ROBOT GOES BACK HOME DESPITE ALL THE PEOPLE

Paloma de la Puente, Diego Rodriguez-Losada, Luis Pedraza and Fernando Matia
DISAM - Universidad Politecnica de Madrid, Jose Gutierrez Abascal 2, Madrid, Spain

Keywords: Mobile Robots Navigation, Localization and Mapping, Reactive Control, Dynamic Points.

Abstract: We have developed a navigation system for a mobile robot that enables it to autonomously return to a start point after completing a route. It works efficiently even in complex, low structured and populated indoor environments. A point-based map of the environment is built as the robot explores new areas; it is employed for localization and obstacle avoidance. Points corresponding to dynamical objects are removed from the map so that they do not affect navigation in a wrong way. The algorithms and results we deem more relevant are explained in the paper.

1 INTRODUCTION

Autonomous navigation around indoor environments is a difficult task for a mobile robot to achieve, especially if there are people passing by frequently. A good start point is presented in (Borenstein et al., 1996) by breaking down the general problem of robot navigation into three questions:"Where am I?", "Where am I going?", "How should I get there?". So, the first (and main) problem encountered when dealing with this issue is the necessity of knowing where the robot is at every moment.As the robot moves, errors in odometry information increase significantly hence making it essential that these data be corrected. Different probabilistic methods for performing this correction using measurements received from stereoceptive sensors (such as laser range-finders or sonars) have thus far been developed, being those capable of building a map at the same time for proper representation of the environment the most effective and popular ones.

As for the second question, the goal to be reached is often defined by the user. It may be given by higher level tasks depending on the particular application.

The last question is challenging as well. A first step is motion control, which is better addressed by means of a closed-loop controller using position feedback (Siegwart and Nourbakhsh, 2004). With a regulator of this kind, path planning comes to computing a sequence of passing points leading to the target. Once a nominal trajectory has been obtained, safe navigation requires reactive control, for the robot should be able to change its behavior if a situation that endangers its mission appears. Regarding this, several strategies have been used in the literature to face obstacle avoidance. Some of the proposed solutions (Feiten et al., 1994), (Yang and Li, 2002) consist of sending special drive and steer velocity commands when an obstacle is detected. The latter and other authors do so through fuzzy control. An interesting and generic approach is an iterative algorithm found in (Lamiraux et al., 2004) for real time deformation of previously collision free paths when operating with nonholonomic robots.

Dynamic objects in the environment bring about further difficulties in map building and reactive control. If the problem is simplified and observed features are represented as permanent in the map, it is still useful for localization purposes but there will be discrepancies with reality. It also increases the map's size unnecessarily and may result in the robot avoiding obstacles which are no longer there. (R.Siegwart et al., 2002) tackle this issue applying the EM algorithm and making use of an a priori map of the environment. They also address other aspects of robot navigation in populated exhibitions, remarking the importance of introducing novel combinations and adaptations of different preexisting approaches. (Hhnel et al., 2003) developed a statistical method to identify measurements corresponding to dynamic objects and perform localization and building of occupancy grid maps, all in the context of the EM algorithm.

In this paper we present a system which allows

de la Puente P., Rodriguez-Losada D., Pedraza L. and Matia F. (2008).
ROBOT GOES BACK HOME DESPITE ALL THE PEOPLE.
In *Proceedings of the Fifth International Conference on Informatics in Control, Automation and Robotics - RA*, pages 208-213
DOI: 10.5220/0001491502080213

the robot to build a point-based map of its static surroundings while being teleoperated from somewhere; this map is afterwards used by the robot to localize itself and make its way back to its original pose, avoiding any obstacles which may be near the initial path. Experiments in highly crowded and cluttered environments have been carried out successfully.

This work is to be used in an ambitious project concerning the autonomous setup of an interactive robot at museums and trade fairs. The robot's name is Urbano and it already counts on a robust localization system based upon geometrical features over SLAM-EKF that originates 2D precise maps in real time (Rodríguez-Losada, 2004). Here we expose an efficient less complex model with the specific demonstrator of the returning home utility. This work has been partially founded by DPI-2004-07907-C02-01.

The paper is organized as follows. Section 2 includes a description of the algorithms corresponding to localization and mapping. In Section 3 we present the movement and reactive control implemented algorithms. Section 4 accounts for experimental results we have obtained. Finally, Section 5 contains our conclusions and future working lines.

2 LOCALIZATION AND MAPPING

2.1 Localization and Map Building

The solution adopted for this problem is a Maximum Incremental Probability algorithm whose foundation is the Extended Kalman Filter(EKF). The system elaborates and continually updates a map built from the observations acquired from the laser measurements.

The state vector used is defined as the robot's global pose, $[x_R, y_R, \theta_R]^T$, whereas the odometry measurements (incremental, so referred to the robot's local coordinate system) represent the system inputs, $\vec{u} = [u_x, u_y, \theta_u]^T$. Time subscripts are omitted in the latter so as to simplify notation; measurements obtained last are always the ones considered. According to this, the state equation is put as:

$$\vec{x}_{R_k} = \vec{x}_{R_{k-1}} \oplus \vec{u} = f(\vec{x}_{R_{k-1}}, \vec{u}) \tag{1}$$

where $\oplus$ represents the composition of relative transformations.Odometry measurements can be modelled as a gaussian variable, $\vec{u}_k \sim N(\hat{\vec{u}}_k, Q)$.

2.1.1 Predictor Equations

Applying the definition of the $\oplus$ operator, the predicted state, $\tilde{x}_{R_k}$, will be given by:

$$\begin{pmatrix} \tilde{x}_{R_k} \\ \tilde{y}_{R_k} \\ \tilde{\theta}_{R_k} \end{pmatrix} = \begin{pmatrix} \hat{x}_{R_{k-1}} + u_x cos\hat{\theta}_{R_{k-1}} - u_y sin\hat{\theta}_{R_{k-1}} \\ \hat{y}_{R_{k-1}} + u_x sin\hat{\theta}_{R_{k-1}} + u_y cos\hat{\theta}_{R_{k-1}} \\ \hat{\theta}_{R_{k-1}} + \theta_u \end{pmatrix} \tag{2}$$

And the state's covariance prediction, $\tilde{P}_k$, is computed from:

$$\tilde{P}_k = F_x \hat{P}_{k-1} F_x^T + F_u Q F_u^T \tag{3}$$

being $F_x = \frac{\delta f}{\delta \vec{x}} |_{\tilde{\vec{x}}_{R_k}}$, $F_u = \frac{\delta f}{\delta \vec{u}} |_{\tilde{\vec{x}}_{R_k}}$ (we use the best state estimation obtained up to now)

The algorithm is initiated with $\hat{x}_0 = 0, \hat{P}_0 = 0$.

2.1.2 Corrector Equations and Data Association

Ideally, each observation $\vec{o}_i = [o_{ix}, o_{iy}]^T$ (or laser measurement received) is implicitly related to the previous estimation of the state by means of composition with it and comparison to the corresponding $\vec{l}_j = [x_{lj}, y_{lj}]^T$ point in the map. The resultant expression is known as the innovation of observation i:

$$\vec{h_{ij}} = \vec{x_R} \oplus \vec{o}_i - \vec{l}_j = h(\vec{x}_R, \vec{o}_i, \vec{l}_j) = \vec{0} \tag{4}$$

which combined with $\oplus$ definition is the same as:

$$\vec{h_{ij}} = \begin{pmatrix} -x_{lj} + \tilde{x}_R + o_{ix} cos\tilde{\theta}_R - o_{iy} sin\tilde{\theta}_R \\ -y_{lj} + \tilde{y}_R + o_{ix} sin\tilde{\theta}_R + o_{iy} cos\tilde{\theta}_R \end{pmatrix} = \vec{0} \tag{5}$$

If we denote $H_{x_{ij_k}} = \frac{\delta \vec{h_{ij}}}{\delta \vec{x}} |_{\tilde{\vec{x}}_{R_k}, \vec{o}_i}$, $H_{z_{ij_k}} = \frac{\delta \vec{h_{ij}}}{\delta \vec{o_i}} |_{\tilde{\vec{x}}_k, \vec{o}_i}$ for every iteration k, the covariance of each innovation is given by:

$$S_{ij_k} = H_{x_{ij_k}} \tilde{P}_k H_{x_{ij_k}}^T + H_{z_{ij_k}} R H_{z_{ij_k}}^T \tag{6}$$

where R is the covariance of laser measurements.

We have followed the *Nearest Neighbor* strategy to pair each observation with its correspondent map point. Computation of the Mahalanobis distance for innovation h_{ij_k} once we have got its covariance matrix S_k will let us select that map point which minimizes such distance. If Mahalanobis test is passed for that association, then it is taken into account at the correction step; otherwise it will be added as a new point of the map vector. Among all the $h_{ij}, H_{x_{ij}}$ and H_{zij} matrices obtained for an observation at a certain iteration, only those corresponding to the map point, if any, associated to it will be kept to correct the estimation. They will be denoted $h_{i_{min}}, H_{xi_{min}}, H_{zi_{min}}$. As more associations are made, there are more measurements to

be used. This is contemplated by filling other matrices containing joint information from all of them:

$$\vec{h} = [h_{1_{min}}, \ldots, h_{t_{min}}]^T \quad (7)$$

$$H_x = [H_{x1_{min}}, \ldots, H_{xt_{min}}]^T \quad (8)$$

$$H_z = \begin{pmatrix} H_{z1_{min}} & & \\ & \ddots & \\ & & h_{t_{min}} \end{pmatrix} \quad (9)$$

where t is the total number of associations. Now we get the global S matrix and the Kalman gain from

$$S = H_x \tilde{P} H_x^T + H_z R H_z^T \quad (10)$$

$$K = \tilde{P} H_x^T S^{-1} \quad (11)$$

The corrected values are updated from this information and the prediction values obtained from 2 and 3:

$$\hat{\vec{x}} = \tilde{\vec{x}} - Kh; \quad (12)$$

$$\hat{P} = (I - KH_x)\tilde{P} \quad (13)$$

2.2 Removal of Dynamic Points

The algorithm presented above incorporates to the map every observed point which cannot be associated to a pre-existing one. If we stick to it, points corresponding to a dynamic object detected at a given moment are included in the map when they are first seen (just as any other object is) and nothing is done to avoid them being there forever. Here we propose a method that removes map points far from being observed at a given moment, before the correction and update of the map take place.

Firstly, an evolvent polygon of the laser measurements is constructed in a recursive way. The general procedure is summarized in the following lines:

- A segment from the first measurement to the last is taken.
- The furthest observation from that segment is sought among the others.
- If the separation between the segment and that observation is high enough, the process is repeated between the first observation and the one selected and then between this observation and the last one.
- When for one segment there is no observation at a greater distance than a threshold, its limits are stored in a vector containing the polygon vertices.

Some other simplifications are made in order not to generate too many vertices. Map points that remain quite inside the polygon cannot correspond to currently observed objects because if so, they would either have been included as polygon vertices or they would be very close to them (due to the threshold and the simplifications mentioned above).

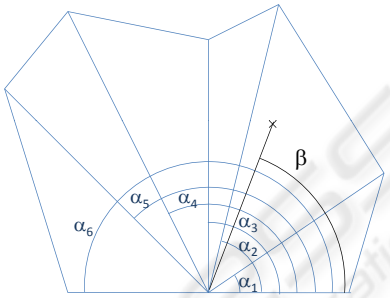

Figure 1: Angles to be measured for seeing if a point is inside an unconvex polygon.

To find out whether a map point is inside the polygon, we compute a series of angles ($\alpha_1 \ldots \alpha_v$ in figure 1) as soon as the vertices extraction is over. Then, for every point in the map which is not too near the polygon's border we determine its coordinates in the laser reference system and then compute the value of the β angle represented in the same figure. Selecting those alphas β lies in between (α_1 and α_2 in the figure's case) we obtain the corresponding vertices and use their distance to the robot to establish if that map point should be erased. We use a conservative criterion that leaves outside the map only those points at a smaller distance than the closest of both vertices. This prevents any map points being removed incorrectly.

The computational cost of creating the polygon and finding the values of the needed angles is upper bounded, since the number of observations provided by the laser is constant. Notice that only one angle has to be computed for each point in the map, which makes it a not very cumbersome algorithm.

3 MOTION CONTROL

3.1 Motion Controller

To get the robot moving from one point to another we have employed gain scheduling, implementing a controller in agreement to the divide and conquer approach. The error is a linear combination of three angles. In first place is the difference between the present orientation of the robot and the one it should

have to look straight ahead to the next goal. The second angle is that one the robot should turn to have the same global orientation as the current trajectory segment it is at. The third angle is similar to the second although in this case it is the next trajectory segment that is considered. Weights of 0.3, 0.2 and 0.5 have been used to produce smooth anticipative movement following a given trajectory. We define several intervals for that global error and vary a constant value of drive velocity and the gain of a proportional controller for the steer velocity within each of them.

In the experiments described here, the given trajectory is obtained by saving the robot's pose every time it moves a constant distance.

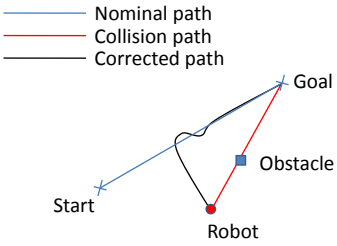

Figure 2: Possible paths if the robot is not near the defined trajectory.

If the robot were to be quite separated from the defined trajectory, this controller would take it to the goal not across the nominal path but across one in which there might be an obstacle(figure 2).

The corrected path is obtained by forcing the robot to approach the nominal path before heading for the target. For this purpose we have included control laws that regulate angles δ_1 or δ_2 in figure 3 depending on which side of the path the robot is at. We use angles in $[-\pi,\pi]$ along all our work, which is what the `ForceInRange` function has been defined for. These laws are employed instead of the initial ones only when the robot is far enough from the defined path.

When the robot is told to go back, it turns 180 at first and then come into action the rules that have just been commented. At a certain distance from home, the robot begins to slow down until it reaches its final pose. Then it turns to adopt the orientation it had when it abandoned home.

3.2 Path Deformation

To get a trajectory free of obstacles we displace the points of the nominal trajectory which are affected by objects detected in the environment.To begin with, we make sure that consecutive points in the initial trajectory are at a short enough distance, let it be 0.1m.For each point i we follow the same steps. At first, a point

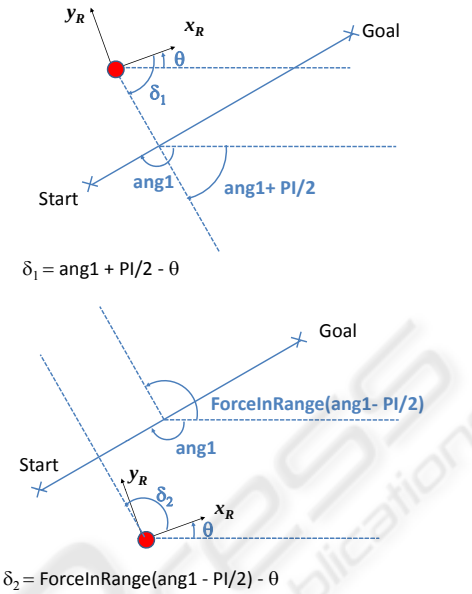

Figure 3: Angles through which control is executed.

p is defined on the normal to the segment joining that point of the trajectory and the next one so that it is 1m away from the former. The vector going from point i to point p will be referred to as v_1. We define a vector v_2 with origin at the considered trajectory point and end at the subsequent obstacles detected. The orthogonal projection of this vector v_2 onto v_1 determines point p_2. Its distance to the trajectory point i is d, which may be positive or negative if the obstacle is at one side or another of the trajectory.

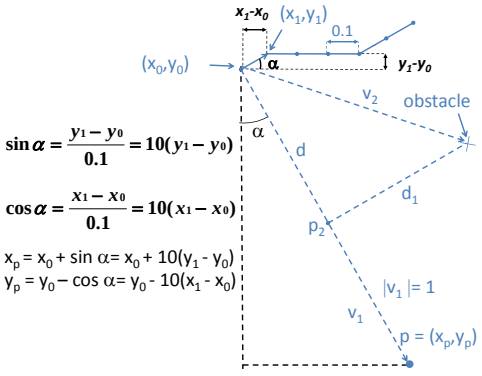

Figure 4: Definition of point p and other magnitudes for the trajectory point $i = 0$.

The distance from p_2 to the obstacle will be called d_1. All these magnitudes are represented in figure 4.Only those objects resulting in a small enough value of d_1 take part in the deformation relative to that

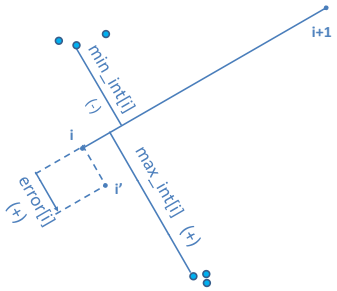

Figure 5: Displacement applied to each point of the nominal trajectory.

path point. Correcting d by considering the influence of the robot radius and afterwards obtaining its minimum and maximum values, an area of permitted movement can be determined. We will call these two distances `min_int[i]` and `max_int[i]`, respectively.

They are initialized with the maximum displacement allowed, for the case of no presence of obstacles and other similar situations. Their mean will be the displacement,`error [i]`, to apply to that path point in order to leave it at an intermediate distance between the limits found for each of both sides (see figure 5). To generate a smoother deformation we use the average of that error value and the previous and next ones:

$$\text{error2}[i] = (\text{error}[i-1] + \text{error}[i] + \text{error}[i+1])/3$$

4 EXPERIMENTS AND RESULTS

The system has been tested in different environments, using simulators (one we have developed ourselves and also MobileSim, by Activemedia Robotics) and real robot data from text files at the beginning, and the robot Urbano afterwards. Urbano stands on a B21r platform and has a laser SICK LMS200 mounted on top. The architecture of the system is the one in fig. 6.

The most significant experiments we have conducted took place at our laboratory, which has very narrow areas and people often coming to and fro. The

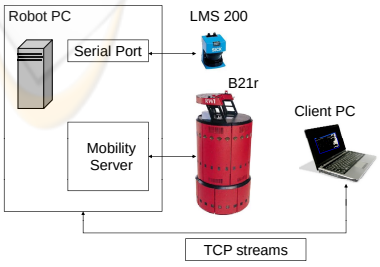

Figure 6: Urbano's distribution schema.

robot was teleoperated from one end of the laboratory to the other, finding a lot of visitors while building the map.When we had brought it to the main entrance, we told him to return home. Since that moment all of its behavior is absolutely autonomous. In the way back, other people were seen and Urbano was able to avoid them. Some other slight corrections were made in order to follow a path which got apart from close obstacles. One of the maps we obtained is the one in figure 7. The green path is the odometry corresponding to the teleoperation mode. The pink path is the one representing dead-reckoning in the way back. The blue trajectory is the correction for the first stage of the experiment, while the red one is the correction obtained when the robot was coming back home. Cyan points are those which were removed from the map. Most of them were clearly identified with people's successive positions when walking in front of the robot.

Another experiment was performed in the building where lessons are imparted at a time in which there were plenty of students coming out from their classes. The resultant map and the actual environment are shown in figure 8.

5 CONCLUSIONS

The system presented here has two main components, one having to do with localization and map building and one related to control and path deformation.

The implemented localization and mapping algorithm is a Maximum Incremental Probability method based on the Extended Kalman Filter (EKF). Its main drawback is the fact that it does not consider the uncertainty in the map itself, but it allows for a higher degree of simplicity and has proven an appropriate behavior on the experimental conditions of a wide variety of tests apart from those exposed in this paper (using data obtained by several real robots in different indoor environments, some of them having large odometry errors). The elimination of points makes more realistic and reliable maps which represent the last structure observed by the robot. If the same map were used in another experiment, instead of making the robot build a new one in real time,it would get adapted to the configuration of the environment at that moment. This strategy also prevents mistakes in data association and reduces maps' size when possible.

The developed control module enables the robot to achieve a final target without hitting any obstacles. A high level technique keeps the path to be followed away from the objects detected by the laser, and the controller does not let the robot get far from this collision free path. Obstacles at lower height than the

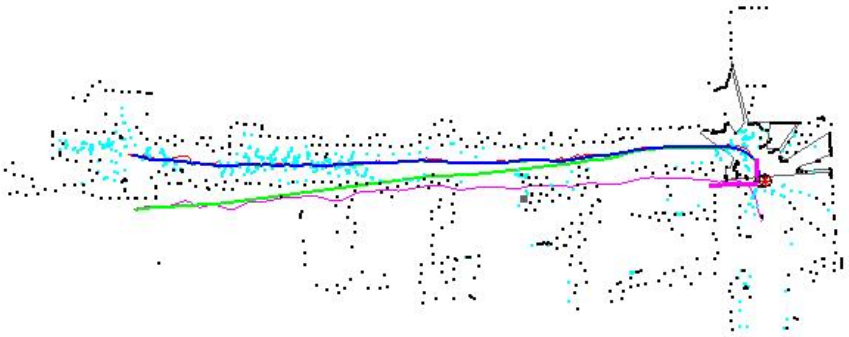

Figure 7: Real time built map of our laboratory.

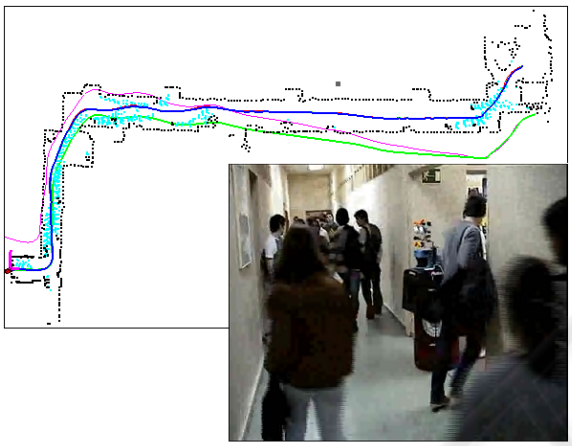

Figure 8: Real time built map of an area plenty of students.

laser sensor cannot be seen. Consequently, not only will these obstacles be left out of the map and cause it to be less accurate but, what is much worse, they will not be considered in path deformation either. As a precaution, Urbano has got some ultrasound sensors at an intermediate height which make the robot stop if they detect it is about to crash, but it might not be enough under some circumstances. This caveat gives rise to one of our next working lines.

6 FUTURE WORK

As previously mentioned, we now orientate our work towards the integration of knowledge models for the autonomous setup of an interactive robot at museums and trade fairs. The reference of this project, called Robonauta, is:DPI2007-66846-c02-01.

To achieve this important objective, some lines related to this work and to Urbano's navigation in general are:

- Extension of geometrical models. Employing a wrist that can precisely position the 2D scanner in different angles for data acquisition, we aim at constructing robust algorithms to create 3D maps which will make the robot's navigation safer.
- Improvements in control, so that the robot can move faster and follow its path more accurately.
- Path definition by means of people tracking or by means of voice commands spoken to the robot.
- Incorporation of new conducts which will enable the robot to follow a learning process by itself.

REFERENCES

Borenstein, J., Everett, H., and L.Feng (1996). *Navigating Mobile Robots: Systems and Techniques*. A.K Peters.

Feiten, W., R.Bauer, and Lawitzky, G. (1994). Robust Obstacle Avoidance in Unknown and Cramped Environments. In *IEEE Int. Conf. Robotics and Automation*.

Hhnel, D., R.Triebel, W.Burgard, and S.Thrun (2003). Map Building with Mobile Robots in Dynamic Environments. In *IEEE Int. Conf. Robotics and Automation*.

Lamiraux, F., Bonnafous, D., and O.Lefebvre (2004). Reactive Path Deformation for Nonholonomic Mobile Robots. *IEEE Transactions on Robotics*.

Rodríguez-Losada, D. (2004). *SLAM Geométrico en Tiempo Real para Robots Móviles en Interiores basado en EKF*. PhD thesis, ETSII-Universidad Politcnica de Madrid.

R.Siegwart, R.Philippsen, and B.Jensen (2002). http://robotics.epfl.ch.

Siegwart, R. and Nourbakhsh, I. (2004). *Introduction to Autonomous Mobile Robots*. MIT Press.

Yang, S. and Li, H. (2002). An Autonomous Mobile Robot with Fuzzy Obstacle Avoidance Behaviors and a Visual Landmark Recognition System. In 7^{th} *ICARCV*.