Robot evolutionary localization based on attentive visual short term memory

Julio Vega and Eduardo Perdices and José M. Cañas[1]

***Abstract*— Cameras are one of the most relevant sensors in autonomous robots. This paper proposes a dynamic visual memory to store the information gathered from a moving camera on board a robot, an attention system to choose where to look at with this mobile camera, and a visual localization approach which uses this visual memory. The visual memory is a collection of relevant task-oriented objects and 3D segments, and its scope is wider than the current camera field of view. The attention system takes into account the need to reobserve objects in the visual memory and the need to explore new areas. The visual memory is useful also in localization tasks since it provides more information about robot surroundings than the current instantaneous image. Several experiments have been carried out both with simulated and real Pioneer and Nao robots to validate the system and each of its components.**

I. INTRODUCTION

Cameras have been incorporated in the last years to robots as common sensory equipment. They are very cheap sensors and may provide much information to robots about their environment, but extracting relevant information from the image flow is not easy. Vision has been used in robots for navigation, object recognition, 3D mapping, visual attention, robot localization, etc.

Robots usually navigate autonomously in dynamic environments, and so they need to detect and avoid obstacles. Many works have been presented in vision based navigation and control, even without requiring any reconstruction of the environment [Remazeilles *et al.*, 2006]. Obstacles can also be detected through 3D reconstruction. Recovering 3D-information has been the main focus of the computer vision community for decades. Stereo-vision methods are the classic ones, based on finding pixel correspondences among the two cameras and triangulation, despite they fail with untextured surfaces. Vision depth sensors like Kinect offer now a different technology for visual 3D reconstruction.

The visual representation of interesting objects around the robot may improve the quality of robot's behavior as it handles more information when making decisions. This poses a problem when the objects lie beyond the current field of view of the camera. To solve it, some systems use omnidirectional vision. Others, like humanoids or robots with pantilt units, use mobile regular cameras that can be orientated at will and manage a visual memory of robot surroundings that integrate the information from the images taken from different locations. The problem of selecting where-to-look-at at every time, known as gaze control or *overt attention* ([Cañas *et al.*, 2008],[Zaharescu *et al.*, 2005]), arises there. Usually the need to quickly explore new areas and the need to reobserve known objects to update their positions, etc. influence that selection.

Arbel and Ferrie presented in ([Arbel and Ferrie, 2001]) a gaze-planning strategy that moves the camera to another viewpoint around an object in order to recognize it.

Robots need to know their location inside the environment in order to unfold the proper behavior. Using robot sensors (specially vision) and a map, the robot estimates its own position and orientation inside a known environment. Robot self-localization has proven to be complex, especially in dynamic environments and in those with much symmetry, where sensors values can be similar at different positions.

Grid based probabilistic localization algorithms were successfully applied with laser or sonar data in small known environments [Burgard and Fox, 1997]. They use discretized probability distributions and update them from sensor data and movement orders, accumulating information over time and providing a robust position estimation. Particle filters use sampled probability functions [Dellaert *et al.*, 1999] and extend the techniques to larger environments. At the beginning the maps were provided in advance but in the last years the SLAM techniques tackle localization simultaneously with the map building. There are many particle filter based SLAM techniques. In addition, one of the most successful approaches is Mono-SLAM from Andrew Davison [Newcombe and Davison, 2010; Gerardo Carrera and Davison, 2011] based on a fast Extended Kalman Filter for continuous estimation of 3D points and camera position from relevant points in the image.

In ([Mariottini and Roumeliotis, 2011]) Mariottini and Roumeliotis presented a strategy for active vision-based localization and navigation of a mobile robot with a visual memory where previously visited areas are represented as a large collection of images. It disambiguates the location taking into account the sequence of distinctive images, while concurrently navigating towards the target image. In [Jensfelt and Kristensen, 2001], Jensfelt et al. presented an active global localization strategy that uses Kalman filtering (KF) to track multiple robot pose hypotheses. Their approach can be used even with incomplete maps and the computational complexity is independent on the size of the environment.

In this paper we propose a visual perceptive system for an autonomous robot composed of two modules. First, a short term visual memory of robot surroundings fed with images from a mobile camera. The memory stores 3D segments of

*This work was supported by the project S2009/DPI-1559, RoboCity2030-II, from the Comunidad de Madrid and by the project 10/02567 from the Spanish Ministry of Science and Innovation

[1]All of them are with Esc. Tec. Sup. Ingeniería de Telecomunicación, University of Rey Juan Carlos, Fuenlabrada, Spain julio.vega@urjc.es, eperdices@gsyc.es, jmplaza@gsyc.es

ISBN: 978-84-695-3472-4

the environment and objects. A gaze control algorithm has been developed to select where the camera should look at at every time. Second, a visual localization algorithm that uses current image or the current contents of this memory to estimate robot position.

The remainder of this paper is organized as follows. Second section presents the design of the proposed visual system. The next two sections describe its two building blocks: attentive visual memory component and localization component. The fifth section includes some experiments, both with simulated and real robots, performed to validate our system. Finally some conclusions are summarized.

II. Design

The proposed perceptive system is designed for autonomous robots that use a mobile camera, like that on the head of humanoids or in robots with pan-tilt units. The block diagram of the robot control architecture is showed in Figure 1. It has been divided into two main components: `active_visual_memory` and `localization`. They will be described in detail in the following sections.

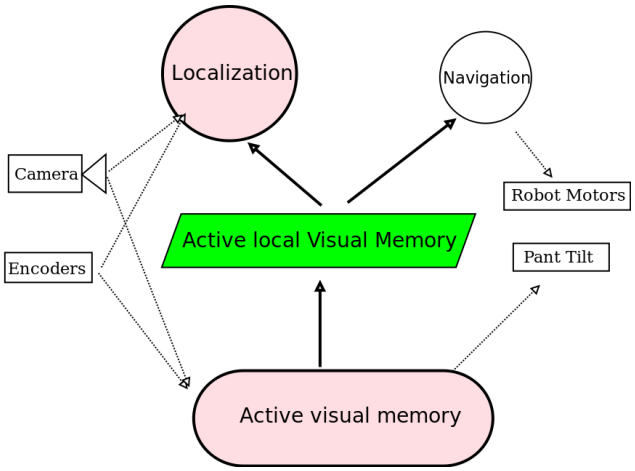

Fig. 1. Block diagram of the proposed visual system

First, the `active_visual_memory` component builds a local active visual memory of objects in the robot's surroundings. The memory is built analyzing each camera image looking for relevant objects (like segments, parallelograms, arrows, etc) and updating the object features already stored in the memory like their 3D position. The memory is dynamic and is continuously coupled with camera images. The new frames confirm or correct the object features stored in memory, like their 3D relative position to the robot, length, etc.. New objects are introduced in memory when they appear in images and do not match any known object.

This memory has a broader scope than the camera field of view and objects in memory have more persistence than the current image. Regular cameras typically have 60 degrees of scope. This would be good enough for visual control but a broader scope may improve robot responses in tasks like navigation, where the presence of obstacles in the robot's surroundings should be taken into account even if they lie out of current field of view.

This memory is intended as local and short-term. Relative object positions are estimated using robot's odometry. Being only short term and continuously correcting with new image data there is no much time to accumulate error in the object relative position estimation. Currently the system deals only with objects on the floor plane (ground hypothesis) and uses a single camera. It can be extended to any 3D object position and two cameras.

In order to keep this short term visual memory consistent with reality, the system has mechanisms to properly refresh it. The camera is assumed to be mobile, typically mounted over a pan-tilt unit. Its orientation may be controlled and changed during robot behavior at will, and so, the camera may look towards different locations even if the robot remains static. In order to feed the visual memory, an overt attention algorithm has been designed to continuously guide camera movements, choosing where to look at at every time. It has been inserted inside the `active_visual_memory` component and associates two dynamic values to each object in memory: *salience* and *life* (quality). Objects with low life are discarded and objects with high salience are good candidates to look at.

The position of objects already in memory are themselves foci of attention in order to refresh their perceived features. Random locations are also considered to let the robot explore new areas of its surroundings. In addition, new foci of attention may also be introduced to check the presence of some hypothesized objects. For instance, once the robot has seen three vertices of a parallelogram, the position of the fourth one is computed from the visual memory and ordered as a tentative focus of attention for the camera.

Second, a vision based localization algorithm has been developed in the `localization` component. It uses a population of particles and an evolutionary algorithm to manage them and find the robot position. The health of each particle is computed based on the current image or based on the current contents of the visual memory. The local visual memory provides information about robot's surroundings, typically more than the current instantaneous sensor readings. In this way, the visual memory may be used as a virtual sensor and its information may be used as observations for the localization algorithm. Because of its broader scope it may help to improve localization, especially in environments with symmetries and places that look like similar according to sensor readings.

III. Attentive short term visual memory

The goal of our visual memory is to do a visual tracking of the various basic objects in the scene surrounding the robot. It must detect new objects, track them updating its relative position to the robot and remove them from the memory once they have disappeared.

The first stage of the system is a 2D analysis, in which 2D segments in the current image are detected using the Solis algorithm [Solis *et al.*, 2009]. Then the system puts

ISBN: 978-84-695-3472-4

these objects in 3D space according to the *ground-hypothesis*, assuming they all are flat on the floor, and stores them in memory maybe merging with already existing 3D segments.

Each 3D visible object already stored in memory is projected on the current image plane. The system refutes/corroborates such predicted segments, comparing them with those extracted from current image. This comparison leads to three sets of segments. First, in case of matching the features of the stored 3D segment are updated taking into account the new observation. Second, if a segment is identified in the current image but does not match any prediction, the system creates a new one in 3D. Before inclusion in the 3D memory some post-processing is needed to avoid duplicates due to noise in the images, for instance comparing the relative position between segments, as well as its orientation and proximity. And third, for predicted segments not really observed in current image their quality goes down and eventually will be removed from memory.

The memory content is a set of 3D segments situated on the robot coordinate system as can be seen in Figure 2. In addition, the memory can establish relationships between them to make up more complex elements such as arrows, parallelograms, triangles or other objects. The 3D analysis of the angles formed by each segment provides information about how the segments are connected to each other. This feature can be used to merge incomplete or intermittent segments. Similarly, we can extract the position of a possible fourth vertex using the information about other edges and/or possible parallelogram vectors. This capability makes our algorithm robust against occlusions, which occur frequently in the real world.

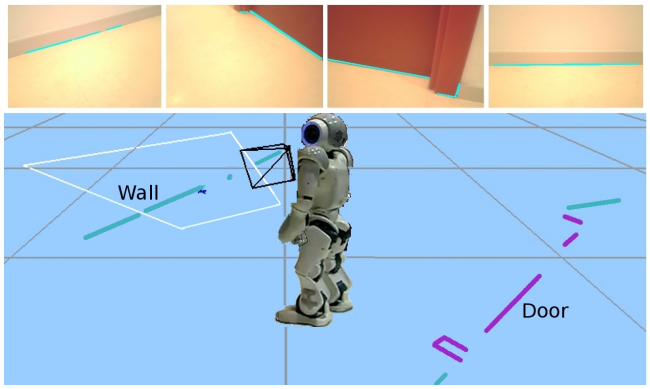

Fig. 2. Visual memory with 3D segments coming from four images of robot surroundings

The objects may eventually disappear from the scene, and then they should be removed from the memory in order to maintain coherence between the representation of the scene and the reality. To forget such old elements, we created the *life* dynamics. Every time an object is observed in current image, its life is increased, with a maximum saturation limit. Life of unobserved objects decreases over time. It is a measurement of the quality of the object estimation, the inverse of its uncertainty. When the object's life goes below a given threshold it is simply discarded from visual memory. This way, there is a need to periodically reobserve objects to keep them alive in memory.

$$Life(t) = \begin{cases} min(MAX_{LIFE}, Life(t-1) + \Delta) \\ \qquad \text{if object observed} \\ Life(t-1) - 1 \\ \qquad \text{otherwise} \end{cases}$$

We have designed a mechanism to control the camera movements to track objects and explore unknown areas from the scene. To control the movement of the pan-tilt unit, we introduced the dynamics of salience and attention points. Each memory element has a 3D position in the scene (XYZ) and an associated salience that summarizes the system's desire to look at it. It grows over time and vanishes every time that element is visited, following the next equation.

$$Salience(t) = \begin{cases} Salience(t-1) = 0 & \text{if object attended} \\ Salience(t-1) + 1 & \text{otherwise} \end{cases}$$

The point with highest salience will be the next to be visited. If the salience is low, it will not be visited now. The mechanical commands for the pan-tilt unit in order to look at that location can be computed using inverse kinematics. When the gaze-control module chooses a given object, it is going to look at it for certain interval (3 seconds), tracking it if it moves spatially. For this tracking, and to avoid excessive oscillations, we use a *P-controller* to control the speed of the pan and tilt and thus continually focus that object on the image center.

Furthermore, it may be interesting to look for new objects in the scene. For this our system periodically inserts scanning points with high salience in memory. This search is especially interesting at the beginning, when there are many unknowns areas of the scene where the objects of interest can be.

IV. Evolutionary visual localization

We have designed a new approach to solve robot self-localization specifically designed to bear symmetries. It is an evolutionary algorithm, a type of meta-heuristic optimization algorithm that is inspired by the biological evolution.

In this kind of algorithms, candidate solutions are so-called "individuals", which belong to a population that evolve over time using genetic operators, such as mutation or crossover. Each individual is a tentative robot position and is evaluated with a quality function which calculates its "health", that is, a measure to know how good its localization is with regard to the optimal solution. We have defined two different health functions, one based on instantaneous measures of robot sensors and another one based on the visual memory contents.

The main idea of the algorithm consists of keeping several races competing among each other in several likely positions. In case of symmetries from observations, we will create new races on various positions where the robot might be located. After some iterations, predictably, new observations will provide information to reject most of races and we

ISBN: 978-84-695-3472-4

will obtain the real robot pose from the best race. On each iteration the algorithm performs several steps:

- Health race calculation using the information obtained after analyzing images
- Explorers creation: We spread randomly new individuals with the aim to find new candidate positions where new races could be created.
- Races management: We create, merge or delete races depending on their current state.
- Races evolution: We evolve each race by using genetic operators. Besides, if the robot is moving, we update all races taking into account this movement.
- After calculating each race health, we will select one of them to set the current robot pose.

A. Health calculation from instantaneous images

In this version of the algorithm we analyze the current image to obtain characteristic lines (Figure 3 (left)), with the Solis line detection algorithm. Besides, we sample these observed lines with points to make the comparison between lines easier. All these selected sampling points will be used as input data to calculate the health of each individual. In addition, when computing the health of an individual placed at certain location the theoretical observation is computed using the map, projecting the map's lines into the camera placed at that location. It contains the lines the robot would see if it were placed at that location (Figure 3 (right)).

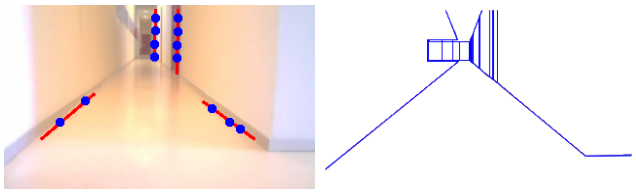

Fig. 3. Lines detected in current image and theoretical image

The health of an individual placed at certain location is computed comparing the theoretical set of visible objects from that location (theoretical observation, Figure 3 (right)) with objects currently observed (real observation,3 (left)). The more similar the predicted objects and the observed ones are, the more likely such location is the correct one. We cover all sampling points (N) over the detected lines and calculate for each one the Euclidean distance d_i in pixels to the closest theoretical line with the same type than the point. After calculating d_i for each point, we can compute the health with the next equation, where M is a normalization factor.

$$H = 1 - \frac{\sum_{i=0}^{N} \frac{d_i}{N}}{M} \quad (1)$$

B. Health calculation from visual memory

In case of using the visual memory we don't need to analyze each image, just the current visual memory contents from the `active_visual_memory` component, that is, the set of 3D segments inside, relative to our robot. So we can't compare lines in image as we did before. Besides, we have to take into account that lines may not be detected completely or they may be divided into several small lines. Thus, to calculate the current health of an individual we cover all lines belonging to the visual memory, for each line we get its extremes and calculate the Euclidean distances d_{j_s} and d_{j_e} to the closest theoretical line with the same type, that is, similar to health function with instantaneous images but in 3D. After calculating d_{j_s} and d_{j_e} for each line, we can calculate the health as follows:

$$H = 1 - \frac{\sum_{i=0}^{N} \frac{(\frac{d_{j_s}+d_{j_e}}{2})}{N}}{M} \quad (2)$$

C. Race management and evolution

Since our algorithm handles several races at the same time, we need to know when to create a new race, delete it, or merge two races. Besides, our approach has a maximum number of races N to keep computation time low enough. A new race is created whenever we have new candidates (coming from the explorers creation), whose health is better than in our current races. Furthermore, races are merged when they are very close to each other, and deleted if their health is lower than a threshold.

In each iteration, each race evolves all its individuals through three genetic operators, elitism (select the best exploiters), crossover (mix two explorers), and mutation (apply a thermal noise). Besides, in case the robot has moved since the last iteration, we apply a motion operator to all race individuals and explorers using robot odometry.

V. Experiments

To verify our different approaches of visual memory, visual attention and visual localization, we conducted several experiments. Our experimental real platforms were an ActivMedia Pioneer 2DX robot equipped with a Logitech Autofocus camera (2 mega-pixels) and a Nao Robot from Aldebaran Robotics (v.3). Besides, we have used Gazebo 0.9 as robot simulator. All our experiments are implemented on C++ under JdeRobot robotics software platform, which uses ICE as a middle-ware.

A. Attentive visual memory

In the first experiment we want to show how the robot is unable to navigate using only the instantaneous information received from the camera. If robot only uses instantaneous image while it is navigating through a corridor and it is approaching a curve area, it is able to see just some lines in front of itself (4-a), but with short-term memory it can observe that the path in front of itself is a curve (4-b).

In the second experiment, the situation is presented to solve a temporary occlusion. This happens very often in real environments where there are dynamic objects which can obstruct the robot field of view. After a few seconds, robot has recovered environment information thanks to the short-term memory and the visual attention system. This information is showed in Figure 5-a. Then another robot

ISBN: 978-84-695-3472-4

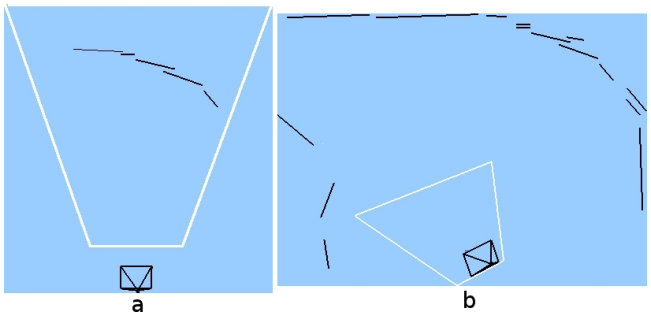

Fig. 4. (a) Instantaneous field of view (b) Short-term memory

appears (Figure 5-b) occluding the field of view of our robot, so it is unable to see anything. This situation is solved by our system because of the persistence of the objects in the short-term memory. As we discussed in section III, the memory is refreshed over time. If it is inconsistent, that is, what the robot sees does not match with the information stored in memory, we give a confidence interval until this situation is solved.

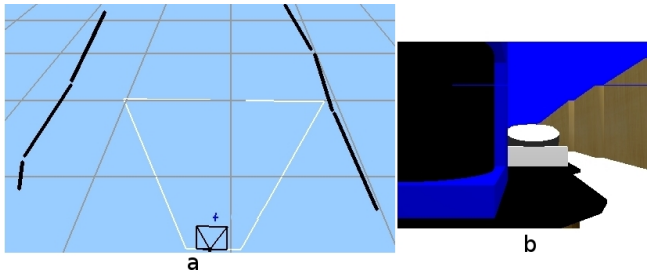

Fig. 5. (a) Short-term memory (b) Occlusion situation

Omnidirectional cameras or even wide-angle cameras can also improve the instantaneous field of view of a conventional camera (Figure 4). The main advantage of our technique compared with them is that the active visual memory can also solve occlusion situations (Figure 5) better.

B. Visual localization from current image

We have performed several experiments to validate the evolutionary algorithm, especially with real robots. The first experiment has been performed with a real Nao robot travelling through a corridor (Figure 6).

This first experiment shows how the algorithm is able to follow the real movement of the robot starting on a known position. Once the robot is located in a known position, we move the robot around the environment and measure its localization error. The red line in Figure 7 shows the calculated positions, the green line the real robot path, and the brown area is the error measured. The average error has been 15,4 cm, the algorithm is able to follow the robot movement even when its instantaneous observations don't provide enough information thanks to robot odometry.

The second experiment (Figure 8) shows how the algorithm works with symmetries and kidnappings. At (1.a), we

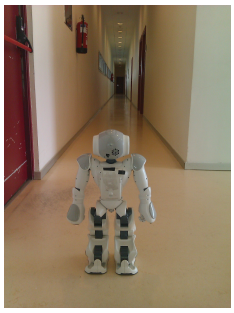

Fig. 6. Nao robot travelling a corridor for experiments

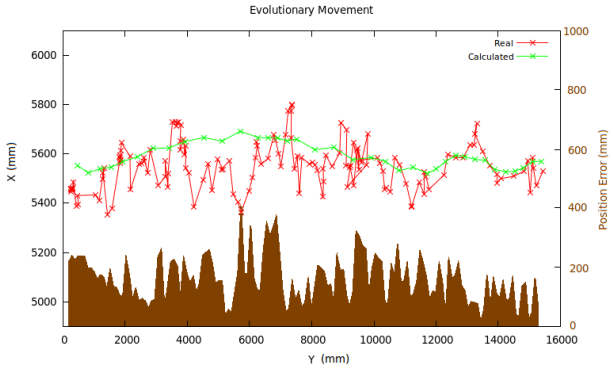

Fig. 7. Estimated localization and position error over time

locate the robot in front of a door, so the algorithm creates several races where the robot may be, the algorithm selects one of them (a wrong one) but keeps another one on the right location, then the robot moves and obtains more information from the world and finally rules the wrong location out and selects the correct one (1.b). Afterwards, we kidnap the robot to another location (2.a) and it takes a while until the robot changes to the new right location. It happens because the location's reliability changes gradually to avoid changing with false positives, but after a while, it changes to the new position (2.b). A second kidnap is performed (3.a), this case is similar to the first one, at first it selects a wrong localization (very close to the correct one, 3.b), but after some iterations it changes to the correct one (3.c).

The average error after selecting the correct race was 14,9 cm and 3,2 degs. The recovery time (the time spent until the algorithm calculates a new plausible pose after a kidnapping) was 29,6 secs.

C. Visual localization from visual memory

In this experiment, we tested our whole system including visual memory, visual attention and visual localization on the real Nao humanoid. Initially, the robot is located in the middle of our department corridor gathering information about its surroundings. It is able to move autonomously around the corridor, moving its head and body in order to detect all segments around it in a few seconds. Figure 2 shows some snapshot images taken in this experiment and the short-term memory built with them. Figure 9 shows the

ISBN: 978-84-695-3472-4

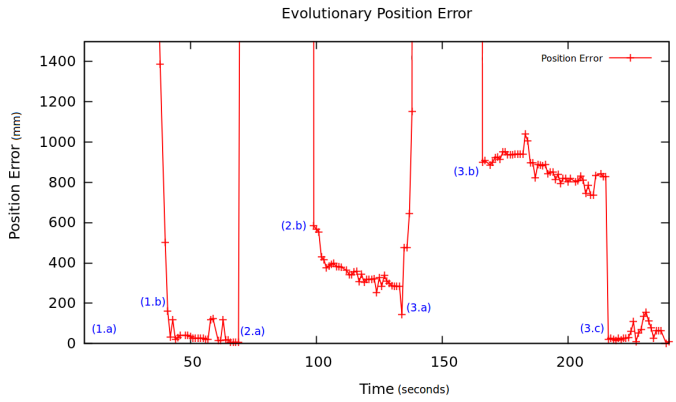

Fig. 8. Position error over time

localization estimation obtained using that visual memory contents.

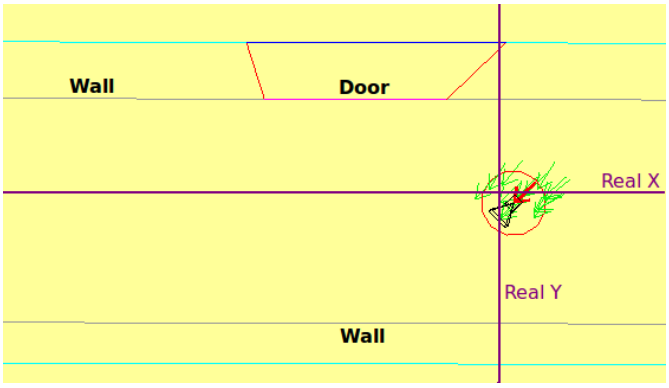

Fig. 9. Localization calculated with visual memory

VI. Conclusions

In this paper a visual perception system for autonomous robots has been presented. It processes the images from a mobile camera and builds a short term local memory with information about the objects around the robot, even if they lie outside the current field of view of the camera. This visual memory stores 3D segments and simple objects like parallelograms with their associated properties like position, uncertainty (inverse of *life*), color, etc.. It allows better navigation decisions and even better localization as includes more information than the current image, which can even be temporary occluded.

An overt visual attention mechanism has been created to continuously select where the mobile camera should look at. Using a *salience* dynamics and choosing the most salient point the system shares the gaze control between the need to reobserve objects on the visual memory and the need explore new areas, providing also Inhibition of Return.

We developed a visual self-localization technique that uses an evolutionary algorithm. It keeps a population of particles to represent tentative robot positions and the particle set evolves as new visual information is gathered or with robot movements. It has been especially designed to bear symmetries, grouping particles into races. There is one race for each likely position and inside it individuals do the fine grain search. It can work both with just the current image or the contents of the visual memory.

This visual perception system has been validated both on real robots and in simulation. The memory nicely represents the robot surroundings using the images from the mobile camera, whose movement is controlled by our attention mechanism. The memory is dynamic but have some persistence to deal with temporary occlusions. The localization works in real time, provides position errors below 15cm and 5 degrees and is robust enough to recover from kidnappings or estimation errors in symmetric environments.

We are working in extending the system to stereo pairs and dealing with objects at any 3D position, not just the floor. We are also studying how to deal with more abstract objects like tables and chairs into the visual memory. Regarding localization we are working in introducing a monoSLAM EFK for each race of the evolutionary algorithm and improve them to extract localization information from abstract objects, not only 3D points.

References

[Arbel and Ferrie, 2001] T. Arbel and F. Ferrie. Entropy-based gaze planning. *Image and Vision Computing, vol. 19, no. 11, pp. 779-786*, 2001.

[Burgard and Fox, 1997] W. Burgard and D. Fox. Active mobile robot localization by entropy minimization. *Proc. of the Euromicro Workshop on Advanced Mobile Robots, Los Alamitos, CA, pp. 155-162*, 1997.

[Cañas *et al.*, 2008] J.M. Cañas, M. Martínez de la Casa, and T. González. Overt visual attention inside jde control architecture. *International Journal of Intelligent Computing in Medical Sciences and Image Procesing. Volume 2, Number 2, pp 93-100, ISSN: 1931-308X. TSI Press, USA*, 2008.

[Dellaert *et al.*, 1999] Frank Dellaert, Wolfram Burgard, Dieter Fox, and Sebastian Thrun. Using the CONDENSATION algorithm for robust, vision-based mobile robot localization. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, CVPR-99*, volume 2, pages 588–594, Fort Collins, Colorado (USA), June 1999. IEEE Computer Society.

[Gerardo Carrera and Davison, 2011] Adrien Angeli Gerardo Carrera and Andrew J. Davison. Lightweight slam and navigation with a multi-camera rig. In *Proceedings of the 5th European Conference on Mobile Robots ECMR 2011*, pages 77 – 82, September 2011.

[Jensfelt and Kristensen, 2001] P. Jensfelt and S. Kristensen. Active global localization for a mobile robot using multiple hypothesis tracking. *IEEE Trans. on Robotics and Automation, vol. 17, no. 5, pp. 748-760*, 2001.

[Mariottini and Roumeliotis, 2011] G.L. Mariottini and S.I. Roumeliotis. Active vision-based robot localization and navigation in a visual memory. *Proc. of ICRA*, 2011.

[Newcombe and Davison, 2010] Richard A. Newcombe and Andrew J. Davison. Live dense reconstruction with a single moving camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010*, pages 1498 – 1505, San Francisco, California (USA), June 2010.

[Remazeilles *et al.*, 2006] A. Remazeilles, F. Chaumette, and P. Gros. 3d navigation based on a visual memory. *Proc. of ICRA*, 2006.

[Solis *et al.*, 2009] A. Solis, A. Nayak, M. Stojmenovic, and N. Zaguia. Robust line extraction based on repeated segment directions on image contours. *Proc. of the IEEE Symposium on CISDA*, 2009.

[Zaharescu *et al.*, 2005] A. Zaharescu, A. L. Rothenstein, and J. K. Tsotsos. Towards a biologically plausible active visual search model. *Proc. of Int. Workshop on Attention and Performance in Computational Vision WAPCV-2004, pp. 133-147*, 2005.

ISBN: 978-84-695-3472-4