A Distributed Processing Architecture for Vision Based Domestic Robot Navigation

Marcel-Titus Marginean and Chao Lu
Computer and Information Sciences, Towson University
8000 York Rd, Towson, MD 21252, USA
tmargi1@students.towson.edu
clu@towson.edu

Abstract - The paper proposes a distributed control architecture for indoor robot navigation employing both on board and external computer vision units. A communication protocol for cooperative localization and obstacle avoidance is developed, where the visual processing is performed cooperatively between an on board unit (robot) and an external unit (base station), each of which has access to the on-board (mobile) and house mounted (fixed) cameras respectively. A set of data-structures and algorithms for distributed video processing and decision making are proposed. The idea of an aspect oriented indexing system for a database of visual objects used in object recognition is introduced.

Keywords: Computer Vision, Robot Navigation, Distributed Control.

1 Introduction

Anticipated for a long time in science-fiction literature domestic robotics is timidly starting to appear. While the first applications like vacuum-cleaners or toys do not share much with the versatile robotic servants envisioned in literature, it is just a matter of time before more useful robots appear, mainly driven by the demand for help from the aging population in the industrialized world. An active area of research is already taking place for technologies to achieve what is called independent assisted living where the senior citizens are enabled by technology to live alone in their own houses as opposed to being moved into an assisted living facility while being able to receive help as needed [14]. Domestic robots are explored to play a prime role in this field in the not so distant future [12, 14].

One of the main obstacles faced by the development of mobile robots is the ability to properly operate in domestic environments where they must be able to safely navigate and avoid objects, people or pets [13]. While non-visual methods have been attempted [13], computer vision emerges as the most promising technology [3, 4, 5, 6, 15,], but this brings with it the major challenge of processing in real-time the humongous amount of information captured by cameras on an energy efficient embedded computer.

In this paper we propose a method to reduce the processing required by the on board computer by taking advantage of the house wireless network, fixed cameras and additional processing capabilities available on a base station where higher power consumption required by faster CPUs is not a problem. The on board computer is still in charge of basic navigation and it should be able to maintain course and operate autonomously for periods when it is out of sight from the fixed cameras; however the base station computer(s) shall be able to provide localization help when the robot is in view and handle the mission. The more powerful computer on the base station maintains a database of objects, images, and performs high level object recognition tasks not only on the images from fixed cameras but also on the pre-processed images sent by the mobile unit (robot). By doing preprocessing of images on the mobile unit and sending sub-images for recognition only when needed, we should consider to optimize the amount of Wi-Fi bandwidth consumption.

Moreover, we reduce the amount of processing by taking advantage of prior knowledge about the environment. Since the robot operates in a house already augmented with sensors, a map of the building, and the position of the camera is readily available, sensors, artificial landmarks or beacons can be placed in areas of special interests with their location being known beforehand.

The paper is organized as follows: In the next sub-section a brief literature review is presented; section 2 is dedicated to the principal of computer vision methods relevant to the proposed system; the proposed architecture is introduced and discussed in section 3, while section 4 gives the conclusions drawn and the future research plan.

1.1 Previous Work

A combination of ultrasonic sensors, laser range-finders, and RFID tags were used [13] for indoor robot navigation without using computer vision. Stereo vision has been used for mapping [16] the data being structured as a 2.5D occupancy, elevation and slope grid. Davidson [17] presented a method to do real-time localization and mapping of the environment using a monocular camera, while [18] artificial landmarks are placed on the ceiling and a vertical looking camera is used to detect their position and orientation and infer the pose of the robot. Visual sonar [21] is a relative new technique attempting to recover depth information from monocular images by deriving cues based on real life constraints used to eliminate the ambiguity inherent to monocular vision.

Due to the large amount of processing required for computer vision, researchers have always tried to employ a whole plethora of methods to improve the localization by using various *pre-defined* or *innate* knowledge about the environment or by aiding the visual localization system with external information. In [1] for outdoor navigation and mapping the computer vision is aided by a differential GPS and location information is processed by a distributed Extended Kalman Filter. Dead-reckoning is used by Cobos et al. [2] beside the Visual Odometer to help with robot localization. In Cluj-Napoca [3] they used a laser beam to detect dynamic obstacles, while a laser scanner has been employed by Biber, Fleck, and Duckett [4] to collect data for model building. High level prior-knowledge of the environment has been employed [15], where the indoor space has been modeled as horizontal and vertical planes having different orientations while the obstacles (objects) have not been modeled only noted in the grid.

In line with our research, Pizarro et al. [5] used a rig of calibrated and synchronized cameras to achieve robot and obstacle localization with a collaborative system employing external cameras, also presenting a mobile robot in [6].

2 Vision Techniques

Having external cameras in the environment where the robot is navigating provides a series of opportunities for better localization compared with a monocular camera robot. In our research we plan to apply a combination of vision based localization methods. The first one is the ability to use epipolar geometry to resolve the mobile camera pose or to precisely map an object position and shape by matching the image seen by the robot with the image seen by the fixed camera. Tracking of moving objects is the second method, where we plan for the external fixed cameras to track robot movement on the observed area and provide external aid for localization. Optical flow of reactive navigation is going to be our fall-back option when the robot has to navigate on an unobserved portion between two observed locations. Lastly, artificial markers can be attached to key points on the unobserved location such that the robot can localize its position by triangulation on the map.

2.1 Epipolar Geometry

In the epipolar geometry technique two or more cameras oversee the same scene from two different points, and they can have different orientations (Figure 1). The point X is projected on the first image plane as x and on the second image as x' respectively. The two points are being related by the equation $x'^{T}Ex=0$, where the Essential Matrix E= $\hat{T}R\ is$ as presented in [7].

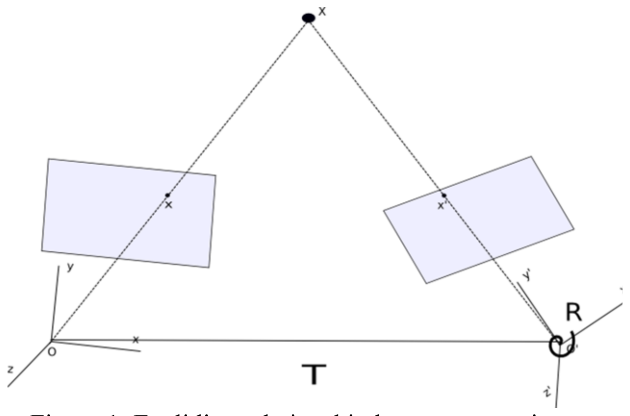

Figure 1. Euclidian relationship between two viewers

Once a minimum of 8 matching pairs of points in the two images have been detected by a feature tracking algorithm the essential matrix E can be computed using the "Eight-point Algorithm" [7]; and from it the position and orientation of robot camera in respect to a fixed camera can be recovered using Singular Value Decomposition (SVD) method [19].

2.2 Tracking

The base station uses images from fixed cameras to detect and track the robots, people, pets, and objects that are being moved in the environment. Fixed cameras enable us to employ a background extraction method. Moving objects are detected by subtracting the current image from a reference background model. In our work we use the *BackgroundSubtractorMOG* provided by OpenCV library which employs a Gaussian mixture-based Background/Foreground Segmentation Algorithm [8].

For each detected object we keep track of the position, movement vector, apparent size and shape, local histogram, and the probability of it being a (particular) robot. When two or more image blobs join into a bigger blob and/or separate, the histogram and previous trajectory, size and shape are used to try to keep their identification. Tracking algorithms assume contiguous trajectory for moving objects and attempts to match the movement of a blob with a robot by comparing the blob movement with the movement reported by the robot.

If prior knowledge about the given robot, such as color, shape and size are already present in the database, this knowledge is also used to update the probability associated with the object.

2.3 Optical Flow Reactive Navigation

Optical flow is a measure of the image changes due to motion during a given time interval *dt*. The optical flow field is the velocity field that represents the motion of the object in 3D space projected on the two dimensional plan of the image. As presented in Sonka et al. [9] it is possible to use optical flow to calculate the real world coordinate x(t), y(t), z(t) of a point from its image projection x'(t), y'(t). When the movement happens along the camera's axis the formulas will be:

$$x(t) = \frac{x'(t)w(t)D(t)}{V(t)}$$

$$y(t) = \frac{y'(t)w(t)D(t)}{V(t)}$$

$$z(t) = \frac{w(t)D(t)}{V(t)}$$

where w(t) is speed, D(t) is the distance of a point from the focus of expansion (FOE) measured in the image with V(t)=dD/dt being its velocity.

Having the ability to measure the distance from the objects in front of the robot mounted camera enables the robot to employ reactive obstacle avoidance and to maintain a prescribed distance from the walls when navigating into a hallway. This method is to be employed when the robot has to navigate alone (without the help of the base station) in an unobserved area located between two observed rooms. Because optical flow is calculated on board it is also used for fast obstacle avoidance; the robot being able to react immediately to avoid collisions autonomously without the time required to communicate with the base station, as shown in Figure 2.

3 Architecture

3.1 Overview

The proposed system consists of a fixed unit (computer system referred to as Base Station) connected to a number of wired or wireless IP cameras overlooking the operating space and one or more mobile units (robots) both navigating inside the building. Each robot is equipped with a (monocular or a stereo) camera, an Inertial Measurement Unit consisting of MEMS accelerometers and gyroscopes and optional other sensors. The robot is capable of dual mode navigation: autonomous (reactive mode) and guided (map based).

The video from the fixed camera is continuously received by the Base Station. The Base Station uses the video streams from fixed cameras to perform object tracking and recognition, and to also maintain the current 3D model of the environment by keeping track of the objects or inhabitants.

The images from the robot camera are processed on board by the embedded computer, which sends, in real time, to the Base Station only a status vector. However the Base Station can request from the robot either the latest image or a specified sub-region of the latest image in order to do epipolar calculations. Upon a successful match the Base Station provides the robot with better position estimates. The robot is continuously doing on board optical flow calculations, which are used to immediately react to potential collisions or to stay on the prescribed trajectory, when out of sight from the fixed camera.

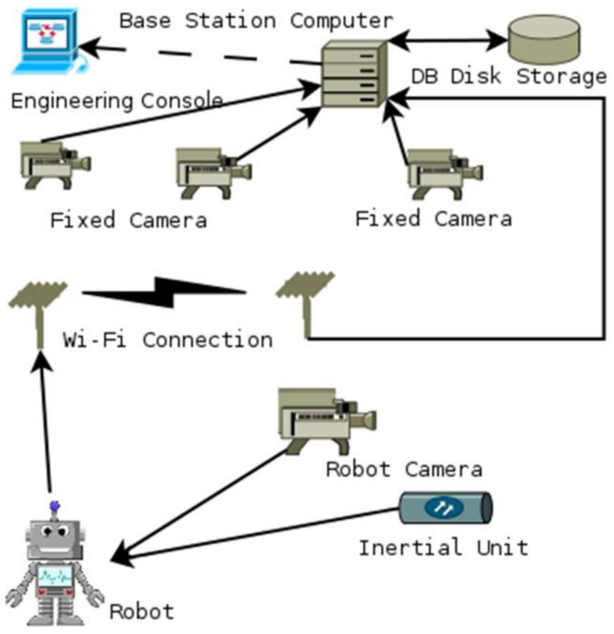

Figure 2. Major Components

3.2 Software Architectures

A modular software architecture (Figure 3) is developed where different vision tasks are assigned to specialized modules.

The images from fixed cameras are processed by Camera Module (CM) with a correspondence of one running module for each fixed camera. The CM is responsible for image pre-processing and tracking of moving objects. At every frame the CM broadcasts to the Situation Awareness Module (SAM) the status of each tracked object into a Tracking Message.

SAM maintains a map of the environment, robots, and non-robot animated entities (humans, pets, or other moving objects), and as much information as the system can gather. When a new moving object is detected SAM can request from CM sub-images of moving objects for deeper analysis and pattern recognition. A database (DB) of images for known or prior tracked objects is maintained for the scope of object recognition.

For each Robot a Robot Module (RM) is also running on the base station. The RM is in constant communication with the software running on the embedded computer on board of the robot which is called an Autonomous Robot Module (ARM). The RM is responsible for mission planning and epipolar localization by matching the image captured by ARM with images requested from the CM. Practically all the robot's visual processing and control software is distributed between base station (RM) and the mobile unit (ARM).

During a typical navigation sequence the RM interrogates SAM for a map of the grid between the current position and the target, and then it uses the Dijkstra algorithm to find the path with the following constrains; that the width along the path be at least a cell wider than the known width of the vehicle. Once the path has been selected, the RM downloads the navigation instructions into ARM and instructs SAM to keep providing real-time tracking information. The robot navigates along the prescribed path driven by a PID controller using the tracking information as feedback. During navigation it is possible that obstacles unknown to SAM may be encountered. For example a table that has never been moved from its place has been interpreted as a pattern on the floor when the grid was built. When approaching the table the optical flow on ARM detects it as an obstacle, the robot stops and relays the information to the RM. The RM can now map the object using epipolar geometry and send the information to SAM to update the occupancy grid. Then the navigation re-starts with a new path planning.

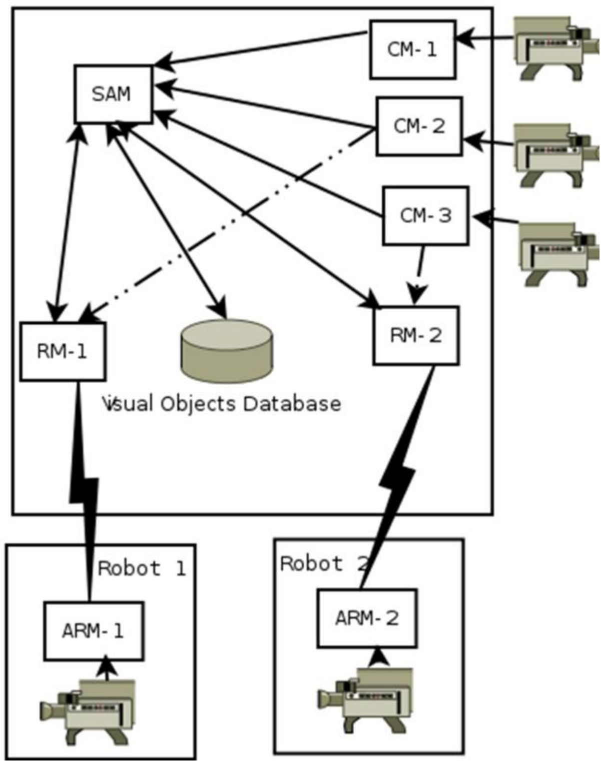

Figure 3. Software Modules

3.3 Protocol

CMs use HTTP or raw TCP to acquire images from IP cameras at the camera's maximum speed (usually 5 to 15 frames/sec.), for each frame each CM sends a *Tracking Message* to the SAM. The tracking message contains for each moving object: an object ID unique for all the systems, position and size of the images, estimated position/size in 3D, image and 3D estimated motion vector, and a status field indicated if this is a new tracked object, and a level of confidence in the tracking.

SAM can send to the CM an *Extended Info Request* for a given blob asking for extra info like the object histogram or texture descriptor. CM also honors a *Blob Image Request* from SAM by sending the image around the blob of interest and a binary mask to differentiate the blob body as tracked from its immediate surroundings.

The RM will send a periodic *RobotStatus* message to SAM informing it where the robot guidance software believes the robot is located and the degree of confidence for this belief, along with current status vector. If SAM determines that a RM is in an observed area but its position does not match any tracked blob SAM can ask RM to run a localization procedure.

The RM can send an *Image Request* message to a CM in order to get a sub-image to perform epipolar localization. The RM can also ask SAM for the tracking status of all the blobs in a specified area in order to match the movement as reported by the ARM with a movement of a blob and therefore identify the correct correspondence between a blob and the robot.

Once the RM is confident about its position matching a given blob, the RM can ask SAM to associate the robot to that blob allowing SAM to provide real-time updates of the robot position based on the tracking.

3.4 Environment Model

SAM is required to keep a live map of the operating environment. The map is represented as an occupancy grid representing the floor. Objects and people are modeled as prisms or an assembly of prisms occupying a particular spot on the occupancy grid. Stairways will be modeled separately once we decide to go ahead with stairs capable robots, but for now they will just be represented as obstacles. The size of the grid cell is chosen to be about the size of the smallest object the robot is supposed to manipulate. For each object, besides the position and surface, a radius of uncertainty is maintained which is expressed in multiple cell sizes, because same objects are not visited closely by the robot causing uncertainty in the position to be higher than the actual cell size. Each object, movable or fixed, will have a unique *id*. The occupancy grid will maintain an *id* of the object located there while each object maintains the position cell of its central axis. Objects larger than a cell will be referenced by multiple cells in the grid. Each grid cell also maintains information about its color, texture descriptors, or a sub image of it. Various meta-data like hazards or roughness can be added as needed for future practical developments.

The objects will be represented as prisms with facets being elevated surfaces from a generator plane. More precisely for each surface we store the Euler Angles (φ, θ, ψ) defining the generator plane in which the elevation grid is located, the center of the elevation grid (x, y, z) and on each cell the height from the plane to the respective surface. Besides the elevation, each cell contains information about the color, texture or sub-image mapping.

To allow the researchers to understand the built model, as well as to improve parsing algorithms, an Engineering Console is being developed. This is an OpenGL 3D viewer being able to connect to SAM and receive the data model that SAM has built. The viewer allows navigation in the virtual environment representing SAMs model with basic keyboard and mouse controls. The data structures presented above can be easily translated in OpnGL by tessellating each cell of occupancy grid or object surface elevation map in two triangles and binding the image as an OpenGL texture on the tessellated surface.

3.5 Objects Database

A robot mission may consist of going to a particular area and retrieving an object. Therefore besides localization we need object recognition abilities in order to be able to perform the task. A disk based object database is maintained where pre-scaled images of objects of interest acquired from various angles are stored; each object is being labeled not only with a name but with the last known location.

While the direct operation “search for object Z” can be accomplished by retrieving from database the set of features recorded for the named object and match it in the image it sees, a much more complicated problem is the reverse problem “label the known objects that are seen”. The reverse problem can arise in landmark based navigation where the robot needs to identify the location by recognizing the surrounding objects and landmarks. In direct problems one object has to be recognized from the objects the robot sees, in the reverse problem the robot still sees M objects and from them it has to recognize the X known objects from a set of N objects it has in its database. Moreover, based on the point of view the projections of an object may differ significantly and occlusions require the ability to handle partial views of each of the projections mentioned above. This is a complicated problem that needs a whole new set of data structures and algorithms. We propose a three tiered database architecture modeled not dissimilar with the memory organization in a human. The first tier (short term memory) keeps in RAM only the objects in direct view. The second tier (mid-term memory) holds objects not currently acted upon but which were recognized in the current session. The second tier memory is stored on a disk while having some pointers kept into the RAM cache. The structure of data in the second tier memory is closer to the first tier in respect that it stores fragments of object images from multiple viewpoints. Because storing multiple views of an object is a space consuming method, the objects that have not been worked with in a long time and were not yet indexed may be “forgotten” to free space. This will require a dynamic self-organizing structure like a Splay-Tree, which brings to the top “memories” freshly accessed. For recognition from the first and second tiers we plan to elaborate on the work about Aspect Graphs done by Ulrich et al. [10], while Shape Masks presented in [11] are investigated as an alternative or an additional method.

The third tier (long term memories) will index the information from the second tier and delete them from it once indexed. The format of long term indexed data will be different from the second tier in order to save space. Therefore at this level no set of sub-images are kept but a more compact representation like Semantic Nets and Voxel based modeling [9] are looked upon.

The big challenge here is to find a Visually Indexed Data-Base system (VIDB) that can provide faster access the same way databases use indexes to accelerate access to records of interests by performing look-up in O(log(N)) time. The object to be recognized is analyzed for a set of vectors of features, each vector of features is being encoded in a binary representation. The binary representation of each vector sampled at a few standard resolutions will form the Visual Index for a given pose of an object. From the image viewed by the robot a set of vectors is estimated and then each of them is searched for in the database to match the stored views.

The algorithm for data reduction and visual indexing is expected to be a very computational intensive task and therefore it will be an off-line algorithm working mostly during the periods when the rest of the system is idle, taking advantage of any spare CPU cycle for this job. This process may be looked at as being somehow similar to the action of dreaming in humans which may be used to sort out and “index” the quotidian experiences [20].

4 Challenges and Future Works

While epipolar geometry is a very powerful technique in practice, it is relatively hard to use especially in our environment where the distance and pose from the two cameras to the scene may vary widely. The feature matching can fail because the two cameras do not see the same scene or because of the encoding noise inherent to IP cameras (JPEG compression). To make the matter worse in the typical domestic or office environment carpet or furniture with repeating patterns are often used, this can trigger false localization by fooling the feature matching algorithm if the robot and fixed cameras overlook the same patterns in different places. To overcome these challenges we do not rely exclusively on epipolar matching but we use it only as a method for better localization when possible.

The pixilation due to encoding noise can easily be confused by the detection algorithm with moving objects fooling the tracking algorithm. To overcome this problem the tracking algorithm needs to be smart enough to figure out that this sort of movement is not compatible with a contiguous trajectory required for moving objects, hence it must be noise.

The experiments performed with tracking algorithm showed that the positions of the detected blobs resulting from background subtraction are not accurate enough for navigation. To overcome this problem a Kalman filter is being attached to each tracked object to provide a sub-pixel estimation of the position. For accurate navigation around objects, the edges of the objects must be precisely identified. For this we are currently experimenting with Watershed segmentation, where the object blobs are being eroded to generate the segmentation markers.

The biggest challenge in blob tracking we encountered is blob segmentation in less than ideal illumination. For example, when the platform vehicle (black with blue) moves farther away from the camera in front of a dark gray piece of furniture, in less than perfect lighting condition the Background Subtractor fragmented the vehicle blob into a set of unconnected blobs. To overcome this challenge we had to modify the data structures to allow for multi-blob objects, and we are experimenting with a fuzzy logic algorithm where every tracked object lay bids on various detected blobs located in the next predicted position. If geometry and fuzzy histograms are not enough to resolve the ambiguity beyond a reasonable doubt a fall back strategy employing feature matching will be employed.

Representing each room with its own occupancy grid can create some problems because of the fact that when the robot is in a position to see both rooms at the same time through the door, the map created by SAM may not be useful for object matching. Using a single occupancy grid for a large building with many rooms creates performance issues. We will be looking for a two level map with eventually overlapping regions in our future research.

The Visual Indexed database is by far the biggest challenge we have to solve in our future work. Decomposing an object in a region and representing the relationship between them into a semantic net combined with Ulrich et al. Aspect Graphs [11] is our best bet, but of course this may change as the research progresses.

References

[1] R. Madhavan, K. Fregene, and L. E. Parker, “Distributed Cooperative Outdoor Multirobot Localization and Mapping,” *Autonomous Robots,* Vol. 17 Issue 1, pp. 23-39, July 2004.

[2] J. Cobos, L. Pacheco, X. Cuffi, and D. Caballero, “Integrating Visual Odometry and Dead-Reckoning for Robot Localization and Obstacle Detection”, IEEE International Conference on Automation Quality and Testing Robotics (AQTR), Cluj-Napoca, May 2010.

[3] A. L. Majdik, I. Szoke, L. Tamas, M. Popa, and G. Lazea: “Laser and Vision based map building techniques for Mobile robot navigation”, IEEE International Conference on Testing Robotics (AQTR), Cluj-Napoca, May, 2010.

[4] P. Biber, S. Fleck, and T. Duckett, “3D Modeling of Indoor Environments for a Robotic Security Guard”, IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshoop, San Diego, June, 2005.

[5] D. Pizarro, M. Marron, D. Peon, M. Mazo, J. C. Garcia, M. A. Sotelo, and E. Santiso, “Robot and Obstacle Localization and Tracking with an External Camera Rig”, IEEE International Conference on Robotics and Automation, Pasadena, pp. 516-521, May, 2008.

[6] P. Chakravarty, and R. Jarvis: “External Cameras & A Mobile Robot: A cooperative surveillance system”, Australian Conference on Robotics and Automation (ACRA), December, 2009.

[7] K. Huang and Y. Ma, “A Survey of Geometric Vision”, *Robotics and Automation Handbook*, CRC Press, 2005.

[8] P. Kaewtrakulpong, and R. Bowden, ”An Improved Adaptive Background Mixture Model for Real-time Tracking with Shadow Detection”, 2nd. European Workshop on Advanced Video Based Surveillance Systems, pp 135-144, 2001.

[9] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis, and Machine Vision*, PWS Publishing, 1998.

[10] M. Ulrich, C. Wiedermann, and C. Steger, “Combining Scale-Space and Similarity-Based Aspect Graphs for Fast 3D Object Recognition”, IEEE Transactions on Patters Analysis and Machine Intelligence 34(10), October, 2012.

[11] M. Marszalek, and C. Schmid, “Accurate Object Recognition with Shape Masks”, *Int J Comput Vis*, Volume 97, pp 191-209, April, 2012.

[12] D. L. Recio, E. M. Segura, L. M. Segura, and A. Waern, “The NAO Models for the Elderly”, 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI), Tokio,pp. 187-188, March, 2013.

[13] S. A. Mehdi, C. Armbrust, J. Koch, and K. Berns., “Methodology for Robot Mapping and Navigation in Assisted Living Environments”, The 2nd International Conference on Pervasive Technologies, Corfu Greece, June, 2009.

[14] Y. H. Wu, C. Fassert, and A. S. Riguad, “Designing robots for the elderly: Appearance issue and beyond”, *Archives of Gerontology and Geriatrics*, Vol. 54 Pp. 121-126, January-February, 2012.

[15] P. E. Lopez-del-Teruel, A. Ruiz, and L. Fernandez, “GeoBot: A High Level Visual Perception architecture for Autonomous Robots”, IEEE International Conference on Computer Vision Systems, pp. 19, January, 2006.

[16] A. A. S. Souza, and L. M. G. Goncalves, “2.5-Dimensional Grid Mapping from Stereo Vision for Robotic Navigation”, Brazilian Robotic Symposium and Latin American Robotics Symposium, pp. 39-44, October, 2012.

[17] A. J. Davison, “Real-Time Simultaneous Localization and Mapping with a Single Camera”, Proceedings of the

Ninth IEEE International Conference on Computer Vision, Vol. 2, pp. 1403, 2003.
[18] C. S. Fernandes, M. F. M. Campos, and L. Chaimowics, “A low-cost localization system based on Artificial Landmarks”, Brazilian Robotic Symposium and Latin American Robotics Symposium, pp.109-114, October, 2012.
[19] R.I. Hartley, and A. Zisserman, “Multiple View Geometry in Computer Vision”, Cambridge University Press, March, 2004.
[20] R. Khosla, and Q. Zhang, “A computational account of dreaming: Learning and memory consolidation”, Cognitive Systems Research Vol. 10, pp. 91-101, June, 2008.
[21] M. C. Martin, “Evolving Visual Sonar: Depth from monocular images”, Evolutionary computer vision and Pattern Recognition Letters, Vol. 27, Issue 11, pp.1174-1180, August, 2006.