A novel approach for the global localization problem.

Alfredo Toriz* , Abraham Sánchez** and Maria A. Osorio**

ABSTRACT

This paper describes a simultaneous planning localization and mapping (SPLAM) methodology focussed on the global localization problem, where the robot explores the environment efficiently and also considers the requisites of the simultaneous localization and mapping algorithm. The method is based on the randomized incremental generation of a data structure called Sensor-based Random Tree, which represents a roadmap of the explored area with an associated safe region. A continuous localization procedure based on B-Splines features of the safe region is integrated in the scheme.

RESUMEN

Este artículo describe una metodología de planificación, localización y mapeo simultáneos enfocada en el problema de localización global, el robot explora el ambiente eficientemente y también considera los requisitos de un algoritmo de localización y mapeo simultáneos. El método está basado en la generación aleatoria incremental de una estructura de datos llamada árbol aleatorio basado en sensores, la cual representa un mapa de caminos del área explorada con su región segura asociada. Un procedimiento de localización continuo basado en características B-splines de la región segura se integró en el esquema.

Recibido: 6 de Enero de 2012
Aceptado: 14 de Febrero de 2012

INTRODUCTION

SLAM (Simultaneous Localization And Mapping) a-pproaches are used simultaneously with classic exploration algorithms [11]. However, the result obtained by the SLAM algorithm strongly depends on the trajectories performed by the robots. Classic exploration algorithms do not take localization uncertainty into account, when the robots travel through unknown environments, the uncertainty over their position increases and the construction of the map becomes difficult. Consequently, the result could be a useless and inaccurate map. With the integrated exploration or SPLAM (simultaneous planning localization and mapping), the robot explores the environment efficiently and also considers the requisites of the SLAM algorithm [5, 8].

An integrated exploration method is introduced in [8] to achieve the balance of speed of exploration and accuracy of the map using a single robot [8]. Freda et al. [5] use a sensor-based random tree (SRT). The tree is expanded as new candidate destinations near the frontiers of the sensor coverage are selected. These candidate destinations are evaluated considering the reliability of the expected observable features from that points. The tree is used to navigate back to past nodes with frontiers when no frontiers are present in the current sensor coverage. Recently, a novel laser data based SLAM algorithm using B-Spline as features has been developed in [9]. EKF is used in the proposed BS-SLAM algorithm and the state vector contains the current robot pose[1] together with the control points of the splines. The observation model used for the EKF update is the intersections of the laser beams with the splines contained in the map. In our proposal, we did not use the EKF, we use an integrated exploration feature-based approach.

Palabras clave:
Secuestro; SLAM; algoritmo aleatorizado.
Keywords:
Kidnapping; SLAM; randomized algorithms.

The basics of the B-splines are briefly presented in Section *Fundamental of B-splines.* The proposed approach to solve the kidnapping problem is detailed in Section *The Splam Approach.* Simulation results are discussed in Section *Experimental Results.* Finally, conclusion and future work are detailed in Section *Conclusions and Future Work.*

*Robotics Department, LIRMM, UMR 5506 - CC 477161 rue Ada, Montpellier - France. E mail:alfredo.torizpalacios@lirmm.fr.
**Facultad de Ciencias de la Computación, Benemérita Universidad Autónoma de Puebla, Av. 14 Sur y San Claudio, c.p. 72570, Puebla, Puebla, México. E-mail: {asanchez, aosorio}@cs.buap.mx

[1] One can distinguish between the robot position which is a 2-tuple (x, y), and the robot pose, which includes the orientation thus being a 3-tuple, (x, y, θ).

FUNDAMENTAL OF B-SPLINES

Let U be a set of $m+1$ non-decreasing numbers, $u_0 \leq u_2 \leq u_3 \leq \ldots \leq u_m$. The u_i's are called knots, the set U the knot vector, and the half-open interval $[u_i, u_{i+1})$ the i-th knot span. Note that since some u_i's may be equal, some knot spans may not exist. If a knot u_i appears k times (i.e., $u_i = u_{i+1} = \ldots = u_{i+k-1}$, where $k > 1$,u_i is a multiple knot of multiplicity k, written as $u_i(k)$. Otherwise, if u_i appears only once, it is a simple knot. If the knots are equally spaced (i.e., $u_{i+1} - u_i$ is a constant for $0 \leq i \leq m-1$), the knot vector or the knot sequence is said uniform; otherwise, it is non-uniform. The knots can be considered as division points that subdivide the interval $[u_0, u_m]$ into knot spans. All B-spline basis functions are supposed to have their domain on $[u_0, u_m]$. To define B-spline basis functions, we need one more parameter, the degree of these basis functions, p. The i-th B-spline basis function of degree p, written as $N_{i,p}(u)$, is defined recursively as follows:

$$N_{i,0}(u) = \begin{cases} 1, & \text{if } u_i \leq u \leq u_{i+1}; \\ 0, & \text{otherwise.} \end{cases}$$
$$N_{i,p}(u) = \frac{u-u_i}{u_{i+p}-u_i} N_{i,p-1}(u) + \frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}} N_{i+1,p-1}(u) \quad (1)$$

The above equation is usually referred to as the Cox-de Boor recursion formula [1]. Given $n+1$ control points $P_0, P_1, ..., P_n$ and a knot vector $U = u_0, u_1, ..., u_m$, the B-spline curve of degree p defined by these control points and knot vector U is

$$C(u) = \sum_{i=0}^{n} N_{i,p}(u) P_i \quad (2)$$

where $N_{i,p}(u)$'s are B-spline functions of degree p. The form of a B-spline curve is very similar to that of a Bézier curve [10].

THE SPLAM APPROACH

The strategy adopted to make the exploration process is named SRT, which is based on the construction of a data structure called Sensor-based Random Tree (SRT) that represents the roadmap of the explored area with a safe region associated (SR); each node of the tree ($\mathcal{T}$) consists of a position of the robot and its associated local safe region (LSR) constructed by the perception system of the robot. A continuous localization process is performed using a newly developed method based on the comparison of environmental characteristics such as turns and curves compared with the new environmental curves extracted from the LSR current position. In each iteration, the algorithm first gets the LSR associated with the current configuration of the robot, q_{curr}. Once obtained, the function EXTEND_TREE will be responsible for updating the tree by adding the new node and its LSR. Besides, the vector representing the curves that constitutes the environment will be updated by adding new curves drawn from the LSR that are not still part of it. The next step is to process the local frontier $\mathcal{F}$, where obstacles and free areas are identified.

Generally, $\mathcal{F}$ is a collection of discrete arcs. After obtaining these boundaries and if there are still free zones, the procedure RANDOM_DIR generates random directions in order to get one within the free arc. Then a q_{cand} configuration is generated, giving a step in the direction θ_{rand}. The step size is chosen as a fixed fraction of the radius of the LSR in that particular direction. Consequently q_{cand} will be collision-free due to the shape of S. If no free boundary arc is found, the robot will go back to the position of the parent node, q_{curr} and the exploration cycle will begin again.

Once the configuration q_{cand} is obtained, the procedure VALID_CONF will make sure that this new configuration is valid, i.e. that this new position is outside the LSR's of other nodes in the tree. If this new configuration is valid, it will be the new target configuration q_{dest} for the robot, however, if after a maximum number of attempts it is not possible to find a configuration q_{cand}, the parent node q_{cand} will be the new q_{dest} configuration and the robot will go back to the its parent configuration. Once the configuration q_{dest} is obtained, the function MOVE_TO(q_{curr}, q_{dest}) will move the robot to this configuration as follows : First, it will look in a previously defined list of control inputs ($list\_U$), an entry that approximates the robot's position to the position q_{dest} from the position q_{curr}, named $u_{control}$. Once obtained, $u_{control}$ will be applied to the robot, resulting in the displacement of the robot near to the goal. At this point, the odometric position stored in the memory of the robot and the increase in x, y and θ between the old and the current odometric position (x, y, θ) will be obtained again. This information, reported by the robot will be essential to get the position detected by the feature-based localization method with B-splines proposed in [12]. The algorithm will be repeated until q_{curr} and q_{dest} configurations are the same. This localization procedure uses B-spline curves to represent the frontiers between the free regions and the obstacles of a complex environment. It obtains in the first instance the estimated position of the robot, adding to the last position the position increments $\triangle x, \triangle y$ and $\triangle\theta$ made by the robot from the previous odometric position to the current odometric one; then the estimation of the surrounding environment captured by the sensor is obtained and the robot is located in the estimated position in the space, see [12] for more details.

A novel proposal for the kidnapping problem

The localization capability of a mobile robot is central to basic navigation and map building tasks. The two main instances of mobile robot localization problem are the continuous pose maintenance problem and the global localization also known as the 'robot kidnapping' problem. Global position estimation is the ability to determine the robot's position in an a priori or previously learned map, given no information other than that the robot is somewhere in the region represented by the map. Once a robot's position has been found in the map, local tracking is the problem of keeping track of that position over time. While existing approaches to position tracking are able to estimate a robot's position efficiently and accurately, they typically fail to globally localize a robot from scratch or to recover from localization failure. Global localization methods are less accurate and often require substantially more computational power.

Next, we will describe our approach to solve the global localization problem. The most important process for any kidnapping method is the collection and treaty of certain characteristic marks that would help the system to relocate the robot once it has been kidnapped. Initially when performing the task of exploration, this is exploited to find distinctive features of the environment and store them in a list, to be pretreated to acquire a format that in our case is easily recognizable. First we get the distances of these points found in q_{curr} position; these distances are sorted in ascending order, while we take the longest distance that is not repeated and turn the other features so that the points that make this distance are in an angle of 0 degrees. Then we get the angles between the marks. If two different distances between two different characteristics have the same length, they will be ordered according to the angles obtained, hence the importance of the rotation made by taking a no repeated distance, see figure 1.

Finally, all distances and angles data will be used to build a code that will be useful in the searching process. It will contain the number of landmarks involved N, the ordered distances Dn, the angles of each of the lines connecting the landmarks An and their position on the map being built $(Xi, Yi, i = 1, ..., n)$. The code or signature generated with the described process is $[N\ Dn\ An\ Xi\ Yi]$.

An important part of the kidnapping approach, is the exact knowledge of when the robot was moved from the last position to an unknown localization. This is possible because an inherent feature of our exploration method, the local security region (LSR). If at any time during the exploration and localization processes, the robot is unable to associate any of the curves in its current LSR with those obtained in the q_{curr} position, it means that the robot has been kidnapped. At this point, it will enter into a kidnapping state, and the system will try to find and connect any of the features stored in the memory with the features in the new environment.

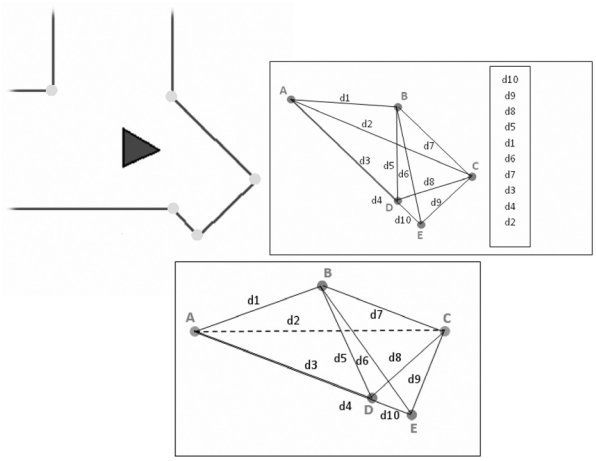

Figure 1 . The characteristic landmarks found, the distance between the landmarks in ascendent order and the rotated points to enforce a slope angle of 0 degrees in d2.

To maximize information and minimize the searching time, an auxiliary structure is created to store the environment that had been built up to that point and restart construction of the environment starting again from the position $(0, 0, 0)$. It means that no matter what position in the former map the robot is believed to be; after identifying the kidnapping, the system will reset its position to the coordinates $(0, 0, 0)$ to start again. As mentioned, the former map will not be wasted, it will be stored with the hope of finding a known area, and adjust the map currently being built at this position and merge the two structures into one. It is important to know that if any characteristic or high information content areas are detected before the kidnapping, the system will not be able to relocate the robot in the map being created and therefore the kidnapping may not be resolved. In this case the previous map will be discarded and it will be considered that there is no kidnapping to solve. On the contrary, if at the moment in which the robot is kidnapped there are several codes or signatures of the areas, the process will normally continue, with the construction of a new map, until it finds distinctive zone that enforces the following procedure:

- It will create a digital signature of the zone, following the steps mentioned above.
- It will verify the stored list to see if any of the codes contain the same number of landmarks than the landmarks found.
- If this number matches, the elements corresponding to the distances and the angles will be taken to find any code that contains similar data with epsilon of 0.01 m for distances and 0.01 radians for angles.

At this point, if there is already a candidate, we will obtain the coordinates of the stored landmarks and analyze the correspondence between the stored and the new found landmarks using the cross-reference. The ordered list that contains the stored and new landmarks of every distance has two options for relating the new landmarks in the list for every stored node. This can be seen in figure 2.

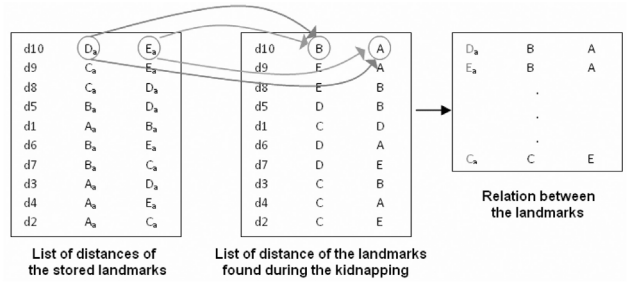

Figure 2 . Relation of stored landmarks and landmarks found during the kidnapping process.

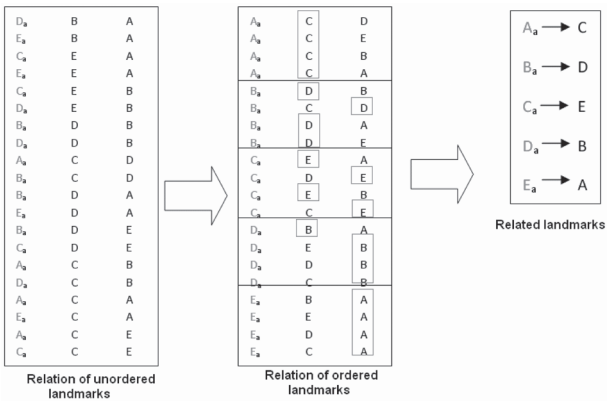

Figure 3 . Obtaining process of the related landmarks.

Once the relationship matrix is obtained, it will be sorted according to the first column that corresponds to the nodes of the stored list. Once ordered, the similar elements of the first column will be extracted and checked to see which element is $N-1$ times in columns 2 and 3, with N as the number of landmarks in the code. Finally this element will be the landmark corresponding to the stored landmark. This method will be repeated for all the different elements in the first column.

After applying the above method we already know exactly which node of the area currently seen corresponds to the node in the storage area, and we have the following:

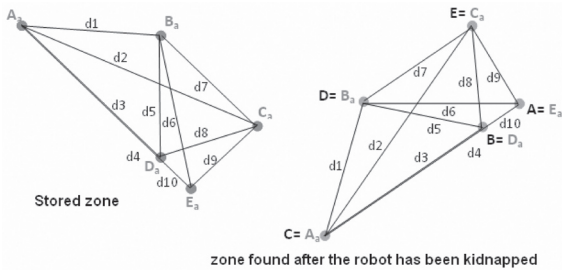

Figure 4 . Stored zone and zone found after the kidnapping.

With the area found and the nodes connected, the next step is to perform a kind of correction consisting in finding the angle and the displacement X, Y to match the area found in the kidnapping procedure with the area stored, i.e. the rotation and the translation processes will be performed on the map that is being created so that it can merge with the map saved at the time of the kidnapping. This correction is similar to the correction in the location method. First, a complete graph with the marks will be made, since the relationship between the nodes of the two graphs is known. Subsequently, the task is to find the difference of angles between them to perform angular adjustment with the equation used in the localization phase:

$$e_\theta^* = arg\ min_{e\theta} \sum_{i=1}^{N_a} \sum_{j=i+1}^{N_a} c_{ij}(\alpha_{ij}^{ref} - \alpha_{ij}^{curr})^2 \qquad (3)$$

Where α_{ij}^{ref} is the angle formed with the nodes ij in the graph that will be used to perform the correction and α_{ij}^{curr} is the angle formed with the nodes ij in the graph that is being corrected. The weight c_{ij} depends on the segment's length (the orientation of shorter segments is more sensitive to dislocations of the endpoints). After the angular correction e_θ^* is obtained, the translation adjustments ex and ey are processed using again the complete characteristics graph generated before. The generic segment's midpoint corresponding to the characteristics

graph stored in the list is denoted by $M_{ij}^{ref} = (x_{ij}^{ref}, y_{ij}^{ref})$, and the corresponding midpoint in the characteristics graph found in the kidnapping is denoted by $M_{ij}^{curr} = (x_{ij}^{curr}, y_{ij}^{curr})$.

Clearly, for any e_θ given, the coordinates $(x_{ij}^{curr}, y_{ij}^{curr})$ depend only on ex and ey respectively. Therefore, the optimal estimation of the translation adjustments can be obtained with:

$$e_x^* = arg\ min_{ex} \sum_{i=1}^{Na} \sum_{j=i+1}^{Na} c_{ij}(x_{ij}^{ref} - x_{ij}^{curr})^2 \qquad (4)$$

$$e_y^* = arg\ min_{ey} \sum_{i=1}^{Na} \sum_{j=i+1}^{Na} c_{ij}(y_{ij}^{ref} - y_{ij}^{curr})^2 \qquad (5)$$

Once obtained, the angular and translation adjustments will be applied to the tree built during the kidnapping. Later, to ensure a good fit, a global positioning final step is performed adjusting the curves segments of the new environment that are sufficiently close to the stored in the old map. If this condition between the maps exists, the localization process is carried out. Finally and in order to keep only one structure corresponding to the complete map during the merger, the new environment nodes that are closer than a certain threshold of saved map nodes will be discarded as they will be considered as a repetition of the stored. Similarly, the nodes above this threshold but which are within the LSR of any node will be added as child nodes of the owner of the LSR. The exploration will continue in the leaf nodes of the new structure.

EXPERIMENTAL RESULTS

For our experiments we are using both simulated and real Pioneer P3DX robot equipped with front and rear bumper arrays, a ring of eight forward ultrasonic transducer sensors (range-finding sonar) and a Hokuyo URG-04Lx laser range finder. The Pioneer P3DX robot is an unicycle robot. The LIRMM[a] environment was used in the experimental and simulation tests (the environment contains several corridors). Figure 5 show the two final maps: at the left, the obtained map without localization (only with odometric estimates); at the right, the obtained map with the proposed approach. If we compare the two final maps, one can mention that when the robot does not use the localization process, it collides frequently with the obstacles as can be appreciated in the left image.

For the association process, the basic procedure proposed in [9] was retaken, but with a new functionality derived from the properties of the B-splines, i.e., we ensure that these new curves and those belonging to the environment have the same lengths. This new feature enabled better and more accurate association of the data collected by the sensor than the association obtained with the basic method originally proposed in the approach presented in [9]. The updating of the environment is not performed in every robot's movement, but when a new position in the frontier between the known and the unknown environment is reached, because the LSR contains the necessary information for the localization process, saving processing time while exploring. Next figures show the kidnapping process proposed in this work[12].

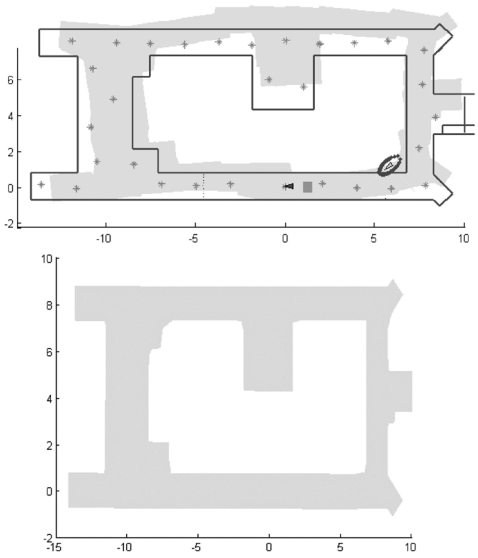

Figure 5 . Final map only with odometric estimates and the final map with localization.

The geometric properties of the ends of the curves was also considered, i.e., one can make a final check of the association taking the distances between the ends of the curves of the environment and the ends of the curves corresponding to the estimated position and verify their similitude. This exhaustive verification is necessary because the nature of the proposed localization method. It can be said that the approach presented made a good use of the parametric representation of the environment characteristics at the time of the data association.

Due to the exploration system used, in which each node in the exploration tree contained a robot position with its associated LSR, and the features of the localization method, the map updating is performed every time a new LSR is obtained, unlike

[a]Laboratoire d'Informatique de Robotique et de Microélectronique de Montpellier, France

the approach presented in [9]. The updating of the environment is not performed in every robot's movement, but when a new position in the frontier between the known and the unknown environment is reached, because the LSR contains the necessary information for the localization process, saving processing time while exploring. Next figures show the kidnapping process proposed in this work.

In the first instance, the solution provided to this problem is in fact very simple, if the robot is not able to recognize the local safe region on which it is working, then one can consider that the robot is in the kidnapping state. The problem with this first solution resides in the fact to consider that if the robot explores in a corridor-like zone and it is kidnapped and placed in a similar zone. In this case the robot will not have an idea of its current situation and it will continue with the exploration task, the final result will be a completely erroneous map.

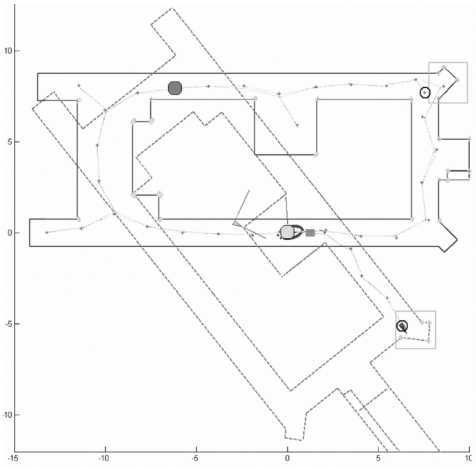

Figure 6 . Snapshot showing the execution of the proposed kidnapping strategy.

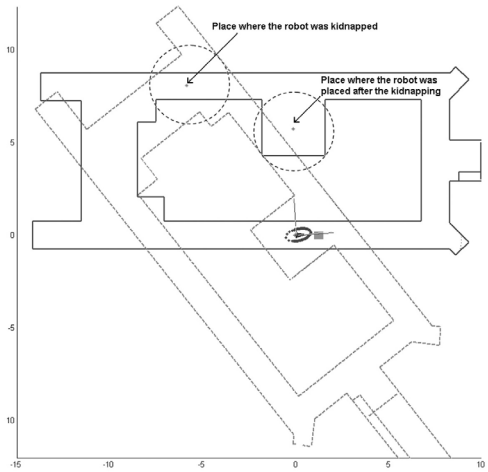

Figure 7 . One can see in this figure, the place where the robot was kidnapped and the place where the robot was put after the kidnapping.

Taking this possible scene, one second solution was considered, which consists of taking advantage of the natural backtracking process of the SRT method [3]. If the robot returns to a parent node and it is incapable to reach it; the node is inaccessible by the presence of the obstacles. Then the algorithm will enter to the kidnapping phase and the current branch in which it arose the problem will comprise the obstacles that they will have to be relocated when this phase is solved. Here, the new root node will be the node son since this node could not be connected; all the structure will be displaced and then the new root has the coordinates (0,0,0).

CONCLUSIONS AND FUTURE WORK

Ever since SLAM was firstly introduced, many approaches have been investigated. Most of the early SLAM work was point-feature based. The main drawback with point-feature based SLAM is that measurements acquired from typical sensors did not correspond to point feature in the environment. After the raw sensor data is acquired, post processing is required to extract point features. This process may potentially introduce information loss and data association error. Further more, in some situation, the environment does not have enough significant structure to enable point features to be robustly extracted from them. We can mention that we have developed a robust SLAM tool that is not limited to environments with linear features. The localization method is perfectly suited for new curves that can be increasingly seen in everyday's life. The theory and implementation of the B-splines was a powerful tool in our approach, and can be adapted to environments where the previous methods only considered simple descriptions.

As future work we have considered a great challenge: the extension of our proposal to the case of integrated exploration with multiple robots, which will take to us to the search of a solution to the multi-robot localization problem.

REFERENCES

[1] De Boor, C. (2001). A practical guide to splines. *Applied Mathematical Sciences 27* , revised edition. Springer-Verlag.

[2] Chang, H. J., George Lee, C. S., Yung-Hsiang, L., Hu, Y. C. (2007). P-SLAM: Simultaneous localization and mapping with environmental-structure prediction. *IEEE Transactions on Robotics,* 23(2):281-293.

[3] Espinoza L, J., Sánchez L., A., Osorio L., M. (2007). Exploring unknown environments with mobile robots using SRT_Radial. *IEEE Int. Conf. on Intelligent Robots and Systems,* pp. 2089-2094.

[4] Feder, H. J. S., Leonard, J. J., Smith,C. M. (1999). Adaptive mobile robot navigation and mapping. *The International Journal of Robotics Research,* 18(7):650-668.

[5] Freda, L., Loiudice, F., Oriolo, G. (2006). A randomized method for integrated exploration. *IEEE Int. Conf. on Intelligent Robots and Systems,* pp. 2457-2464.

[6] Jensfelt, P., Kristensen, S. (2001). Active global localization for a mobile robot using multiple hypothesis tracking. *IEEE Transactions on Robotics and Automation,* 17(5):748-760.

[7] Liu, Z., Shi, Z., Xu,W. (2010). Multi-component information-equalized extended strong tracking filter for global localization: A scheme robuts to kidnapping and symmetrical environment. *Robotics and Autonomous Systems,* 58:465-487.

[8] Makarenko, A. A., Williams, S. B., Bourgante, F., Durrant-Whyte, H. F. (2002). An experiment in integrated exploration. *IEEE Int. Conf. on Intelligent Robots and Systems,* pp. 534-539.

[9] Pedraza, L., Dissanayake, G., Valls Miro, J., Rodriguez-Losada, D., Matia, F. (2009). Extending the limits of feature-based SLAM with B-Splines. *IEEE Transactions on Robotics,* 25:353-366.

[10] Rogers, D. F. (2001). An introduction to NURBS with historical perspective', *Morgan Kaufmann Publishers.*

[11] Thrun, S., Burgard, W., Fox, D. (2005). *Probabilistic robotics.* The MIT Press.

[12] Toriz P., A., Sanchez L., A., Zapata, R., Osorio, M. A. (2010). Building feature-based maps with B-splines for integrated exploration. To appear in *LNAI.*

[13] Yamaguchi, F. (1988). *Curves and surfaces in computer aided geometric design.* Springer-Verlag.