, "Enhanced computer vision with Microsoft Kinect sensor: A review", *IEEE Transactions on Cybernetics*, vol. 43, no. 5, 2013, 1318–1334. DOI: 10.1109/TCYB.2013.2265378.

[16] J. Jakubiak, M. Drwięga, A. Kurnicki, "Development of a mobile platform for a remote medical teleoperation robot". In: *Proc 21st Int Conf Methods and Models in Automation and Robotics (MMAR)*, 2016, 1137–1142. DOI: 10.1109/MMAR.2016.7575298.

[17] J. Jakubiak, M. Drwięga, B. Stańczyk, "Control and perception system for ReMeDi robot mobile platform". In: *Proc 20th Int Conf Methods and Models in Automation and Robotics (MMAR)*, 2015.

[18] A. Kadambi, A. Bhandari, R. Raskar. *"Computer Vision and Machine Learning with RGB-D Sensors"*, chapter "3D Depth Cameras in Vision: Benefits and Limitations of the Hardware". Springer, 2014.

[19] K. Kamarudin, S. Mamduh, A. Shakaff, S. Saad, A. Zakaria, A. Abdullah, L. Kamarudin, "Method to convert Kinect's 3D depth data to a 2D map for indoor SLAM". In: *IEEE 9th Int Coll Signal Processing and its Applications (CSPA)*, 2013, 247–251.

[20] K. Kamarudin, S. Mamduh, A. Shakaff, A. Zakaria, "Performance analysis of the Microsoft Kinect sensor for 2D simultaneous localization and mapping (SLAM) techniques", *Sensors*, vol. 14, no. 12, 2014, 23365–23387. DOI: 10.3390/s141223365.

[21] D. Kırcalı, F. B. Tek, "Ground plane detection using an RGB-D sensor". In: L. R. Czachórski T., Gelenbe E., ed., *Information Sciences and Systems 2014*, Cham, 2014. DOI: 10.1007/978-3-319-09465-6_8.

[22] E. Lachat, H. Macher, M.-A. Mittet, T. Landes, P. Grussenmeyer, "First Experiences with Kinect v2 Sensor for Close Range 3d Modelling", *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2015, 93–100. DOI: 10.5194/isprsarchives-XL-5-W4-93-2015.

[23] P. Łabęcki, D. Belter, "System calibration method for a walking robot", *Journal of Automation, Mobile Robotics and Intelligent Systems*, vol. 7, no. 2, 2013, 39–45.

[24] K. Okada, S. Kagami, M. Inaba, H. Inoue, "Plane segment finder: algorithm, implementation and applications". In: *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 2, 2001, 2120–2125. DOI: 10.1109/ROBOT.2001.932920.

[25] A. Oliver, S. Kang, B. Wünsche, B. MacDonald, "Using the kinect as a navigation sensor for mobile robotics". In: *Proceedings of the 27th Conference on Image and Vision Computing New Zealand*, 2012, 509–514. DOI: 10.1145/2425836.2425932.

[26] S. Oßwald, J. S. Gutmann, A. Hornung, M. Bennewitz, "From 3D point clouds to climbing stairs: A comparison of plane segmentation approaches for humanoids". In: *11th IEEE-RAS Int Conf Humanoid Robots*, 2011, 93–98. DOI: 10.1109/Humanoids.2011.6100836.

[27] PCL. "Point Cloud Library". http://pointclouds.org.

[28] A. Peer, M. Buss, B. Stańczyk, D. Szczęśniak-Stańczyk, W. Brzozowski, A. Wysokiński, M. Tscheligi, C. A. Avizzano, E. Ruffaldi, L. van Gool, A. Fossati, K. Arent, J. Jakubiak, M. Janiak, "Towards a remote medical diagnostican for medical examination". In: *NextMed MMVR21*, 2014.

[29] N. Rafibakhsh, J. Gong, M. K. Siddiqui, C. Gordon, H. F. Lee, "Analysis of xbox kinect sensor data for use on construction sites: Depth accuracy and sensor interference assessment". In: *Construction Research Congress*, 2012. DOI: 10.1061/9780784412329.086.

[30] C. Rockey. "ROS package depthimage_to_laserscan". http://wiki.ros.org/depthimage_to_laserscan.

[31] ROS. "Robot Operating System". http://www.ros.org.

[32] T. Wiedemeyer. "IAI Kinect2". https://github.com/code-iai/iai_kinect2, 2014 – 2015.

[33] M. Y. Yang, W. Förstner, "Plane detection in point cloud data". In: *Department of Photogrammetry, University of Bonn*, 2010.

[34] Z. Zhang, "Microsoft Kinect Sensor and Its Effect", *IEEE MultiMedia*, 2012, 4–10.

[35] S. Zug, F. Penzlin, A. Dietrich, T. T. Nguyen, S. Albert, "Are laser scanners replaceable by kinect sensors in robotic applications?". In: *Robotic and Sensors Environments (ROSE), 2012 IEEE International Symposium on*, 2012, 144–149. DOI: 10.1109/ROSE.2012.6402619.