Localization and Navigation of an Assistive Humanoid Robot in a Smart Environment

Enric Cervera
Robotic Intelligence Lab
Jaume-I University
Castello-Spain
ecervera@icc.uji.es

Amine Abou Moughlbay
Institut de Recherche en Communications
et Cybernétique de Nantes (IRCCyN)
Ecole Centrale de Nantes - France
Amine.abou-moughlbay@ec-nantes.fr

Philippe Martinet
IRCCyN - Ecole Centrale de Nantes
Institut Pascal - Clermont Ferrand
France
Philippe.Martinet@irccyn.ec-nantes.fr

***Abstract*—Assistive humanoids that manipulate objects in everyday environments are potentially useful to improve the lives of the elderly or disabled. To retrieve or deliver objects at home, precise localization is needed. But localization of humanoid robots is a challenging issue, due to rough odometry estimation, noisy onboard sensing, and the swaying motion caused by walking. To overcome these limitations, we advocate for the use of external sensors for localization and navigation in a smart home environment. As opposed to a stand-alone self contained robot, our humanoid benefits from the information coming from other sensing devices in the environment. In order to achieve robust localization while walking, and retrieve an object from the floor, we use RGBD camera information from an external Kinect sensor. Monte Carlo localization estimates the 6D torso pose estimation of the humanoid, which is then used for closed-loop navigation control. Experiments with a NAO humanoid point out that, by cooperating with the environmental sensors, the overall precision of robot navigation is dramatically improved.**

I. Introduction

Object retrieval is remarked as a high priority task for assistive robots by people with physical disabilities. Humanoid robots could potentially help people with motor impairments to retrieve dropped objects [1], [2]. But robust and precise robot localization is a prerequisite for such task, and humanoid localization remains a challenging issue due to inaccurate foot step odometry and noisy onboard sensor observations during walking [3].

Since our humanoid is targeted to human-friendly indoor environments, a smart home endowed with networked sensors would provide additional useful information to monitor both humans and robots. In smart environments, components are working together by exchanging information via the local network. The main idea behind the smart home concept is to use networked robots to integrate different services within the home as a means to control and monitor the entire living space [4]. Such services are not realized by a single full-equipped robot but by a combination of different elements such as environmental sensors, cameras and human communicating and cooperating through the network.

The Kinect is a powerful low-cost sensor which provides color and range images, suitable for human motion detection and tracking. Dingli et al. [5] have created an Ambient-Assisted Living application which monitors a person's position, labels objects around a room and raises alerts in case of falls. Stone and Skubic [6] have investigated this sensor for in-home fall risk assessment. Ni et al. [7] use color-depth fusion schemes for feature representation in human action recognition.

In this paper, we present a localization and navigation method for a humanoid robot in an indoor smart environment. The Kinect sensor is used to monitor and track the 6D pose of the robot. Localization and navigation towards a detected object can then be achieved precisely. The rest of the paper is organized as follows: related work on humanoid localization is discussed in Section II; Section III presents the architecture of the system; localization and control of the humanoid is described in Section IV; experimental results are presented in Section V; finally, Section VI draws some conclusions and outlines future work extensions.

II. Related work

Accurate localization, which is considered to be mainly solved for wheeled robots, is still a challenging problem for humanoid robots [3]. When dealing with biped robots, many problems arise such as foot slippage, stability problems during walking, and limited payload capabilities, preventing precise localization in their environment. In addition, humanoids usually cannot be assumed to move on a plane to which their sensors are parallel due to their walking motion.

In the last few years, Monte Carlo methods have been commonly used to perform localization on mobile robots [8], as well as other methods including grid-based Markov localization and Kalman filtering [9]. Furthermore, many studies have been made to solve the humanoid localization problem by tracking their pose in the two-dimensional space. Ido et al. [10] applied a vision-based approach and compare the current image to previously recorded reference images in order to estimate the location of the robot. Owald et al. [11] and Bennewitz et al. [12] compared visual features to a previously learned 2D feature map during pose tracking. Pretto et al. [13] tracked visual features over time for estimating the robot's odometry. Cupec et al. [14] detected objects with given shapes and colors in the local environment of the humanoid and determine its pose relative to these objects.

In addition to that, many techniques using laser range data have also been developed. Stachniss et al. [15] presented an approach to learn accurate 2D grid maps of large environments with a humanoid equipped with a Hokuyo laser scanner. Such a map was subsequently used by Faber et al. [16] for humanoid localization in 2D. Similarly, Tellez et al. [17] developed a navigation system for such a 2D environment representation using two laser scanners located in the feet of the robot.

Since a 2D map is often not sufficient for humanoid motion planning, several methods use 2.5D grid maps which additionally store a height value for each cell. Thompson et al. [18] track the 6D pose of a humanoid equipped with a laser scanner in such a representation. Hornung et al. [3] track a humanoid's 6D pose in a 3D world model, which may contain multiple levels connected by staircases.

In the cited methods, they used either embedded cameras and sensors on the robot or exteroceptive ones which are mounted on the robot's head or body. All these sensors are usually used to track the environment and subsequently localize the robot.

In our method, the sensors are fixed and used to localize the walking robot. The contribution of this paper is a robust localization system for humanoid robots navigating in indoor environments using Kinect cameras. The main goal is to develop a robust system which is able to track and localize the robot when walking, and control its motion precisely enough, to be able to retrieve an object from the floor.

III. System's Architecture

The system is composed of a small humanoid robot NAO, navigating in an indoor smart environment, where a 3D vision system, consisting of one or more low cost Kinect cameras, monitors and tracks both the user and robot activity, as depicted in Fig. 1.

In our current implementation, the system is able to detect small objects lying on the floor plane, as well as to localize the robot. In the future, we will incorporate the skeletal tracking of the Kinect sensor, to be able to interface directly with human gestures. This section briefly describes the hardware components along with the system software framework.

A. Kinect sensor

The used vision system is the Kinect camera, which consists of two optical sensors whose interaction allows a three-dimensional scene analysis. One of the sensors is an RGB camera which has a video resolution of 30 fps. The image resolution given by this camera is 640x480 pixels. The second sensor has the aim of obtaining depth information corresponding to the objects found at the scene. The working principle of this sensor is based on the emission of an infrared signal which is reflected by the objects and captured by a monochrome CMOS sensor. A matrix is then obtained which provides a depth image of the objects in the scene, called DEPTH. An investigation of the geometric quality of depth data obtained by the Kinect sensor was done by [19], revealing that the point cloud does not contain large systematic errors when compared with a laser scanning data.

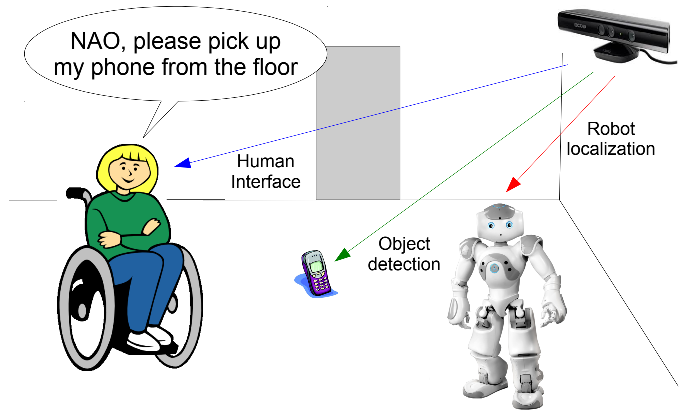


Fig. 1. Overview of the system: the Kinect sensor in the smart environment is able to monitor the user activity, detect objects on the floor, and localize precisely the robot.

B. NAO Robot

The NAO Robot [20], developed by Aldebaran robotics, is a biped robot with 25 Degrees of Freedom (DOF). It has 3-fingered robotic hands used for grasping and holding small objects (it can carry up to 300g using both hands). It is equipped with: 2 ultrasound devices situated in the chest that provide space information in a range of 1 meter, 2 cameras situated on the top and bottom of the head, 2 bumpers (contact sensors on the robot's feet), a gyrometer and an accelerometer (to determine whether the robot is in a stable or unstable position).

C. Software overview

The Point Cloud Library (PCL) is a standalone, large scale, open project for 3D point cloud processing. [21]. The PCL framework contains numerous state-of-the art algorithms including filtering, feature estimation, surface reconstruction, registration, model fitting and segmentation, as well as higher level tools for performing mapping and object recognition.

The tracking module of PCL [22] provides a comprehensive algorithmic base for the estimation of 3D object poses using Monte Carlo sampling techniques and for calculating the likelihood using combined weighted metrics for hyper-dimensional spaces including Cartesian data, colors, and surface normals.

ROS (Robot Operating System) is a software framework which provides libraries and tools to help developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more [23].

With its modular design, ROS makes it easy to develop network robot systems. Seamless interfacing with the Kinect sensor, NAO robot, and PCL is provided.

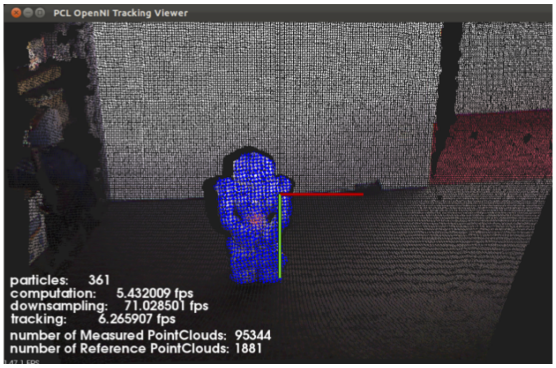


Fig. 2. 3D tracking of the NAO robot with Kinect.

IV. Localization and navigation

A. Localization

This technique consists of tracking 3D objects (position and orientation) in continuous point cloud data sequences. It was originally designed for robots to monitor the environment and make decisions and adapt their motions according to the changes in the world, but it is equally suitable for tracking the robot while it walks around the environment (Fig. 2). Localization has been optimized to perform computations in real-time, by employing multi CPU cores optimization, adaptive particle filtering (KLD sampling) and other modern techniques [22].

In our application, we use a rigid model of the torso and head parts of the robot, to track the system on real-time and find the 3D pose of the robot model even when walking. In future works, an articulated model could be used to track the whole body of the robot.

Using the PCL Cloud tracking technique we can find the actual pose of the robot and the desired one (the object) with respect to the Kinect. Thus the relative 3D pose can be calculated, and we can control the direction of walking in the plane: the position in X and Y directions and the orientation of the robot.

B. Navigation

The NAO robot is able to walk in velocity control mode. This enables the walk to be controlled reactively, as the most recent command overrides all previous commands. However, the walk uses a preview controller to guarantee stability. This uses a preview of time of 0.8s, so the walk will take this time to react to new commands. At maximum frequency this equates to about two steps [24].

The linear velocity of the robot $({}^cV_x, {}^cV_y)^T$ is calculated with respect to the pose error between actual and desired poses $(e_x, e_y, e_\theta)^T$ using a proportional gain λ. The aim is to move the robot linearly to the target.

$$\begin{pmatrix} {}^cV_x \\ {}^cV_y \end{pmatrix} = \lambda \begin{pmatrix} e_x \\ e_y \end{pmatrix} \tag{1}$$

The angular velocity of the robot ${}^c\omega$ may be calculated in two different manners: first, while the error is greater than a given threshold e_{min}, the robot will head towards the line joining its current location and final destination; second, when the error is lower than such threshold, the robot will head towards its final orientation.

$${}^c\omega = \begin{cases} \lambda \arctan(e_y/e_x) & \text{if } \sqrt{e_x^2 + e_y^2} > e_{min} \\ \lambda e_\theta & \text{otherwise} \end{cases} \tag{2}$$

C. Veering correction

Closed-loop control is able to converge in the presence of uncertainties in sensors and actuators. However, faster convergence is achieved if the system is properly calibrated. In a similar way to human beings [25], biped robots suffer from inability to maintain a straight path when walking without vision: slight differences between each leg stride caused by backslash, friction, or motor power will produce a veering behavior which can be estimated and corrected.

A simple veering correction procedure is now introduced: the robot is commanded in open-loop to walk straight away with a constant linear velocity. Its trajectory is recorded with the Kinect sensor, and two fitting steps are performed:

1) Circle fitting: the best fitting circle is computed, as shown in Fig. 3a.
2) Line fitting: the robot is now commanded to walk with a constant linear velocity, and a constant angular velocity, as computed from the previous fitting step. Fig. 3b depicts the recorded trajectory, and the LMS line fitting.

The error is significantly reduced: without correction, for a 1m walked distance, the lateral error grows to 35cm, and the orientation error is 30 (Fig. 3a). With angular correction (Fig. 3b), the orientation error is not noticeable, but a lateral error persists, 25cm for a 1.5m walked distance. This error is compensated an order of magnitude with a lateral velocity term, resulting in only 3cm for the same walked distance (Fig. 3c).

As a result, for a given command motion, the actual velocity sent to the robot $({}^rV_x, {}^rV_y, {}^r\omega)^T$ consists of the original commanded values $({}^cV_x, {}^cV_y, {}^c\omega)^T$ and compensation terms for the lateral and angular velocity. The radius of the fitting circle R and the slope of the fitting line m are used to compute the angular and lateral velocity compensation respectively:

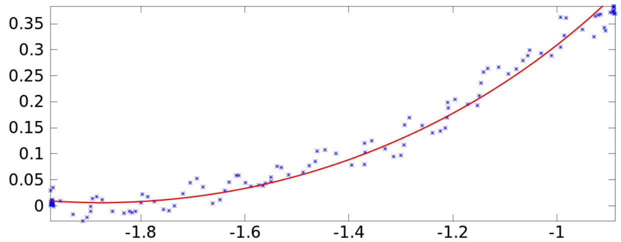
$$\begin{pmatrix} {}^rV_x \\ {}^rV_y \\ {}^r\omega \end{pmatrix} = \begin{pmatrix} {}^cV_x \\ {}^cV_y - m{}^cV_x \\ {}^c\omega - R{}^cV_x \end{pmatrix} \tag{3}$$

Finally, the velocity is translated to NAO's walk arguments (see [24] for details).

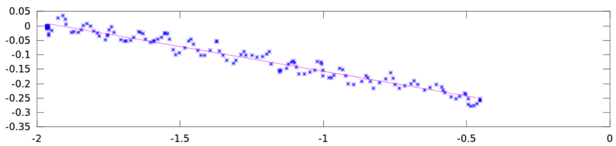
V. Experimental Results

In the experiment, the robot is commanded to approach an object lying on the floor as seen in Fig. 4. The initial distance to the object is 1.5m and the orientation of the initial pose with respect to the final pose is 25 degrees.

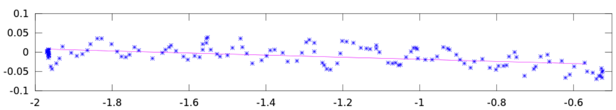
Figs. 5, 6 and 7 depict respectively the commanded velocity, the pose error, and the planar trajectory of the robot.



(a) Circle fitting.



(b) Line fitting.



(c) Final trajectory.

Fig. 3. Veering correction: the robot is commanded open-loop to walk straight with constant linear velocity. First, trajectory is recorded (a) and the best fitting circle is computed. Second, a new trajectory is recorded (b) and a linear fitting is computed. The final trajectory with angular and lateral compensation is shown in (c).

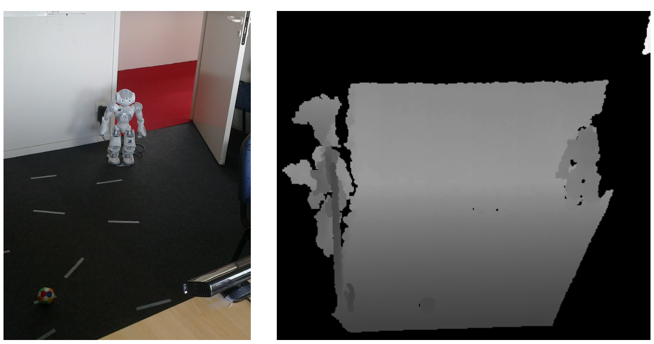


Fig. 4. Experimental setup (external and Kinect views): as NAO robot enters the room, it is commanded to move towards an object lying on the floor (the ball on the left-bottom corner of the image). Floor lines are not considered in the experiment.

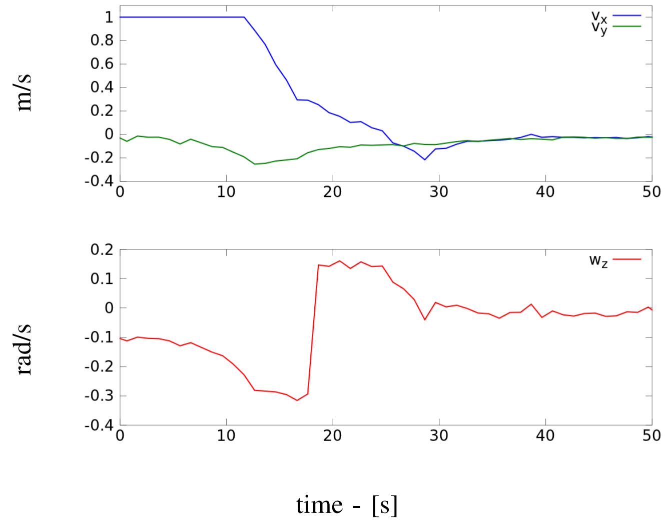


Fig. 5. Robot velocity in NAO's local frame: only planar and angular velocity is sent to the robot controller.

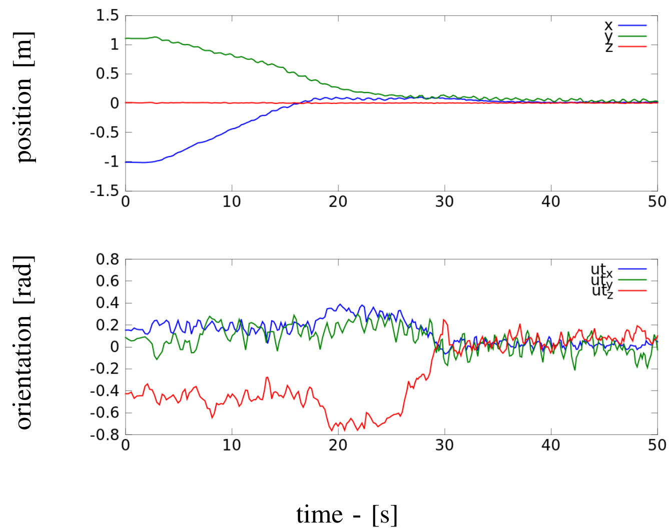


Fig. 6. Pose error of the robot torso (position and orientation), as measured by the Kinect sensor.

As can be seen in Fig. 5, the robot initially walks towards the destination at full speed, then it progressively decreases its velocity as the error is diminished. The profile of the angular velocity reflects the two stages defined in 2: from 0s to 19s, the robot turns towards the destination; after 19s the robot is nearer to the destination than the threshold (fixed to 30cm) and it turns to its final orientation.

This control strategy explains why the angular error ut_z in Fig. 6 does not decrease initially, since this error is measured with respect to the final orientation of the robot.

The trajectory of the robot in the room is depicted in Fig. 7. When using only odometry, the robot is not able to attain the destination goal, but the final pose is 40 cm away from the desired one. On the other hand, localization and closed-loop navigation allows the robot to attain the goal within a few centimeters precision. The final mean position error is $(x, y, \theta) = (0.012, 0.018, 0.07)$ and the standard deviation is $(\sigma_x, \sigma_y, \sigma_\theta) = (0.002, 0.004, 0.05)$. In preliminary experiments, it has been possible to fetch an object from the floor in such conditions.

VI. Conclusion

We have presented a localization and navigation method for a humanoid robot in a smart environment, where a Kinect sensor is used for monitoring and tracking the 3D pose of the robot.

This method is suitable for indoor environments, allowing not only to detect the robot but also to track people who

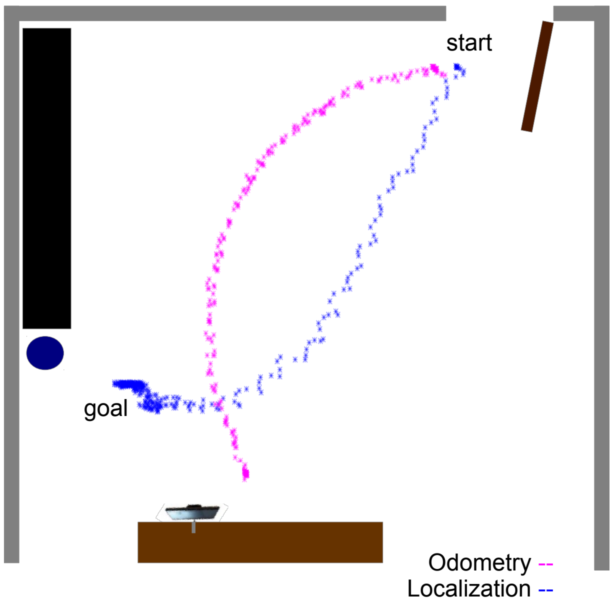


Fig. 7. Trajectories carried out in the experiments: while the odometry diverges, localization is able to attain the goal.

interact with the robot.

The accomplished precision of localization makes it possible to retrieve objects from the floor for assistance and service robotics. Further work will incorporate object manipulation to complete such task.

Further improvements are possible: obstacle detection with the Kinect sensor would allow the robot to navigate robustly in a cluttered environment. The workspace of the robot would be scalable by using more Kinect sensors in a networked system. Finally, multimodal user interaction can be achieved by gesture and voice recognition.

Acknowledgment

This research was partly funded by Ministerio de Ciencia e Innovación (DPI2011-27846), Ministerio de Educación (Programa ”Salvador de Madariaga”, referencia PR2011-015) Generalitat Valenciana (PROMETEO/2009/052) and Fundació Caixa Castelló-Bancaixa (P1-1B2011-54). We wish to thank W. Khalil for making it possible for one of the authors to stay at the IRCCyN, Ecole Centrale de Nantes.

References

[1] A. Jain and C. Kemp, “El-e: an assistive mobile manipulator that autonomously fetches objects from flat surfaces,” *Autonomous Robots*, vol. 28, pp. 45–64, 2010.

[2] C.-H. King, T. L. Chen, Z. Fan, J. D. Glass, and C. C. Kemp, “Dusty: an assistive mobile manipulator that retrieves dropped objects for people with motor impairments,” *Disability and Rehabilitation: Assistive Technology*, vol. 7, no. 2, pp. 168–179, 2012.

[3] A. Hornung, K. M. Wurm, and M. Bennwitz, “Humanoid robot localization in complex indoor environments,” in *2010 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems - IROS'10*, oct. 2010, pp. 1690 – 1695.

[4] A. Sanfeliu, N. Hagita, and A. Saffiotti, “Network robot systems,” *Robotics and Autonomous Systems*, vol. 56, no. 10, pp. 793 – 797, 2008.

[5] A. Dingli, D. Attard, and R. Mamo, “Turning homes into low-cost ambient assisted living environments,” *International Journal of Ambient Computing and Intelligence (IJACI)*, pp. 1–23, 2012.

[6] E. Stone and M. Skubic, “Evaluation of an inexpensive depth camera for passive in-home fall risk assessment,” in *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2011 5th International Conference on*, may 2011, pp. 71 –77.

[7] B. Ni, G. Wang, and P. Moulin, “Rgbd-hudaact: A color-depth video database for human daily activity recognition,” in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, nov. 2011, pp. 1147 –1153.

[8] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, “Robust monte carlo localization for mobile robots,” 2001. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.18.8488

[9] J.-S. Gutmann, “Markov-kalman localization for mobile robots.” in *ICPR (2)*, 2002, pp. 601–604. [Online]. Available: http://dblp.uni-trier.de/db/conf/icpr/icpr2002-2.html#Gutmann02

[10] J. Ido, Y. Shimizu, Y. Matsumoto, and T. Ogasawara, “Indoor navigation for a humanoid robot using a view sequence.” *I. J. Robotic Res.*, vol. 28, no. 2, pp. 315–325, 2009. [Online]. Available: http://dblp.uni-trier.de/db/journals/ijrr/ijrr28.html#IdoSMO09

[11] S. Owald, A. Hornung, and M. Bennewitz, “Learning reliable and efficient navigation with a humanoid.” in *ICRA*. IEEE, 2010, pp. 2375–2380. [Online]. Available: http://dblp.uni-trier.de/db/conf/icra/icra2010.html#OsswaldHB10

[12] M. Bennewitz, C. Stachniss, W. Burgard, and S. Behnke, “Metric localization with scale-invariant visual features using a single perspective camera.” in *EUROS*, ser. Springer Tracts in Advanced Robotics, H. I. Christensen, Ed., vol. 22. Springer, 2006, pp. 195–209. [Online]. Available: http://dblp.uni-trier.de/db/conf/euros/euros2006.html#BennewitzSBB06

[13] A. Pretto, E. Menegatti, M. Bennewitz, W. Burgard, and E. Pagello, “A visual odometry framework robust to motion blur.” in *ICRA*. IEEE, 2009, pp. 2250–2257. [Online]. Available: http://dblp.uni-trier.de/db/conf/icra/icra2009.html#PrettoMBBP09

[14] R. Cupec, G. Schmidt, and O. Lorch, “Experiments in vision-guided robot walking in a structured scenario,” in *Industrial Electronics, 2005. ISIE 2005. Proceedings of the IEEE International Symposium on*, vol. 4, 20-23, 2005, pp. 1581 – 1586.

[15] C. Stachniss, M. Bennewitz, G. Grisetti, S. Behnke, and W. Burgard, “How to learn accurate grid maps with a humanoid.” in *ICRA*. IEEE, 2008, pp. 3194–3199. [Online]. Available: http://dblp.uni-trier.de/db/conf/icra/icra2008.html#StachnissBGBB08

[16] F. Faber, M. Bennewitz, A. Grg, C. Gonsior, D. Joho, M. Schreiber, and S. Behnke, “The humanoid museum tour guide robotinho,” in *in IEEE Int. Symp. on Robot and Human Interactive Communication*, 2009.

[17] R. A. Tllez, F. Ferro, D. Mora, D. Pinyol, and D. Faconti, “Autonomous humanoid navigation using laser and odometry data.” in *Humanoids*. IEEE, 2008, pp. 500–506. [Online]. Available: http://dblp.uni-trier.de/db/conf/humanoids/humanoids2008.html#TellezFMPF08

[18] S. Thompson, S. Kagami, and K. Nishiwaki, “Localisation for autonomous humanoid navigation.” in *Humanoids*. IEEE, 2006, pp. 13–19. [Online]. Available: http://dblp.uni-trier.de/db/conf/humanoids/humanoids2006.html#ThompsonKN06

[19] K. Khoshelham and S. O. Elberink, “Accuracy and resolution of kinect depth data for indoor mapping applications,” *Sensors*, vol. 12, no. 2, pp. 1437–1454, 2012.

[20] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, and B. Marnier et al., “Mechatronic design of nao humanoid,” in *IEEE Int. Conf. on Robotics and Automation*, may 2009, pp. 769 –774.

[21] R. Rusu and S. Cousins, “3d is here: Point cloud library (pcl),” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, may 2011, pp. 1–4.

[22] R. Ueda, “Tracking 3D objects with Point Cloud Library,” http://pointclouds.org/news/tracking-3d-objects-with-point-cloud-library.html, 2012, [Online; accessed 27-June-2012].

[23] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “ROS: an open-source robot operating system,” in *ICRA Workshop on Open Source Software*, 2009.

[24] A. Robotics, “Walk control – NAO Software 1.12.5 documentation,” http://www.aldebaran-robotics.com/documentation/naoqi/motion/control-walk.html, 2012, [Online; accessed 27-June-2012].

[25] C. S. Kallie, P. R. Schrater, and G. E. Legge, “Variability in stepping direction explains the veering behavior of blind walkers,” *Journal of Experimental Psychology: Human Perception and Performance*, vol. 33, no. 1, 2007.