Adaptive Person-Following Algorithm Based on Depth Images and Mapping*

Guillaume Doisy[1], Aleksandar Jevtić[2], Eric Lucet[2] and Yael Edan[1]

Abstract— **Person following by a mobile autonomous robot includes two tasks, person tracking and safe robot navigation. Two person-following algorithms that use depth images from a Microsoft Kinect sensor for person tracking are proposed. The first one, the *path-following algorithm*, reproduces the path of the person in the environment. The second one, the *adaptive algorithm*, uses in addition a laser range finder for localization and dynamically generates the robot's path inside a pre-mapped environment, taking into account the obstacles locations. The Kinect was mounted on a pan-tilt mechanism to allow continuous person tracking while the robot followed the generated path. The two algorithms were tested and their performance compared in a series of trials where the robot had to follow a person walking in an environment with obstacles. With both algorithms the robot could perform continuous person tracking when the obstacles were lower than the height of the camera mount. With the adaptive algorithm the distance travelled by the robot was 29.6% shorter than with the path-following algorithm; however the path-following algorithm does not require a pre-build map of the environment.**

I. INTRODUCTION

Person following for mobile robots is advantageous in applications that require close human-robot interaction. In some applications, such as for a robot companion, having this feature is very important. There are many challenges in development of efficient and human-like person following robot behaviour, e.g., safety of humans and robots, user acceptance or ethical issues.

Person following consists of two tasks, namely person tracking and robot navigation. In real-world applications the person-following algorithms must take into account the environment constraints. For indoor applications mapping the environment allows safer and more efficient robot navigation, but often it must also consider the movement of objects and other people. In such situations, the person-following behavior must be adaptive so the robot can update the path to the desired destination point taking into account the new constraints.

Person detection and tracking is the first necessary step in the person following task. Many proposed person-following algorithms use vision as input [1], [2]. Measurements from a laser range finder (LRF) can be used to extract the patterns of the person's legs [3]; however, similar patterns can represent chairs and tables which makes correct detection difficult. To improve the tracking performance some authors proposed fusion of LRF with infrared camera [4] or with omnidirectional camera [5].

Depth images have been used for visual tracking [6]. They provide information about the distance of the objects in the image. Detection results can be improved through fusion with measurements from other sensors [7] or they can be compared with the stored templates in a pre-built knowledge base [8]. Some methods propose using input from multiple cameras [9], [10].

Recently released Microsoft Kinect sensor is a low-cost and efficient alternative for depth image-based person tracking [11], [12]. Many research groups reported their activity in working on Kinect features, but few have published their results on applications to person tracking [10], [13], SLAM [14], or improved environment mapping [15].

Person following with a mobile robot must first include person detection and tracking. Further, robot navigation and path generation are applied. The initiation of these tasks can be human-operated or autonomous [16]. Various methods for person-following propose using LRF measurements for person legs detection [17], [3]; however, detected patterns are easily confused with tables and chairs, or other people's legs. Color and texture of the person's clothes were used for vision-based tracking and following in [18], [19]. Fusion of LRF and vision-based sensors showed improved person detection [20], [21], [22], [23]. Some authors proposed combining vision-based detection with RFID tracking [24], [25].

A person-following algorithm based on direction following was proposed in [26]. The input from a pair of stereo cameras was used to combine feature detection with pre-built motion models. Another method for robot motion planning based on the learned human motion patterns was proposed in [27].

Mapping of the environment in which the robot operates can simplify the motion-planning task [28]. Mapping and SLAM for mobile robots is a vast field of study [29]. Numerous methods have been proposed based on the input from LRF [30], vision-based sensors [31], 3D images [32], [14], time-of-flight cameras [33], etc. In this paper, path planning using a pre-built map is proposed. This method is compared with a method that does not benefit from mapping, and shows how mapping can allow the robot to adapt to the distribution of obstacles.

*This research was supported by the FP7 EU-funded ITN in the Marie-Curie People Programme: INTRO, grant agreement no. 238486, and partially supported by the Paul Ivanier Center for Robotics Research and Production Management, and by the Rabbi W. Gunther Plaut Chair in Manufacturing Engineering, Ben-Gurion University of the Negev.

[1]G. Doisy and Y. Edan are with the Department of Industrial Engineering and Management, Ben-Gurion University of the Negev, Beer Sheva 8410, Israel doisyg@post.bgu.ac.il, yael@bgu.ac.il

[2]A. Jevtić and E. Lucet are with Robosoft, Technopole d'Izarbel, F-64210, Bidart, France {aleksandar.jevtic, eric.lucet}@robosoft.fr

II. METHODOLOGY

A. Algorithms

Two person-following algorithms are developed and compared: a *path-following algorithm* and an *adaptive algorithm*. These two algorithms use other algorithms to control the robot and to track the position of the person and estimate its position (described in Section III).

B. Hardware

The two algorithms were implemented on a generic differential drive mobile platform with two propulsive wheels and two castor wheels, which comes with basic navigation functions (Robosoft robuLAB10 robotic platform). The robuLAB10 was customized with a rigid structure including three tubes and a tray for laptop PC (Figure 1). On the top of this structure a TRACLabs Biclops pan-tilt mechanism (PT-M) and a Kinect sensor were added. For navigation purposes, the base is equipped with a SICK S300 LRF, which is positioned at the height of $0.24m$ and provides distance measurements of up to $30m$ in an angular field of view of $270°$.

The pan-tilt mechanism has a tilt range of $120°$ and a pan range of $360°$ with a maximum angular velocity of $170°/s$ and a maximum angular acceleration of $3000°/s^2$. The precision of the angular position measurements is $\pm 0.01°$. The mechanism can support a maximum payload of $4kg$ which is more than the weight of the Kinect sensor. In all experiments, the tilt value was set to $0°$ and person tracking was performed only in the horizontal plane, using the pan axis of the pan-tilt mechanism only. The communication between the laptop PC and the mechanism is maintained via a USB port with a data transfer rate of up to $416kbps$.

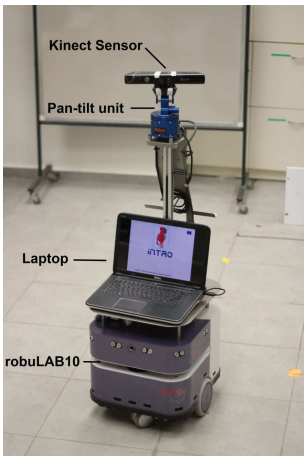

Fig. 1. RobuLAB10 robotic platform with Biclops pan-tilt mechanism, Kinect sensor, and laptop PC.

The Kinect sensor is equipped with an infrared light projector, a depth sensor, a RGB camera, and a multi-array microphone. It also has a motorized tilt that was disabled and was used only for sensor positioning. The depth sensor range is from $0.8m$ to $6m$ with the vertical viewing angle of $43°$ and horizontal viewing angle of $57°$. It provides depth images at the resolution of 640×480 pixels at the maximal frame rate of $30fps$. The Microsoft Kinect SDK provides person detection and person joints position tracking features up to $4m$.

C. Experimental setup

Two sets of experiment were conducted. The first set focused on the performance evaluation of the path-following person following algorithm and the second set focused on the adaptive person following algorithm. In all experiments, the person was instructed not to assist the robot and to walk at a constant speed along a marked path on the ground, regardless of the robot's tracking and/or following performance. This marked path on the ground makes the person travel around obstacles as seen in Figure 3 and 4.

D. Performance analysis

The following performance metrics were used for each trial of each experimental setups to evaluate the proposed person-following algorithms: 1) Path-completion ratio: the length of the ground path from the person start point to the closest point of the robot end point, divided by the total length of the ground path, 2) Number of loss-of-track events: number of events when tracking of the person was lost in a single trial; loss of tracking is defined when no position estimation is provided by the Kinect SDK for a period longer than 500ms, and 3) Robot path length to person path length ratio: the distance travelled by the robot divided by the distance travelled by the person.

For each set of experiment, 10 trials were conducted. For the path-following algorithm evaluation, the error between the person's path and robot's path was computed in addition to the metrics described above. This path error is calculated by resampling robot path data to regular space interval of 1cm and calculating for each resampled point of the robot path the closest distance to the ground path followed by the person.

III. ALGORITHMS

A. Robot control

The robuLAB10 platform uses Robosoft robuBOX open source library. The robuBOX is based on the Microsoft Robotics Developer Studio (MRDS) and written in C#. Its most important component is the Core, which contains the definitions of robots actuators and sensors. All other components interact through these definitions either by implementing or using them. For robot navigation four robuBOX features were exploited, namely the obstacle collision detection, the localization, the differential-drive controller and the path follower. The localization component uses odometry from the wheels to estimate its position and readings from a LRF continuously correct the odometry error if a map of the environment is available.

The obstacle collision detection feature uses the LRF distance measurements and applies two parameters to control the robot's motion. At distances between $0.3m$ and $1m$ from an obstacle the robot speed is reduced proportionally to the distance value. The robot is finally stopped at the distance of $0.3m$ from the obstacle. The distances are calculated within

the robot frame with its origin in the point P_m located at mid-distance of the actuated wheels.

The differential-drive controller is used to set robot's linear and angular speeds. The wheels' velocities are derived from these values by the robot's low-level controller.

The path follower feature allows the robot to follow a list of path points that are added to the buffer and executed sequentially. The path follower implements Morin-Samson's path following with no orientation control [34]. We consider a path $\mathcal{C}$ in the plane of motion, as illustrated on Figure 2. Let us define three frames $\mathcal{F}_0$, $\mathcal{F}_m$, and $\mathcal{F}_s$, as follows. $\mathcal{F}_0 = \{0, \overrightarrow{i}, \overrightarrow{j}\}$ is a fixed global frame, $\mathcal{F}_m = \{P_m, \overrightarrow{i_m}, \overrightarrow{j_m}\}$ is a frame attached to the mobile robot with its origin in the point P_m, and $\mathcal{F}_s = \{P_s, \overrightarrow{i_s}, \overrightarrow{j_s}\}$, which is indexed by the path's curvilinear abscissas, is such that the unit vector $\overrightarrow{i_s}$ tangents $\mathcal{C}$. The control point P is attached to the robot chassis, with the coordinates (l_1, l_2) expressed in the basis of $\mathcal{F}_m$. In the experiments the following values were set: $l_1 = 0.15m$ and $l_2 = 0$.

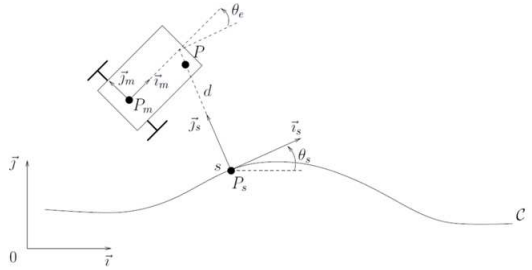

Fig. 2. Representation of the path in the robot motion plane. (Morin & Samson, 2008)

To determine the equations of motion of P with respect to the path $\mathcal{C}$ let us define d as the distance between P and $\mathcal{C}$, and $\theta_e = \theta_m - \theta_s$ as the angle characterizing the orientation of the robot chassis with respect to the frame $\mathcal{F}_s$. Where θ_m is the orientation of the robot chassis in the global frame $\mathcal{F}_0$.The control objective is to stabilize the distance d at zero. For that, the following feedback control law was applied:

$$u_2 = u_1 \left(\frac{\tan \theta_e}{l_1} - k_0 \cdot d \right) \tag{1}$$

where u_1 and u_2 represent the intensities of the robot's longitudinal and angular velocity, respectively, and k_0 is a constant. The detailed proof that d exponentially converges to zero when u_1 is constant and $\theta_e \in (-\pi/2, \pi/2)$ can be found in (Morin & Samson, 2008). The following values were set to $u_1 = 0.5m/s$ and $k_0 = 20$. As a measure of precaution, the maximal heading error was set to $\theta_{e,max} = 60°$, which initiates a recovery procedure that stops the robot and sends it to the last path point in the buffer.

B. Person tracking and position estimation

Tracking of person's skeleton joints is performed for each depth-image frame in the Kinect SDK, using no temporal information [12]. The algorithm uses the variation in depth to find different body parts and applies Random Decision Forests to compute estimated joint positions. It is also able to distinguish between two different persons. The 3D position of the head joint outputted by the algorithm was used to estimate the ground X and Y position of the person. This allows keeping track of the person position in presence of obstacles small in height causing an occlusion of the lower body parts. The outputted person ground position estimation is in the frame of reference of the Kinect sensor. It must be converted in the global frame of reference in order to be used by the path-following algorithm.

To calculate the position estimation in the global frame three direct orthonormal frames of reference were considered:

1) The fixed global frame $\mathcal{F}_0 = \{0, \overrightarrow{i_0}, \overrightarrow{j_0}\}$.
2) The frame attached to the robot $\mathcal{F}_m = \{P_m, \overrightarrow{i_m}, \overrightarrow{j_m}\}$. P_m is at the center of the robot and both $\overrightarrow{i_m}$ and $\overrightarrow{j_m}$ are in the horizontal plane; $\overrightarrow{i_m}$ is pointing in the forward direction of the robot.
3) The frame attached to the Kinect sensor $\mathcal{F}_k = \{P_k, \overrightarrow{i_k}, \overrightarrow{j_k}\}$. P_k is at the center of the Kinect sensor and both $\overrightarrow{i_k}$ and $\overrightarrow{j_k}$ are in the horizontal plane; $\overrightarrow{i_k}$ is pointing in the forward direction of the sensor.

P_m in $\mathcal{F}_0$, denoted $P_{m(\mathcal{F}_0)}$, and the angle between $\overrightarrow{i}$ and $\overrightarrow{i_m}$, denoted $\theta_{m(\mathcal{F}_0)}$, are known from odometry. P_k in $\mathcal{F}_m$, denoted $P_{k(\mathcal{F}_m)}$, is known from the hardware configuration of the robot: $P_{k(\mathcal{F}_m)} = (-0.08, 0)$. The angle between $\overrightarrow{i_m}$ and $\overrightarrow{i_k}$, denoted $\theta_{k(\mathcal{F}_m)}$, is given by the pan axis position measurement of the pan-tilt mechanism. The position of the person in $\mathcal{F}_k$, denoted $Person_{(\mathcal{F}_k)} = (X_{Person(\mathcal{F}_k)}, Y_{Person(\mathcal{F}_k)})$ is given by the output of the Kinect sensor. The angle between the forward direction of the Kinect sensor, $\overrightarrow{i_k}$, and the person, denoted $\theta_{Person(\mathcal{F}_k)}$, can be calculated:

$$\Theta_{Person(\mathcal{F}_k)} = \tan \left(\frac{Y_{Person(\mathcal{F}_k)}}{X_{Person(\mathcal{F}_k)}} \right) \tag{2}$$

The position of the person in $\mathcal{F}_m$, denoted $Person_{(\mathcal{F}_m)} = (X_{Person(\mathcal{F}_m)}, Y_{Person(\mathcal{F}_m)})$ can be calculated:

$$\begin{aligned} Person_{(\mathcal{F}_m)} = Person_{(\mathcal{F}_k)} * \\ \begin{pmatrix} \cos\left(\theta_{k(\mathcal{F}_m)}\right) & \sin\left(\theta_{k(\mathcal{F}_m)}\right) \\ -\sin\left(\theta_{k(\mathcal{F}_m)}\right) & \cos\left(\theta_{k(\mathcal{F}_m)}\right) \end{pmatrix} + \\ P_{k(\mathcal{F}_m)} \end{aligned} \tag{3}$$

The angle between the forward direction of the robot, $\overrightarrow{i_m}$, and the person, denoted $\theta_{Person(\mathcal{F}_m)}$, can be calculated:

$$\Theta_{Person(\mathcal{F}_m)} = \tan \left(\frac{Y_{Person(\mathcal{F}_m)}}{X_{Person(\mathcal{F}_m)}} \right) \tag{4}$$

Finally, the position of the person in $\mathcal{F}_0$, denoted $Person_{(\mathcal{F}_0)} = (X_{Person(\mathcal{F}_0)}, Y_{Person(\mathcal{F}_0)})$, can be calculated:

$$\begin{aligned} Person_{(\mathcal{F}_0)} = Person_{(\mathcal{F}_m)} * \\ \begin{pmatrix} \cos\left(\theta_{m(\mathcal{F}_0)}\right) & \sin\left(\theta_{m(\mathcal{F}_0)}\right) \\ -\sin\left(\theta_{m(\mathcal{F}_0)}\right) & \cos\left(\theta_{m(\mathcal{F}_0)}\right) \end{pmatrix} + \\ P_{m(\mathcal{F}_0)} \end{aligned} \tag{5}$$

C. Pan-tilt mechanism control

In order to make the Kinect sensor always point in the direction of the person tracked, a control law of the pan axis of the pan-tilt mechanism was developed. The output of this control law is an angular speed command of the pan axis, denoted $\dot{\theta}_{k(\mathcal{F}_m)(Command)}$.

A first approach to compute the speed command was to implement a P-controller using the angular position of the person in the Kinect frame, $\theta_{Person(\mathcal{F}_k)}$, as the measurement and a 0° angle as the target, $\theta_{Person(\mathcal{F}_k)(Target)}$.

$$\begin{aligned}\dot{\theta}_{k(\mathcal{F}_m)(P-control)} &= K_{p(Pan)} \cdot error \\ &= K_{p(Pan)} \cdot \\ &\quad \left(\theta_{Person(\mathcal{F}_k)(Target)} - \theta_{Person(\mathcal{F}_k)}\right)\end{aligned} \tag{6}$$

$\theta_{Person(\mathcal{F}_k)}$ is given by equation (2) and $\theta_{Person(\mathcal{F}_k)(Target)} = 0^\circ$.

Then the angular speed command is set equal to the output of the P-controller:

$$\dot{\theta}_{k(\mathcal{F}_m)(Command)} = \dot{\theta}_{k(\mathcal{F}_m)(P-control)} \tag{7}$$

We used $K_{p(Pan)} = 4\ s^{-1}$. This first approach using equation (6) for computing the speed command is able to maintain the sensor in the direction of the tracked person when the robot is not moving. However, when the robot is moving, the system is not reactive enough to keep track of the person. Loss of tracking happens when the robot is rotating or turning. To compensate for the robot rotation, a second approach was developed. Information from the odometry pose estimation is used to calculate the angular speed of the robot in $\mathcal{F}_0$, denoted $\dot{\theta}_{m(\mathcal{F}_0)}$, from two successive measurements of the robot orientation in the global frame: $\theta_{m(\mathcal{F}_0)(t-1)}$ and $\theta_{m(\mathcal{F}_0)(t)}$.

$$\dot{\theta}_{m(\mathcal{F}_0)} = \frac{\theta_{m(\mathcal{F}_0)(t)} - \theta_{m(\mathcal{F}_0)(t-1)}}{T(t) - T(t-1)} \tag{8}$$

where $T(t)$ and $T(t-1)$ are the time of the current measurement of angular speed and the time of the previous measurement of angular speed, respectively.

Using the additive inverse of the angular speed yields a robot rotation compensation speed command.

Finally, the speed command to send to the pan axis of the pan-tilt mechanism is calculated by summing the output of the P-controller and the robot rotation compensation speed command:

$$\begin{aligned}\dot{\theta}_{k(\mathcal{F}_m)(Command)} = \dot{\theta}_{k(\mathcal{F}_m)(P-control)} + \\ \dot{\theta}_{k(\mathcal{F}_m)(Counter-rotation)}\end{aligned} \tag{9}$$

This approach using equation (9) is the one used in this work.

This algorithm requires an estimation of the person position. In case of a loss of tracking, the recovery procedure continues to apply the last pan axis speed command for 500 ms and then setting the pan axis to its neutral position, $\theta_{k(\mathcal{F}_m)} = 0^\circ$, while waiting for a new person position estimation.

D. Path-following algorithm

The principle of the path following algorithm is to make the robot take the same path as the person it follows. It uses the succession of person position estimations in $\mathcal{F}_0$, denoted $Person_{(\mathcal{F}_0)}$, and is calculated from equation (5), to generate a set of points to send to the robot path follower previously described in the robot control section. However, the $Person_{(\mathcal{F}_0)}$ points cannot be directly sent to the robot path follower. They are too noisy when the robot is moving, as described in the experimental results.

Hence the $Person_{(\mathcal{F}_0)}$ points are first filtered:

- Points which imply that the person accelerates faster than 1 g are ignored.
- Points which imply that the person moves faster than $1.5m/s$ are ignored.
- Jitter reduction of $15cm$ radius is applied: if a point is not farther than $15cm$ from the previous point, it is ignored.

Then the path connecting the succession of points is smoothed using a moving average technique of span 5. Finally, as the robot path follower needs a path with points separated by an interval of $2cm$ to properly work, points are interpolated by using uniform cubic B-splines. This also ensures further smoothing of the path. After filtering, smoothing and interpolation, the output point, denoted $Person_{(\mathcal{F}_0)(Filtered)}$ is sent to the robot path follower.

E. Adaptive algorithm

The idea of the adaptive algorithm is to continuously recompute the best path for the robot to go to the person taking into account the obstacles in the environment. Hence, if a shorter way than the path the person took to go to its current position exists, the robot will be able to use it. The optimal path is computed using an implementation of the Karto library which uses the Monte Carlo Localization algorithm [35].

The adaptive algorithm uses the filtered and smoothed person position estimation, $Person_{(\mathcal{F}_0)(Filtered)}$, described in the previous section. Each time a position estimation is received, it is compared to the last position estimation used to generate the robot path. If the distance that separates these two position estimations is superior to 50 cm, a new path using the last position estimation is computed and sent to the robot. This approach is needed in order to limit the frequency of the re-computation of the path which, when too high, saturates the computer and makes the robot oscillate and change its course too often.

IV. RESULTS AND DISCUSSION

A. Path-following algorithm

For each of the 10 trials the robot was able to follow the person until the end of the path (Table I). The average 0.9 loss-of-track event per trial did not affect the performance of the following thanks to the efficiency of the tracking recovery procedure. Figure 3 illustrates this success and shows both

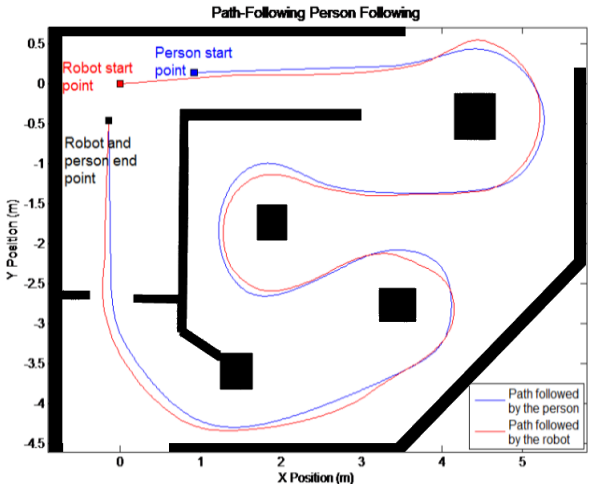

Fig. 3. Person and robot paths of a sample trial of the evaluation of the path-following algorithm.

TABLE I

EXPERIMENTAL RESULTS OF THE EVALUATION OF THE PATH-FOLLOWING ALGORITHM

Path-following algorithm	Average	Max	Min	Standard Deviation
Path-completion success ratio [%]	100	100	100	0
Number of loss-of-track events per trial	0.9	2	0	0.88
Robot path length to person path length ratio [%]	100.5	103.7	97.6	1.6
Path Error [cm]	11.04	40.27	0	7.64

robot and person path close from each other along with the obstacle setups from a typical trial.

The path taken by the person is reproduced accurately with an average path error of 11.04 cm, a standard deviation of 7.34 cm and a maximum error of 40.27 cm (Table I). The agility and accuracy of this method are fully understood when comparing the results with the 40 cm width of our robot. Thanks to this accuracy it is possible to perform person following in an environment with obstacles without the need of detecting and actively avoid the obstacles.

However, when comparing the distance covered by the human and the robot it appears that they are nearly the same. This is due to the principle of this algorithm: the path taken by the person is accurately followed and hence is not optimal; in case of a possible shorter path, it will not be taken by the robot.

B. Adaptive algorithm

In term of path-completion ratio the adaptive algorithm performed as good as the path-following algorithm with a 100% completion for all the trials; and similarly it was not affected by the nearly same average 1.1 loss-of-track event per trial. Figure 4 illustrates this success but shows also how the adaptive algorithm enables the robot to take a shorter path when it can. Over the 10 trials the distance travelled by the robot was 70.9% of the distance travelled by the person, with a maximum of 82.1%, a minimum of 57.6% and a standard deviation of 6.5%.

Hence, the adaptive algorithm presents the advantage of minimizing the distance travelled by the robot compared to

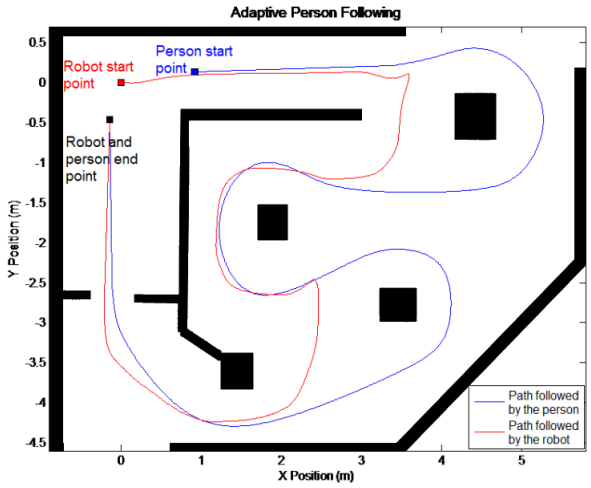

Fig. 4. Person and robot paths of a sample trial of the evaluation of the path-following algorithm.

TABLE II

EXPERIMENTAL RESULTS OF THE EVALUATION OF THE ADAPTIVE ALGORITHM

Adaptive algorithm	Average	Max	Min	Standard Deviation
Path-completion success ratio [%]	100	100	100	0
Number of loss-of-track events per trial	1.1	3	0	1.1
Robot path length to person path length ratio [%]	70.9	82.1	57.6	6.5

the path-following algorithm. However, this requires a pre-build map of the environment.

V. CONCLUSIONS AND FUTURE WORK

Two person-following algorithms that use depth information from a Kinect sensor were presented. Both use the Kinect sensor mounted on a pan-tilt mechanism for 360-angle tracking and implement path generation from a sequence of estimated person's positions. The path following algorithm generates sequentially a path that reproduces the path taken by the person using each new updated position of the person. On the other hand, the adaptive algorithm, recomputes from scratch the shortest path to the person each time the person has moved more than 50 cm. Both person-following algorithms were equally successful in following the person with a 100% path completion ratio. However, the adaptive algorithm minimized the distance travelled by the robot: it travelled in average 29.1% less than the person it followed whereas the path-following algorithm made the robot travel in average 0.5% more. Yet which algorithm is best to use is subject to discussion. The adaptive algorithm minimizes the distance travelled but presents the important constraint of needing a-priori information about the environment (i.e. a map). This can be an advantage in situations where the cost of travel of the robot is expensive or in situations where the maximum speed of the robot is inferior to the walking speed of the person followed.

Future work should focus on path optimization without a-priori information. The case of the robot standing in the way of the person was not investigated in this work. Hence algorithms must be developed to adapt the path of the robot

in order not to block the way of the person when she/he changes suddenly of direction. Furthermore, strategies to recover from complete occlusions from other persons or walls should be improved.

REFERENCES

[1] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," *Computer Vision and Image Understanding*, vol. 104, no. 2-3, pp. 90–126, Nov. 2006.

[2] Z. Jia, A. Balasuriya, and S. Challa, "Vision based data fusion for autonomous vehicles target tracking using interacting multiple dynamic models," *Computer Vision and Image Understanding*, vol. 109, no. 1, pp. 1–21, Jan. 2008.

[3] J. M. Martinez-Otzeta, A. Ibarguren, A. Ansuategi, and L. Susperregi, "Laser Based People Following Behaviour in an Emergency Environment," in *Proceedings of the 2nd International Conference on Intelligent Robotics and Applications (ICIRA '09)*, 2009, pp. 33–42.

[4] Y. Motai, S. Kumar Jha, and D. Kruse, "Human tracking from a mobile agent: Optical flow and Kalman filter arbitration," *Signal Processing: Image Communication*, vol. 27, no. 1, pp. 83–95, Jan. 2012.

[5] M. Kobilarov, G. Sukhatme, J. Hyams, and P. Batavia, "People tracking and following with mobile robot using an omnidirectional camera and a laser," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, no. May. IEEE, 2006, pp. 557–562.

[6] Y. Salih and A. S. Malik, "Comparison of stochastic filtering methods for 3D tracking," *Pattern Recognition*, vol. 44, no. 10-11, pp. 2711–2737, Oct. 2011.

[7] R. Muñoz Salinas, E. Aguirre, and M. García-Silvente, "People detection and tracking using stereo vision and color," *Image and Vision Computing*, vol. 25, no. 6, pp. 995–1007, Jun. 2007.

[8] J. Satake and J. Miura, "Robust stereo-based person detection and tracking for a person following robot," in *ICRA 2009 Workshop on People Detection and Tracking*, no. May, 2009.

[9] R. Muñoz Salinas, R. Medina-Carnicer, F. Madrid-Cuevas, and a. Carmona-Poyato, "Particle filtering with multiple and heterogeneous cameras," *Pattern Recognition*, vol. 43, no. 7, pp. 2390–2405, Jul. 2010.

[10] M. Luber, L. Spinello, and K. O. Arras, "People tracking in RGB-D Data with on-line boosted target models," in *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2011)*. IEEE, Sep. 2011, pp. 3844–3849.

[11] L. A. Schwarz, A. Mkhitaryan, D. Mateus, and N. Navab, "Human skeleton tracking from depth data using geodesic distances and optical flow," *Image and Vision Computing*, vol. 30, no. 3, pp. 217–226, Dec. 2012.

[12] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *The 24th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2011)*. Colorado Springs, CO, USA: IEEE, Jun. 2011, pp. 1297–1304.

[13] F. Hoshino and K. Morioka, "Human following robot based on control of particle distribution with integrated range sensors," in *2011 IEEE/SICE International Symposium on System Integration (SII)*. IEEE, Dec. 2011, pp. 212–217.

[14] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard, "An Evaluation of the RGB-D SLAM System," in *Proceedings of the 2012 IEEE International Conference on Robotics and Automation (ICRA 2012)*, vol. 3, no. c. IEEE, 2012.

[15] M. Camplani and L. Salgado, "Efficient Spatio-Temporal Hole Filling Strategy for Kinect Depth Maps," in *IS&T/SPIE Int. Conf. on 3D Image Processing (3DIP) and Applications*, 2012.

[16] H. Latif, N. Sherkat, and A. Lotfi, "Information acquisition using eye-gaze tracking for person-following with mobile robots," *Information Acquisition*, vol. 06, no. 03, pp. 147–157, 2009.

[17] E. A. Topp and H. I. Christensen, "Tracking for following and passing persons," in *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*. IEEE, 2005, pp. 2321–2327.

[18] T. Yoshimi, M. Nishiyama, T. Sonoura, H. Nakamoto, S. Tokura, H. Sato, F. Ozaki, N. Matsuhira, and H. Mizoguchi, "Development of a Person Following Robot with Vision Based Target Detection," in *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2006)*. IEEE, Oct. 2006, pp. 5286–5291.

[19] T. Sonoura, T. Yoshimi, M. Nishiyama, H. Nakamoto, S. Tokura, and N. Matsuhira, "Person Following Robot with Vision-based and Sensor Fusion Tracking Algorithm," in *Computer Vision*, X. Zhihui, Ed. Vienna, Austria: InTech, 2008, no. November, pp. 519–538.

[20] C. Cauchois, F. de Chaumont, B. Marhic, L. Delahoche, and M. Delafosse, "Robotic assistance: an automatic wheelchair tracking and following functionality by omnidirectional vision," in *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*. IEEE, 2005, pp. 2560–2565.

[21] X. Ma, C. Hu, X. Dai, and K. Qian, "Sensor integration for person tracking and following with mobile robot," in *Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2008)*. IEEE, 2008, pp. 3254–3259.

[22] A. Konigs and D. Schulz, "Fast visual people tracking using a feature-based people detector," in *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2011)*. IEEE, Sep. 2011, pp. 3614–3619.

[23] V. Alvarez-Santos, X. Pardo, R. Iglesias, a. Canedo-Rodriguez, and C. Regueiro, "Feature analysis for human recognition and discrimination: Application to a person-following behaviour in a mobile robot," *Robotics and Autonomous Systems*, vol. 60, no. 8, pp. 1021–1036, Aug. 2012.

[24] T. Germa, F. Lerasle, N. Ouadah, V. Cadenat, and M. Devy, "Vision and RFID-based person tracking in crowds from a mobile robot," in *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2009)*, vol. 3. IEEE, Oct. 2009, pp. 5591–5596.

[25] T. Germa, F. Lerasle, N. Ouadah, and V. Cadenat, "Vision and RFID data fusion for tracking people in crowds by a mobile robot," *Computer Vision and Image Understanding*, vol. 114, no. 6, pp. 641–651, Jun. 2010.

[26] Z. Chen and S. T. Birchfield, "Person following with a mobile robot using binocular feature-based tracking," in *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2007)*. IEEE, Oct. 2007, pp. 815–820.

[27] K. Qian, X. Ma, X. Dai, and F. Fang, "Robotic Etiquette: Socially Acceptable Navigation of Service Robots with Human Motion Pattern Learning and Prediction," *Journal of Bionic Engineering*, vol. 7, no. 2, pp. 150–160, Jun. 2010.

[28] M. Kobilarov, "Cross-entropy motion planning," *The International Journal of Robotics Research*, vol. 31, no. 7, pp. 855–871, May 2012.

[29] G. Dissanayake, S. Huang, Z. Wang, and R. Ranasinghe, "A review of recent developments in Simultaneous Localization and Mapping," in *2011 6th International Conference on Industrial and Information Systems*. IEEE, Aug. 2011, pp. 477–482.

[30] J. Miura, J. Satake, M. Chiba, Y. Ishikawa, K. Kitajima, and H. Masuzawa, "Development of a Person Following Robot and Its Experimental Evaluation," in *Proceedings of the 11th International Conference on Intelligent Autonomous Systems*, Ottawa, Canada, 2010, pp. 89–98.

[31] W. L. D. Lui and R. Jarvis, "A pure vision-based topological SLAM system," *The International Journal of Robotics Research*, vol. 31, no. 4, pp. 403–428, Feb. 2012.

[32] E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey, and K. Konolige, "The Office Marathon: Robust navigation in an indoor office environment," in *Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA 2010)*. Ieee, May 2010, pp. 300–307.

[33] S. Almansa-Valverde, J. C. Castillo, and A. Fernández-Caballero, "Mobile robot map building from time-of-flight camera," *Expert Systems with Applications*, vol. 39, no. 10, pp. 8835–8843, Aug. 2012.

[34] P. Morin and C. Samson, "Motion control of wheeled mobile robots," pp. 799–826, 2008.

[35] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.