Humanoid Robot Indoor Navigation Based on 2D Bar Codes: Application to the NAO Robot

Laurent George[1] and Alexandre Mazel[1]
Aldebaran Robotics A-Lab
Submission 19 Humanoids 2013 (working copy)

***Abstract*— In this paper, we put forward a method for humanoid robot indoor localization and navigation, using 2D bar codes, running on embedded hardware. Our approach is based on camera image processing and the detection of 2D bar codes stuck to the walls. The particularities of our approach are: it works without odometry, fast localization (≈1s) that allows for kidnapping and falls, and it does not require the manual edition of a map (thus no extra computer is needed).**

Results of an experimental study conducted with three different NAO robots in our office show that the proposed system is operational and usable in domestic environments. The NAO robots are able to navigate through a corridor of 5m in under 86s on average. Moreover using our system the robots manage to map and navigate a complex environment with multiple rooms that includes doors and furniture.

Taken together our results demonstrate that our navigation system could become a standard feature for our robots.

I. INTRODUCTION

Aldebaran[2] main objective is to make humanoid robot a real companion for domestic applications. To reach this goal humanoid robots need to be able to localize and navigate in the user's domestic environment.

We are particularly interested in systems that use only 2D camera information. Such a system should meet the following requirement:

- rely on one 2D camera only (i.e. no laser, nor 3D camera)
- run on the embedded hardware (i.e. no need of a remote computer)
- tolerant to falls, extinction and kidnapping problem
- support of indoor environment update (e.g. adding new rooms, update some way points path)
- usable by a wide audience (e.g. no manual map edition)
- allowing an unsupervised learning of the environment
- does not require a continuous flow of stable images (which is difficult to get on an humanoid robot during the walk)

In this paper we propose to use 2D bar code stick to the walls in order to answer these needs. The proposed approach enables the automatic creation of the indoor environment maps and allows humanoid robot like the NAO robot to navigate the environment.

This paper is structured as follows. Related work is presented in Section II. In Section III we detail our approach based on 2D bar code detection. An implementation of this approach has been evaluated using three NAO robots and is presented in Section IV. We discuss results in Section V. The main conclusions are summarized in Section VI.

II. RELATED WORK

Several techniques have been investigated for indoor environments navigation (e.g. RFID identification, Wlan connectivity, laser [1]). In this section we present related works based on computer vision for indoor navigation [2], [3].

Positioning vision based systems could use landmarks presented in the environment. The landmarks could be natural or artificial. SLAM techniques could be used based on natural keypoints [4] or on specific texture available in the environment, like marks on the wood floor [5]. Using this last approach, a duration of 90s is required to travel a 5m corridor with the NAO robot. However this approach strongly relies on the floor texture and won't be usable on a uniform floor (e.g. in a hospital corridor). Moreover SLAM could be CPU intensive, and would not allows additional tasks during navigation. The introduction of artificial markers could then provide a solution that works in a wide range of environments. Moreover the use of artificial markers allows to simplify the identification of targets which could result in an increase of navigation speed.

Xu et al. investigate the use of printed pictures pasted on the walls of a corridor to track the robot position [6]. But they proposed to manually construct a map of the indoor environment with the position of the printed pictures. Their system allows to navigate a 5 meters corridor in 200s[1]. Elmogy et al. proposed a similar approach based on 2D symbols (e.g. trademarks symbol of restaurant) to fast detect landmark during navigation [7]. They use stereo vision to estimate distance of landmarks and deported processing on a laptop to provide real-time processing.

Another approach consists in using artificial marks that are well suited for fast image detection such as 2D bar codes. Classical bar codes such as QR code or Datamatrix [8] could be used. Specific bar codes have also been designed specifically for mobile robot positioning [9]–[11]. The aim behind the design of new marker, whereas existing bar code such as Datamatrix or QR code are used in numerous industrial contexts, is to provide a marker well suited for localization application and not just to carry information. However the free availability of Datamatrix and QR codes and the existence of dedicated open source library for their

[1][lgeorge, amazel]@aldebaran-robotics.com.
[2]Aldebaran-Robotics, 170 rue Raymond Losserand, Paris, France.

[1]The navigation speed is not provided by the authors, however the video of their navigation allow us to compute an approximate duration.

corners detection make them suitable for pose estimation in robot navigation context.

To our best knowledge there is no previous work based on artificial landmark detection and 2D camera, that runs on embedded hardware (i.e. without requiring remote computer), that does not require the manual creation or the edition of a map and that allows a humanoid robot to navigate accurately in indoor environments.

III. USING 2D BAR CODES FOR LOCALIZATION AND NAVIGATION

This section describes our approach to humanoid robot navigation using 2D bar codes. Bar codes are stuck to the walls in an indoor environment (see Figure 1). The density and placement of the bar codes is strongly linked to the geometry of the room, see Appendix A for examples.

Camera images are acquired and processed in order to estimate the pose x_t (position and orientation) of the robot in respect to the viewed bar codes.

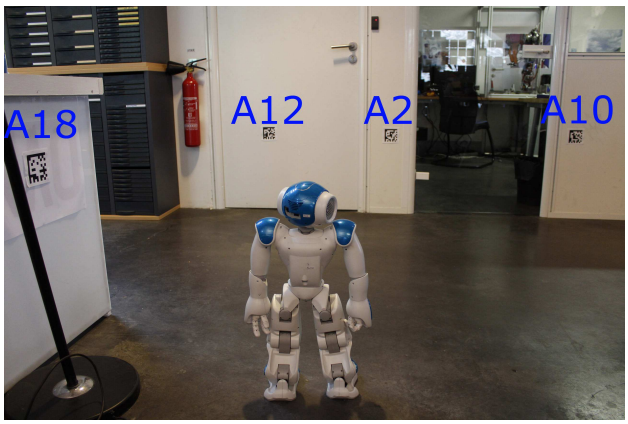


Fig. 1: The indoor environment with the bar codes stuck on the walls.

A training stage is used to learn a 2D map of the bar codes available in the environment. During this stage the robot pan his head to detect the bar codes.

The navigation relies on the robot pose estimation based on the current seen bar codes and on the learned 2D map. A way points path to reach the robot destination is created and updated each time a pose estimation is computed. The map is also used to orientate the robot camera. The next best visible 2D bar code in terms of view angle and distance is computed based on the robot future position. This allows to reduce the searching of bar code and to increase the speed of the navigation. It should be noted that all the processing is done on the on-board computer and does not require any remote connection. We detail the different steps of our approach in the following sections.

A. Bar code detection and pose estimation

We use DataMatrix as the artificial landmarks. A Datamatrix is a two-dimensional bar code [12] composed of an ordered grid of dark and light dots bordered by a finder pattern. The finder pattern defines the shape and the dimension (i.e. number of row and column) of the bar code. In our approach, we use 10x10 matrix (that is the smallest dimensions of a datamatrix) of 8.0 cm. The data information contained in each mark is composed of Ascii digits which constitute a unique label (e.g. 'A01'). Contrary to previous work [8], [11], there is no direct relation between the bar code data and its position in the environment. Thus the user is free to put the bar codes where he wants and does not need to edit them.

In order to estimate 2D bar code pose (position and orientation) 3 steps are used: image preprocessing, detection and recognition process, and finally pose estimation.

a) Preprocessing: The aim of the preprocessing step is to find image regions that seems to contain a bar code. This step involves the use of Canny filter to find the edges in the image [13]. Then a dilation operator is used to remove potential holes between edges segments. A bounding box of each contour is then computed. Finally a score based on size, width/height ratio, area and histogram is computed for each image region. These scores are used to sort the image regions.

b) Detection and recognition process: To extract the label and precise coordinates of the bar code the open Source library libdmtx[2] is used. This software is based on the time consuming Hough algorithm and allows to extract the 4 corners of the bar code [14]. In order to limit the time consumption of the processing we use a timeout of one second for each image and process only the first candidate region based on the preprocessing scores. This allows to increase the responsiveness of the system.

c) 2D bar code pose estimation: To determine the precise pose of a 2D bar code we use the OpenCV implementation of the RANSAC algorithm which allows to compute the unique solution to the perspective-4-points problem [15]. This algorithm requires to find the intrinsic and extrinsic parameter of the camera. Thus we use a calibration step where several views of a chessboard printed pattern is presented to the robot camera[3].

For these processing we use an image resolution of 1280x960 that allows to detect bar code at a distance of 2.5m and a maximum angle of $\pi/4$. However the pose estimation accuracy decreases with high distance ($>$1.5m).

The three steps (preprocessing, detection and pose estimation) allow to estimate the pose of each bar code in respect to the robot camera position.

B. 2D bar codes map creation

A learning step is used to construct a map of the bar codes presented in the environment. This step involves several observation points. Two versions of the learning procedure are available: the supervised one and the automatic version. In supervised version, the user places the robot at chosen observation points (e.g. points far from obstacles, points that

[2] `http://www.libdmtx.org/`

[3] On the NAO robots used during the evaluation the same intrinsic and extrinsic parameter are used.

allow to see numerous 2D bar codes). The user can also tell the robot (e.g. using voice recognition) the name of the current observation point. In the automatic version the robot moves around in the environment and regularly stops to create observation points.

In both versions, for each observation point, the robot looks around by panning his head[4] in order to find the visible bar codes. For each found 2D bar code, the pose is estimated as presented in Section III-A. In order to increase the accuracy of the pose estimation several images are used for the same 2D bar code (in our experimentation we get good results by averaging the estimation on 15 images which takes approximately 15 seconds for each 2D bar code). Moreover an index of confidence based on variance of landmark corners is computed. If the corners positions show an important variance then the landmark is ignored. It could happen when the landmark is far from the camera (i.e. more than 2m).

A local map of visible landmarks is then created (by using the panning angle combined to the pose estimation). This local map consists in a list of landmarks associated to a unique label name. The information stored for each 2D bar code is the coordinates relative to the current observation point, the normal vector which correspond to the bar code orientation, and the optional associated name of the landmark (see Figure 2).

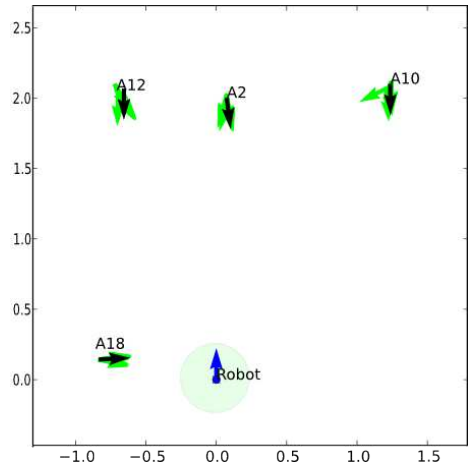


Fig. 2: Local map that corresponds to the picture presented in Figure 1. The map has been automatically build thanks to the bar codes pose estimation and NAO accurate head panning. *Black arrows* indicate the mean position and orientation of 2D bar codes stuck on the walls. *Green circle* indicates the observation point used. All units are in meters.

When numerous local maps have been learned a global 2D map could be constructed. The global map consisted in the connection of the different local maps. The connection of two local maps is done using a bar code that is visible in both local maps. If there is such a landmark the two maps could be connected using rotation and translation of the second local map. If more than one 2D bar code is visible in both local maps, the 2D bar code that is the closest to the observation points is used. Figure 3 shows a global map constructed using two different local maps. The creation of a new global map could be done at any time, for instance when a new local map is learned or when a local map is deleted. This could be used to update the maps when adding new rooms.

[4]This method is particularly efficient on NAO robot where the panning rotation is very accurate. This allows to get precise head yaw angle.

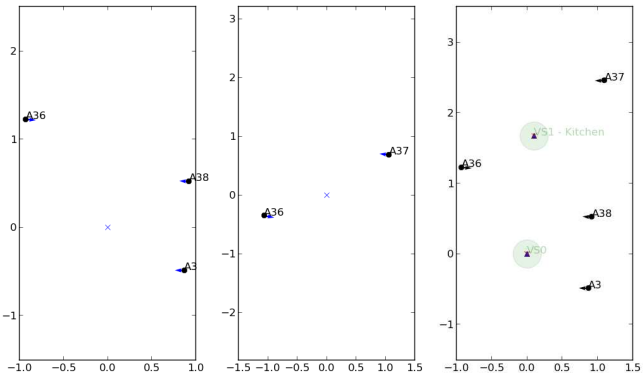


Fig. 3: A global map (on the right) built using two local maps (on the left). The bar code *A36* is used to connect the two local maps (the second local map has been associated to the label name *Kitchen* by the user).

Another feature of the system is the possibility to have different global maps. It allows the robot to navigate rooms that are not connected (e.g. even if the robot knows the kitchen and the bedroom but not the path linking them, the robot can still localize and navigate inside each room; lately the connection could be made, by learning the corridor between the rooms).

The global map also provides information concerning points that are reachable (e.g. no obstacle). For this purpose the observation points are used: the robot has been at each observation point during the learning so they can be considered as reachable. Moreover in the supervised learning it is the human user who has chosen the observation points for their reachability. For this reason we save these observation points in the global map. They will be used to build a way point path for navigation.

Finally, the interaction API also allows the user to enhance the map by adding information such as action to perform near a specific point (e.g. walk slowly near specific point, etc.).

C. Robot pose estimation

The observation of one bar code visible in the global map is sufficient to estimate the pose x_t (position and orientation) of the robot at time t in the environment. Indeed the observation of one landmark allows to compute the robot relative position/orientation to this mark by solving the pose estimation problem. Then using the previously learned information (i.e. position and normal vector of the mark in the global map) the robot position and orientation in the global map could be computed using simple geometrical translation and rotation operations.

This pose estimation is done using a single camera shot. The robot can thus know his localization in about one second (i.e. time for image processing) if a bar code is visible. Moreover if the robot is moved by the user, just after seeing a mark the robot knows his new localization.

D. Navigation

To navigate the indoor environment, a path from the current robot position to a destination has to be computed. Thus three kinds of information are used: the robot pose (position and orientation), the destination point coordinates and the observation points coordinates. The robot pose corresponds to the current robot position and orientation in the global map, estimated thanks to a viewed landmark (see Section III-C). The destination point is any point in the global map. It could be an observation point but not necessarily. The observation points are the places where the poses of the landmarks have been learned (see Section III-B). They are considered to be reachable in the environment. Two observation points are assumed to be connectible (i.e. the robot can go from one point to another) if one bar code is visible from the two observation points.

The first step in the navigation procedure is to go to an observation point. Then from one observation point to another, approach the destination point. And finally go to the destination point. In order to accomplish this procedure, we first determined the closest observation point to the robot position (p_1). We also compute the closest observation point to the destination (p_{last}). Then using a shortest-path algorithm we construct the path of observation points from p_1 to p_{last}.

While navigating, in order to avoid turnaround (i.e. loop during navigation), we use a condition on the angles concerning the use of the first way point p_i. If the use of p_i causes a turnaround we use the second point on the path (p_{i+1}) instead. More precisely we use the following condition:

$r_{coord}, r_{vect} \leftarrow$ robot position, robot orientation
$a1 \leftarrow$ angle$(r_{vect}, p_i - r_{coord})$
$a2 \leftarrow$ angle$(p_i - r_{coord}, p_{i+1} - p_i)$
$b3 \leftarrow$ angle$(r_{vect}, p_{i+1} - r_{coord})$
if $abs(a1) + abs(a2) > abs(b3) + \pi/2$ **then**
    Use p_{i+1}
else
    Use p_i

The whole navigation procedure (closest observation point, path construction) is launched each time the robot position is updated. This is not optimal in terms of computing time, but it allows to be tolerant to kidnapping problems. For instance, if the robot is moved during a navigation, the robot can still continue to walk to his destination.

When the robot is walking from one way point to another, his head orientation is updated in order to always look at the best bar code (in terms of distance and angle of view). To reduce the blur of movement during the walk and still be able to see the landmarks, the camera exposure time is reduced. However the walk could still introduce error in pose estimation, thus the image analyzed during the walk is just used to check that the robot is still on the path. If he has drifted too much since the last movement order the robot is stopped and a new image analysis is done.

IV. EVALUATION

To evaluate our approach to navigation based on 2D bar code we carried out a real-world experiment in an office environment using humanoid robots. Two navigation tasks (corridor navigation, multiple rooms navigation) have been tested.

A. Apparatus

NAO robots have been used during the evaluation. NAO is a small humanoid robot (58 cm tall). It is equipped with two video cameras located in the head. Each camera provides a horizontal field of view of 60.9 degrees, and vertical field of view of 47.6 degrees.

It's known that the more a robot is used, the more the backlash increases which results in inaccurate walk and drift. These errors are unfortunately not detected with the odometry on NAO robot that is why we do not rely on odometry. To check that our approach still allow the robot to navigate with this problem we use three different robot with different wears for the evaluation. A first one just unpack from the box (NAO A), a second one that has been used in our team since some months (NAO B), and the last one that has been used extensively in demonstration (NAO C).

B. Procedure

Two navigation tasks have been used. The first one named **corridor navigation** consists in reaching a point located at 5m from initial position in a corridor. The corridor is 2m20 wide and 6m long. Seven bar codes have been stuck at a height of 55cm on either side of the corridor and at its end[5]. The average distance between marks is about 1m. Five runs have been recorded for each robot.

The second task, **multiple rooms navigation**, aims at addressing a general navigation scenario that could occurs at the home of our customers. It consists in navigating from one point to a non-visible one. We used our office and the office reception (approximate distance: 15 meters, 15 bar codes stuck on walls and doors at a height of 55cm). One run has been recorded for each robot.

For both tasks, the move command used after each position estimation is a movement of 75cm length in the direction of the next way point.

C. Collected data

For each navigation task and each run we recorded the navigation duration, the robot position (based on seen 2D bar codes), and the navigation distance. Moreover the computed theoretical way points, based on the move commands sent to the robot, have also been recorded.

[5]Corridor setup is presented in the video available at `http://bit.ly/navmark`

D. Corridor navigation results

Figure 4 shows the detailed path taken by the first robot to reach his destination. Results of the five runs are presented in the last column of Table I. As expected the navigation task for the newest robot (NAO A), which has a low level of wear, is faster (mean duration of 58.2 sec vs. approximately 100 sec for the two other robots). Moreover the navigation is fast for all the robots. The average duration time on the three robots is below 86 sec which is better than previous results with NAO robot (Osswald et al. reported a navigation duration of 90 sec [5], Xu et al. show a navigation duration of approximately 200s for 5m [6]). This result could still be improved; indeed the theoretical best duration is about 30 seconds (NAO maximum speed in straight line for 5 meters).

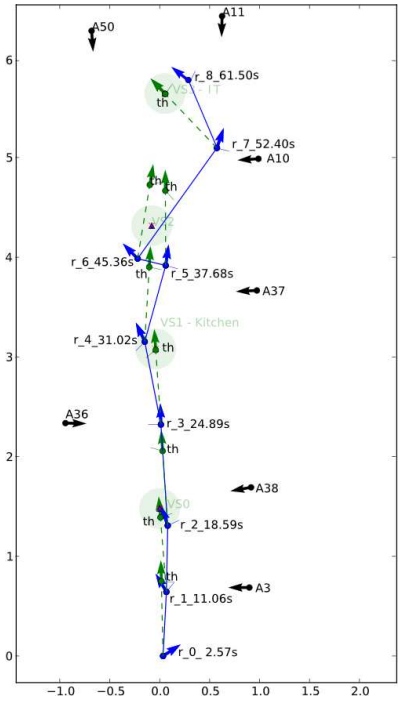


Fig. 4: A navigation path followed by the robot NAO A during the corridor task. It corresponds to the navigation run presented in the video available at `http://bit.ly/navmark`. *Blue arrows* indicate robot position and orientation estimated using bar code detection, including the associated elapsed time. *Green arrows* indicate theoretical robot position computed using the move commands. Current used bar code associated to a robot position are indicated with a *straight line* at the bottom of each blue navigation arrow.

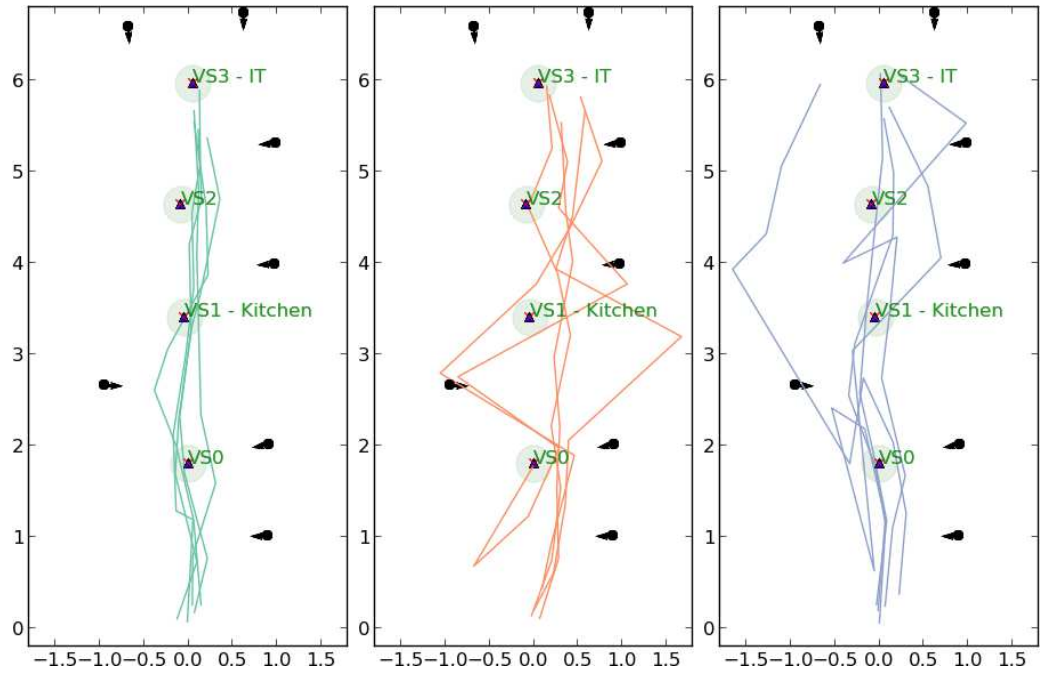
The distance to reachable way points during the navigation is a parameter as important as the duration of the navigation. Indeed if a robot goes far from the navigation reachable points it increases the risk of collisions and falls. To evaluate this distance in the different navigation runs we computed the ratio between theoretical best path (straight path from initial position to destination) and the estimated path followed by the robots. Results are presented in Table I.

TABLE I: Navigation duration and distance error to theoretical path for the three robots on five runs during the corridor navigation task (move command of 75cm length).

Robot	Duration (s)	Estimated error (%)
NAO A	59.73	1.14
NAO A	49.66	2.19
NAO A	56.88	2.47
NAO A	76.56	4.33
NAO A	48.26	1.79
NAO B	77.59	29.95
NAO B	105.53	3.87
NAO B	101.77	27.63
NAO B	94.41	1.11
NAO B	117.67	118.72
NAO C	109.70	45.40
NAO C	57.06	0.91
NAO C	153.15	33.54
NAO C	129.68	78.16
NAO C	64.74	2.96
Avg.	86.83	23.61

The first robot, NAO A, remained close to the theoretical straight path (error below 3% on average). The second and third robots (NAO B and C) show a higher error distance (average error of 33% and 36% respectively). This could be explained by the higher level of wear of the two last robots. Figure 5 show the corresponding navigation paths for each robot. The drift of the robot NAO B and NAO C are clearly visible here, whereas NAO A remains close to the straight theoretical path.

To reduce the error distance to theoretical straight path, two solutions could be explored. The first one consists in doing registration of path planning during movements. However the blur introduced by the motion and the variation in robot pose could introduce errors in pose estimation and results in an important error in move commands. Preliminary results tend to confirm this hypothesis: registration of pose during movement show an increase in navigation duration in preliminary tests. To handle the pose estimation error the use of a filter is required as the UKF filter proposed in [5]. A second approach is to reduce the navigation movement length used after each pose estimation. A side effect of reducing the navigation movement length is an increase in the number of stops which introduces an increase of global navigation duration. On the other side, the robot will benefit of a reduced mark finding duration: it would be better oriented due to reduction of drift and would then require less time consuming panning. Results obtained with robot NAO C when using a movement length of 20cm are presented in Figure 6. These results show that the robot remains close to the straight theoretical path (average error below 19 percent vs. 32 percent using 75cm movements). The duration of the navigation is increased on average by 16 seconds. This increase duration could be compensated by the avoidance of obstacles in more complex navigation task where remaining close to the reachable points is required.



(a) NAO A. (b) NAO B. (c) NAO C.

Fig. 5: Navigation paths for the three robots during the corridor task (move command of 75cm length).

Robot	Duration (s)	Error (%)
NAO C	102.10	24.27
NAO C	122.75	11.08
NAO C	107.36	22.50
NAO C	137.50	16.70
NAO C	124.35	16.81
Avg.	118.8	18.27

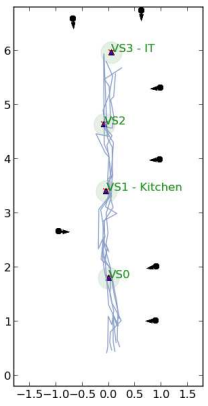


Fig. 6: Duration, error distance and navigation path for robot NAO C (move command is 20cm length). Results show that the robot remains close to the theoretical path.

E. Multiple rooms navigation results

To check that our system allows humanoid robot to navigate from room to room we conducted three navigation runs from our office to the office reception using the three robots. The navigation length is about 15 meters. The two rooms are separated by a door of 80 cm wide. Different furniture or objects are present on the path (e.g. a sofa, tables, a fire extinguisher, etc.). The light is not uniform on the path (e.g. destination room presents a huge window whereas office rooms is illuminated with artificial light).

Results of the runs show that our approach allows all the robots to reach their destination despite the difficulty of the task. The duration of the runs are 245s, 496s and 379s for robot NAO A, NAO B, and NAO C respectively. It should be noticed that the robot NAO B has fallen during the run. He has drifted too much and collided with a piece of furniture. However after he got up, he continues to navigate the environment successfully.

V. DISCUSSION

The main limitation of our approach concerns the intrusiveness of the proposed setup. Indeed black and white bar codes are particularly visible (8cm wide) and thus could be disturbing in the user environment. An interesting solution to this limitation is the one proposed by Huh et al. [16] who proposed to use invisible bar code illuminated by UV-light. Another solution is to use smaller bar code, but it would introduce more error in pose estimation. In our experiment we use the same bar code size, but it could be interesting to use different sizes (e.g. small bar codes in small rooms). A good way to enable this would be to include the physical size of the bar code directly in the bar code data [8].

During the different evaluation runs, humans users present at the office report that the head rotation of the robot head could be disturbing because it went over $pi/4$. A threshold on the head orientation should thus be envisioned.

Concerning the reorientation of the robot during the walk, more analysis should be conducted. In this paper we do

not use movement reorientation during the walk as it shows poorer outcomes in preliminary tests. However this solution provides a more natural walk without any stops and thus should be explored. It could for instance increase acceptance of the robot by the users.

The use of an automatic adaptation of the movement length based on indoor information (e.g. number of obstacles, wide of corridors) or on the robot estimated drift could also be envisioned in order to reduce the error distance between robot and theoretical path.

Concerning the 2D barcode used, datamatrix provided accurate and fast detection. However the use of specific markers designed for robot vision like ARTag [9] needs to be evaluated.

VI. CONCLUSION

In this paper we propose to use bar code landmarks in order to provide an accurate indoor navigation capabilities to NAO humanoid robot. Experimental results obtained with three different robots show that the proposed system is fully operational and robust. Moreover the navigation speed is quite fast compared to previous work found in the literature and the proposed approach even works on robot with a huge drift. Future work will focus on the improvement of the landmark detection during movements and in different light conditions. Applications that could be based on our approach include: objects search in the user's flat, going to the charging station, or game application like hide and seek.

ACKNOWLEDGEMENTS

This work has been funded by the French agency OSEO on the PSPC project Romeo 2. The authors would also like to thank Rodolphe Gelin for his constant support, helpful remarks and his kindness.

REFERENCES

[1] A. Hornung, K. Wurm, and M. Bennewitz, "Humanoid robot localization in complex indoor environments," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 1690–1695.

[2] F. Bonin-Font, A. Ortiz, and G. Oliver, "Visual navigation for mobile robots: A survey," *J. Intell. Robotics Syst.*, vol. 53, no. 3, p. 263296, Nov. 2008. [Online]. Available: http://dx.doi.org/10.1007/s10846-008-9235-4

[3] G. N. DeSouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 2, p. 237267, 2002. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=982903

[4] E. Wirbel, B. Steux, S. Bonnabel, and A. de La Fortelle, "Humanoid robot navigation: from a visual slam to a visual compass," 2013.

[5] S. Osswald, A. Hornung, and M. Bennewitz, "Learning reliable and efficient navigation with a humanoid," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 2375–2380.

[6] J. Xu, C. Wei, C. Wang, P. Wiggers, and K. Hindriks, "An approach to navigation for the humanoid robot nao in domestic environments," 2012. [Online]. Available: http://mmi.tudelft.nl/sites/default/files/jccpk_navigationpaper.pdf

[7] M. Elmogy and J. Zhang, "Robust real-time landmark recognition for humanoid robot navigation," in *IEEE International Conference on Robotics and Biomimetics, 2008*, 2008, pp. 572–577.

[8] H. Kobayashi, "A new proposal for self-localization of mobile robot by self-contained 2d barcode landmark," in *SICE Annual Conference (SICE), 2012 Proceedings of*. IEEE, 2012, pp. 2080–2083.

[9] M. Fiala, "ARTag, a fiducial marker system using digital techniques," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005*, vol. 2, 2005, pp. 590–596 vol. 2.

[10] R. Zheng and K. Yuan, "MR code for indoor robot self-localization," 2008, pp. 7449–7454.

[11] G. Y. Lin and X. Chen, "Robot indoor navigation method based on 2D barcode landmark," *Applied Mechanics and Materials*, vol. 44, p. 12791284, 2011. [Online]. Available: http://www.scientific.net/AMM.44-47.1279

[12] R. Stevenson, "Laser marking matrix codes on pcbs," *Printed Circuit Design and Manufacture*, vol. 22, no. 12, p. 32, 2005.

[13] J. Canny, "A computational approach to edge detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[14] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Commun. ACM*, vol. 15, no. 1, pp. 11–15, Jan. 1972. [Online]. Available: http://doi.acm.org/10.1145/361237.361242

[15] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981. [Online]. Available: http://doi.acm.org/10.1145/358669.358692

[16] J. Huh, K. Lee, W. K. Chung, W. S. Jeong, and K. K. Kim, "Mobile robot exploration in indoor environment using topological structure with invisible barcode," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 5265–5272.

APPENDIX

A. Bar code positioning

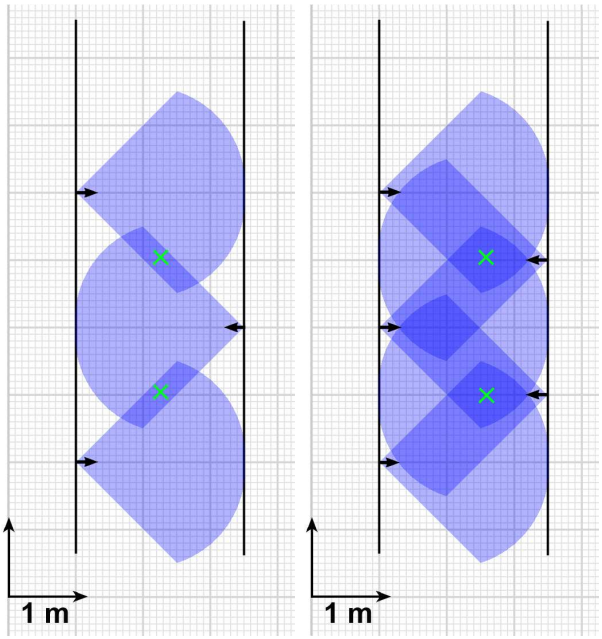


Fig. 7: Theoretical positioning of 2D bar codes in a corridor. On the left a good compromise between coverage and number of bar codes. On the right a full coverage of the corridor. *Black arrows* indicate position and orientation of 2D bar codes stuck on the walls. *Blue areas* show the positions from where the robot could see a bar code with a good confidence. *Green cross* represent a possible robot learning position.