Topological Mobile Robot Navigation using Artificial Landmarks

David Esparza Jesus Savage
Biorobotics Laboratory
Engineering School
Universidad Nacional Autonoma de Mexico

***Abstract*—One of the main goals of a household robot is to navigate in an environment in a safe way. Some prior work has addressed this assumption by using dead reckoning in combination with reactive behaviours to interact with unknown dynamic environment. This paper presents a mobile robot navigation system that uses a topological landmark based representation of the environment from which it is possible to find paths using the Dijkstra algorithm. In this way the robot is able to achieve its goal while avoiding obstacles using a reactive navigation approach based on the coordination of state machines.**

***Keywords*—Path planning, Mobile robot, Reactive Navigation, Landmark**

I. Introduction

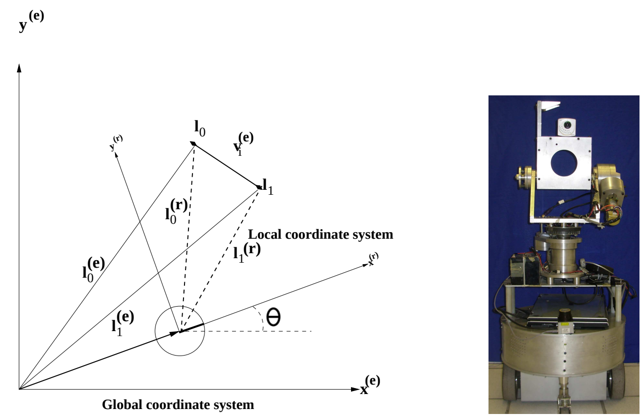
Mobile robot navigation is one of the main problems that a household robot must solve. There are two methods to face this problem, the relative localization and absolute localization. In relative localization, the robot must be able to know its initial position, and using odometry that monitors the wheels, can determine the robot's position at once. On the other hand, absolute localization consist in detecting recognizable characteristics of the environment called natural landmarks, using different types of sensors.

Robot navigation based on pure odometry is unlikely to succeed because it gives an inaccurate localization as consequence of wheel slippage, where the calculated robot position drifts over time. One way to overcome this problem is using trilateration for robot localization [1] using either natural or artificial landmarks located in the environment. Natural landmarks are acquired using different sensors which captures the shape of the environment, but as explained in [2] the main problem in natural landmark navigation is to detect and match characteristic features from sensory inputs. Alternatively, there exist artificial landmarks, which are positioned in the environment with the purpose that the robot can localize itself. The artificial landmarks are designed for optimal contrast and their exact size and shape are known in advance [2]. In this work we investigate the use of artificial landmarks composed of geometrical patterns placed in an indoor environment in order to enable a household robot to navigate and localize itself.

A. Problem definition

The main goal of this work is to combine reactive behaviours to enable a mobile robot to navigate autonomously without collisions towards a target, using a sequence of visual landmarks which correspond to the best route between two distinct places in the environment.

II. Overview of the method

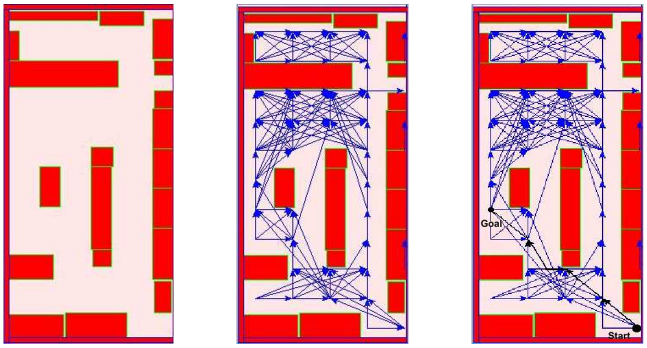


(a) (b)

Fig. 1: The coordinate system of the environment, and the robot's coordinate system in (a) and Mobile Robot TPR8 in (b).

This section illustrates the representation of the environment, which consist of polygons: the borders of the navigational space are represented by a bounding box, and the occupied space of objects and furniture is represented with rectangles inside the boundary of the bounding box. As shown in Fig. 1a the robot frame of reference is fixed to the center of the robot and is denoted by (x^r, y^r), while the global frame of reference is denoted by (x^e, y^e). θ_r is the angle between the x^r axis and the x^e axis. The direction and position of the robot in the global frame of reference is denoted by $q = [x_r, y_r, \theta_r]^T$. The global position of the robot is determined by using the local coordinates and the robot's angle θ_r.

The TPR8 robot [3] was the wheeled mobile robot used in our navigation experiments. This mobile robot consist of a vehicle with two driving wheels mounted on the same axis, and a front and rear free wheel. The motion and orientation are achieved by independent dc motors providing the necessary torques to the center wheels. The control elements consist of a microcontroller Motorola HC12. It is equipped with a laser range finder to detect obstacles, a webcam used to detect



(a) Metric map of the environment

(b) Topological map of the environment

(c) Topological path

Fig. 2: Robot navigation scenario: UNAM Biorobotics Laboratory. In (a) the metric map. In (b) topological map generated by vectorial quantization, in (c) a topological path from the laboratory entrance to the center of the lab.

artificial landmarks, and encoders to perform dead-reckoning in open loop, i.e. calculating the robot's position relying on a previously determined position. A view of the TPR8 robot is presented in Fig. 1b.

III. Path planning using Dijkstra method

In our work, the environment is discretized into a graph where the nodes of the network correspond to the centroids of the free space regions, while the edges between two different nodes represent the navigable path between two adjacent regions in the environment. The centroids of the regions have been estimated using the method proposed by Savage et al. [4], where they use quantization of sonar readings along a set of free regions defined a priori in the environment. In Fig. 2 is shown an example of a topological map of the UNAM Biorobotics laboratory. The undirected graph structure is used to find the shortest route between two different nodes using the Dijkstra algorithm [5]. This graph is recorded in a *connectivity table* C_l whose main purpose, is to store connectivity information, such as vertex coordinates and the distances between each pair of nodes in the graph.

In order to apply the Dijkstra algorithm we use the connectivity table C_l which takes into consideration the position of the robot (the start node) $q_i = (x_i, y_i, \theta_i)$, and the final position (the goal node) $q_m = (x_m, y_m, \theta_m)$. The output of this method correspond to the shortest collision-free path connecting the start node and the goal node and is represented by a vector of consecutive sub-goals $Q_r = \{q_i, ..., q_m | q_j \in C_l\}$.

A. Path Following

Once the path to the goal has been found, the robot must navigate it, following the sequence of sub goals between the robot's initial position and the goal position as shown below:

1) Provide the sub-goal position x_t, y_t.
2) Estimate the current robot position x_r, y_r using dead reckoning.
3) Calculate the resultant virtual force that pulls the robot in the sub-goal direction with a magnitud m and orientation θ:

$$\theta = tan^{-1}\frac{(x_t - x_r)}{(y_t - y_r)}; \;\; if \;\; x_r \neq x_t$$

$$m = \sqrt{(x_t - x_r)^2 + (y_t - y_r)^2}$$

4) The robot executes a move instruction with the magnitude and direction specified in the previous step.

We adopt a solution based on reactive behaviours which enable the robot to follow a path between a given initial position and a given goal position, which is composed of sub-goals identified by artificial located landmarks.

B. Reactive navigation

Without an obstacle avoidance system, the robot simply heads towards an identified goal following a straight-line trajectory. However, in order to avoid collision risks, we have developed a reactive navigation approach, based on a sensing-path following to reach the sub-goals along the path, one by one [6].

Following this approach, the robot is considered as a charged particle navigating through a magnetic field. The behaviour exhibited by the particle will depend on the combination of the two potential forces:

- Attractive force: a sub-goal
- Repulsive potentials: obstacles

The goal position is considered as an attractive potential and is negative charged. In the same way, obstacles have repulsive positive charge. If the robot approaches the obstacles, a repulsive force will act on it, pushing it away from the obstacles. Combining potential fields produces the robot's motion from a start position to a goal position, while avoiding collisions with obstacles as follows:

- Estimate the potential field gradient $U(q)$ over the robot using the vision data and proximity sensor readings.
- Determine the motor force vector $F(q)$ which is used to move the robot in the gradient direction.
- Set the adequate instructions to move the robot according to the induced force and direction specified by the vector $F(q)$.

The continuous iteration of the above cycle manages to handle a reactive navigation, modifying the robot's trajectory as result of the summation of attractive and repulsive potential fields, generated by the sensor readings and the attractive goal coordinate.

C. Obstacles and goals

We follow the electrostatic approximation where the magnitude of the virtual force is proportional to the square of the distance between the robot and the surrounding objects, and between the robot and the sub-goals in the Dijkstra path. We can express the virtual force over the robot as consequence of an obstacle o as shown below:

$$F_{o(rep)} = K\frac{Qq_o}{d^2}\hat{u} \quad (1)$$

where,

$F_{o(rep)} =$ Repulsive potential derived from the magnitude and direction of o

$K =$ The constant of proportionality

$Q =$ "Charge" associated with the robot

$q_o =$ "Charge" associated with the obstacle

$d = \mid x_r - x_o \mid$

$\hat{u} =$ Unit vector in the direction of the obstacles o

While the vector $\hat{u}$ is given by:

$$\hat{u} = \frac{x_r - x_o}{\mid x_r - x_o \mid} \quad (2)$$

Where,

$x_r =$ Vector position of the robot

$x_o =$ Vector position of the obstacle

For convenience we assume that q_o and Q have the value 1, allowing to modify the constant K. The proximity readings are acquired from the laser range finder and the landmarks distance is acquired from the ARToolKit pattern recognition system. We have limited the robot motion to a small constant distance toward the goal direction in order to avoid obstacle collision:

$$F_{atr} = A_t\hat{v} \quad (3)$$

where,

$F_{atr} =$ Attractive potential field associated to the sub-goal t

$A_t =$ Proportionality constant

And the unit vector $\hat{v}$ pointing from the robot position to the the target t is:

$$\hat{v} = \frac{x_r - x_t}{\mid x_r - x_t \mid} \quad (4)$$

Where,

$x_t =$ Cartesian coordinates of the sub-goal t

The total virtual force acting on the robot is calculated by the sum of all the obstacles and sub-goal forces. In this way, if $O = \{\textit{All the detected obstacles}\}$, then the total force F_{net} acting on the robot, is given by the sum of attractive and repulsive forces as shown below:

$$F_{net} = \sum_{\forall_o \in O} F_{o(rep)} + F_{atr} \quad (5)$$

We can express the resultant force F_{net} in terms of its components x, y:

$$\theta = \tan^{-1}\left(\frac{F_y}{F_x}\right) \quad (6)$$

Now we have the robot direction of movement, which comprises the direction of F_{net} force, where the equation 6 defines the steering angle that the robot should follow.

D. Solutions to overcome the limitations of potential fields in robot control

The inherent problem to potential fields occur when the robot is trapped in a local minima which typically occurs when the robot runs into a dead end (e.g. inside a U-shaped obstacle) [7]. We solve trap situations with reactive behaviours that follow a random trajectory towards a free space. In order to have a complete solution using potential fields, a new path to the goal can be calculated at any time when the robot has recovered from trap situations.

E. Landmark detection

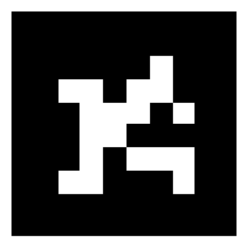


Fig. 3: ARToolKit Marker

We detect ARToolkit markers [8] in the video images. ARToolkit detects markers composed of geometric patterns printed over a flat surface as it is shown in Fig. 3. The resulting measurements give us the pose of the marker with respect to the camera. The ARToolkit markers were placed throughout the free space of the environment to serve as landmarks. Some of the markers were fixed to the walls, while others were held by moveable traffic cones which facilitate their movement during the experiments.

F. Visual navigation based on state machines

By using state machines approach, we want to control a sequence of behaviours for mobile robot exploration and navigation. In the main loop of the system we evaluate a sequence of state conditions and events to determine w1hich state machine must be activated. This is shown in Fig. 4.

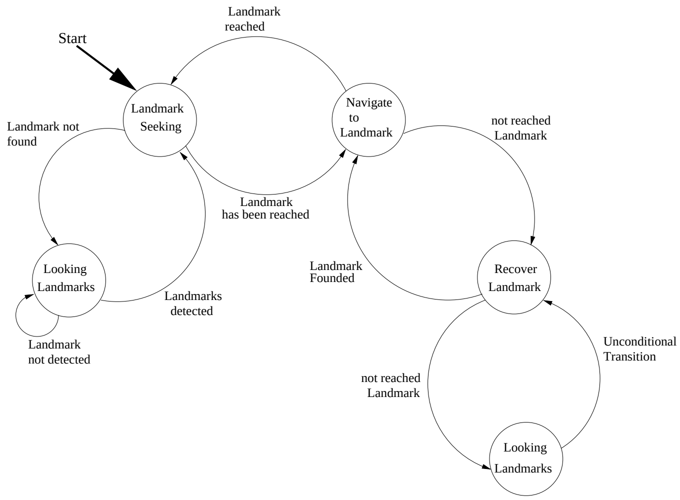


Fig. 4: Visual markers navigation based on state machines

State 1: Landmark-Seeking

Action: The robot executes twelve 30 degree turns to the left (i.e. 360 degrees) looking for a specific *id landmark*.
Inputs: id landmark.
Transitions: If the *id landmark* has been found, Then the transition to the behaviour *Navigate-to-landmark* is performed.

State 2: Looking-Landmarks

Action: Wandering through the environment looking for any landmark.
Transitions: If no one landmark has been found, Then the robot remains in the state *Looking-landmarks*.
If the event *Landmarks detected* is activated, Then the transition to one of the previous behaviours *Landmark-Seeking* or *Recover-Landmark* is executed.

State 3: Navigate-to-Landmark

Action: Points the robot towards the landmark center and moves it to a fixed distance in the landmark direction.
Inputs: id landmark.
Transitions: If the event *Not reached landmark* is activated, Then the transition to the behaviour *Recover-Landmark* is executed.
If the event *Landmark has been reached* is activated, Then the transition to the behaviour *Initial-state* is performed.

State 4: Recover-Landmark

Action: The robot makes a 45 degree turn to the left in 15 degree steps looking for a specific id landmark. If the landmark has not been found, then it goes back to the initial pose and makes a 45 degree turn to the right in 15 degree steps looking for the same id landmark.

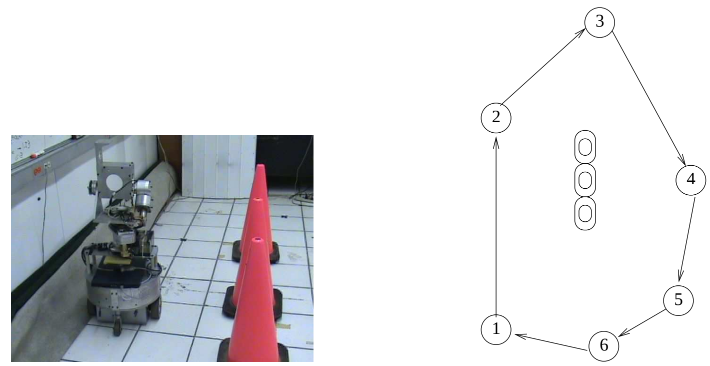
Inputs: id landmark.

Transitions: If the event *Landmark not found* is activated, Then the transition to the behaviour *Landmark-Seeking* is performed.

If the event *Landmark found* is activated, Then the transition to the behaviour *Navigate-to-landmark* is performed.

IV. Experimental results

To verify the performance of our navigation method, we performed a series of experiments with static landmarks in our office environment. Fig. 2 shows a metric representation of the testing environment, the size of which is 5.0 metres by 7.0 metres.



(a) (b)

Fig. 5: The environment setup for dead reckoning navigation. The TPR8 navigating inside the laboratory of biorobotics in (a) and a path which the robot is following in (b).

Dead reckoning navigation

The goal of this test was to evaluate the robot navigation by following a semicircular trajectory to measure the position error at the end of the movement with respect to the goal position as is shown in Fig. 5. In 5 from 13 trials the robot reached the goal with an average error of 4.7 decimetres, while in the remaining trials the robot collided with the surrounding obstacles.

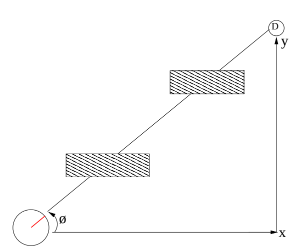


Fig. 6: Obstacle avoidance with potential fields: two rigid obstacles were placed diagonally on the floor

Reactive Navigation

In this test we evaluate the obstacle avoidance navigation using potential fields and proximity distance readings. We conducted two tests of 17 trials where the mobile robot received the mission to reach a given goal position from a given start position, where two objects were placed in the way of the robot as it is shown in Fig 5.

The following parameters were used in order to make the mobile robot to be attracted towards a goal position, while being pushed away from the obstacles in the *potential fields* frame. For the first phase, we hold the constant $K < 0.3$, set the constant of attraction $A = 300$, which modify the strength of the attractive field, and set a security radius to $r_{sec} = 9dm$. As result in 7 of the the 17 trials the robot collided with the surrounding obstacles.

In order to improve the robot obstacle avoidance behaviour, we modified the repulsive constant with $K = 0.5$ and the attractive constant with $A = 240$ resulting in 14 of the 17 trials without collisions, and the robot stopped $5dm$ away from the goal.

(a) (b)

Fig. 7: (a) Visual landmark navigation scenario. (b) Front view of the Lab. room (c) View from the back of the Lab. room

Landmark navigation: In this test we have applied the visual landmarks navigation method combined with reactive behaviours in order improves its navigation results. To build the topological map, we placed 6 landmarks along the environment as is shown in Fig. 7. The position and distance between each pair of nodes were stored in a connectivity table, in order to be able to execute a navigational plan using the landmark based topological map as described below:

- Get the path to follow using the landmark based topological map and the Dijkstra method.
  Find the path between the closest visible landmark to the robot and the goal place, using the landmark based topological map and the Dijkstra method and save the path founded into a vector of sub-goals.
- Using the on board video camera, the robot seeks for the next landmark to follow, which is provided by the path planning module.
- When the vision module detects the goal landmark, the navigation module gives the movement instructions to steer the robot in the direction of the landmark. The robot then will approach to the sub-goal in a reactive way.
- This process ends when then the robot has reached 0.4m away from the current landmark position, then the robot waits for the next sub-goal to follow.
- If the robot has reached the last sub-goal in the path, it stops and notifies it to the user. Otherwise the robot gets the next sub-goal in the path.

In 19 of 25 tests, the robot reached the goal without collision, but it had problems in visual landmark detection when the distance between the robot and the landmark was more than 4 meters which caused the robot to lose the path sequence towards its goal.

V. Discussion

This paper evaluated the reactive based navigation scheme using artificial landmarks in order to determine a navigable path between two different locations in an indoor environment. The system is able to acquire proximity information from a laser range finder to develop a reactive navigation, and path recognition using geometric landmarks located along the the free space in the environment. The trajectories returned by our Dijkstra Path planning module tend to be non smooth, but using the potential fields method, it was possible to drive the robot in a smooth and secure way depending on the potential field parameters assigned.

Future work will focus on adding contextual information to the topological representation of the environment, in order to enable a household robot to perform manipulation tasks in the environment, developing a human-robot interaction and improving the robot localization task.

Acknowledgment

This work was supported by PAPIIT-DGAPA UNAM under Grant IN-107609.

References

[1] L. Feng, J. Borenstein, and B. Everett, "Where am i? sensors and methods for autonomous mobile robot localization." The University of Michigan, Technical Report, 1994, disponible via anonymous FTP desde ftp.eecs.umich.edu/f/people/johannb.

[2] S. Thrun, "Learning metric-topological maps for indoor mobile robot navigation." *Artificial Intelligence*, 1998.

[3] L. Romero, "Robot movil rc-mobile-base-1 + rc-pan-tilt-1." Universidad de Michoacán, Atzimba 168, Centro, C.P. 60250, Paracho Michoacán, México Tel.+52 (423) 52511039., Tech. Rep., 2005.

[4] J. Savage, E. Marquez, and F. Lepe, "Hidden markov models and vector quantization for mobile robot localization," *Robotics and Applications*, October 2005.

[5] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson, *Introduction to Algorithms*, 2nd ed. McGraw-Hill Higher Education, 2001.

[6] M. A. Goodrich, "1 introduction potential fields tutorial."

[7] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," Sacramento, États-Unis, 1991, pp. 1398–1404.

[8] H. Kato, K. Tachibana, M. Tanabe, T. Nakajima, and Y. Fukuda, "Magiccup: a tangible interface for virtual objects manipulation in table-top augmented reality," in *Proc. IEEE Int. Augmented Reality Toolkit Workshop*, 2003, pp. 75–76.