ABCM Symposium Series in Mechatronics - Vol. 1 - pp.8-17

RoboPET: A Semi-Autonomous Robot for Hazardous Inspections

Fabio T. Ramos
University of São Paulo, Av. Prof. Mello Moraes, 2231 - CEP: 05508-900 - São Paulo, SP - Brazil
fabioram@usp.br

Marko Ackermann
University of São Paulo, Av. Prof. Mello Moraes, 2231 - CEP: 05508-900 - São Paulo, SP - Brazil
marko_ackermann@hotmail.com

Andre H. Saheki
University of São Paulo, Av. Prof. Mello Moraes, 2231 - CEP: 05508-900 - São Paulo, SP - Brazil
andre.saheki@poli.usp.br

Rodrigo S. Sartori
University of São Paulo, Av. Prof. Mello Moraes, 2231 - CEP: 05508-900 - São Paulo, SP - Brazil

Fabio G. Cozman
University of São Paulo, Av. Prof. Mello Moraes, 2231 - CEP: 05508-900 - São Paulo, SP - Brazil
fgcozman@usp.br

Lucas A. Moscato
University of São Paulo, Av. Prof. Mello Moraes, 2231 - CEP: 05508-900 - São Paulo, SP - Brazil
lamoscat@usp.br

Abstract. *Small robots can be used for inspection in hazardous environments, such as oil rigs or chemical plants. A small autonomous robot, called RoboPET, was developed for remote operations: the robot was designed in an integrated manner and solutions for its mechanical parts, controls, drivers, interfacing, and sensing have been generated. This paper describes all these aspects of the RoboPET project, presenting real tests with the robot. The problem of collision avoidance for a mobile robot in an unknown environment is also discussed; this problem is handled using active vision to identify several features on imagery captured by the robot. When a possible obstacle is identified, the robot analyzes the obstacle using a Bayesian network for inferences. A Bayesian network that can run in real-time was constructed; this network uses Adaptive Conditioning, an anytime algorithm for inference with Bayesian networks. The control scheme uses the inferences generated by the Bayesian network to send commands directly to the robot drivers. The robot can also capture panoramic images using spheric mirrors; optionally, panoramic images can be created as a mosaic of several images captured by the robot. Such panoramic images can be sent to the operator through the Internet; consequently, the robot can be easily tele-operated. The robot' software was implemented in the Java language and uses Java Advanced Imaging and Java Media Framework to support imaging tools and Java Native Interface to access the robot's drivers.*

Keywords. *robotics, probabilistic robotics, artificial intelligence, Bayesian networks, decision theory.*

1. Introduction

Autonomous robots is an important field in artificial intelligence and has been the focus of intense research in the last few years. The operations involved in an autonomous robot include communication, sensing, control, reasoning and learning and have been used in many applications such as underwater exploration, inspections, agriculture, cargo handling and planetary exploration. The need to integrate many different tasks into a single system makes the design and the construction of autonomous robots a difficult and challenging activity.

Most applications of autonomous robots are mobile robots for indoor environments, where robots have to interact with objects, sensing and moving them without previous knowledge about the environment or human supervision. An example of this task is the inspection of factories or power plants after an accident. In Chernobyl and Tree Mile Island for instance, robots have assisted people in diagnosing risks and cleaning up nuclear waste. In such tasks, the robot should move around avoiding obstacles and taking as many pictures as possible so as to provide the engineers a good assessment of damages. With that information, engineers, firemen and the maintenance team can build a better idea of the situation in the hazardous zone and produce a more accurate plan for saving equipment and avoiding explosions.

The robot presented in this paper was designed to operate in situations as the one described above. Called RoboPET, it has a semi-autonomous behavior and was designed to operate in rugged terrain. By semi-autonomous behavior, it is understood that a human supervision is always necessary although the robot can also make its own decisions. Basically the robot is driven by an operator who receives pictures from the robot's environment, broadcasted through the Internet. The operator can send commands to go ahead, stop, turn right or turn left and go back. However, if the robot detects an

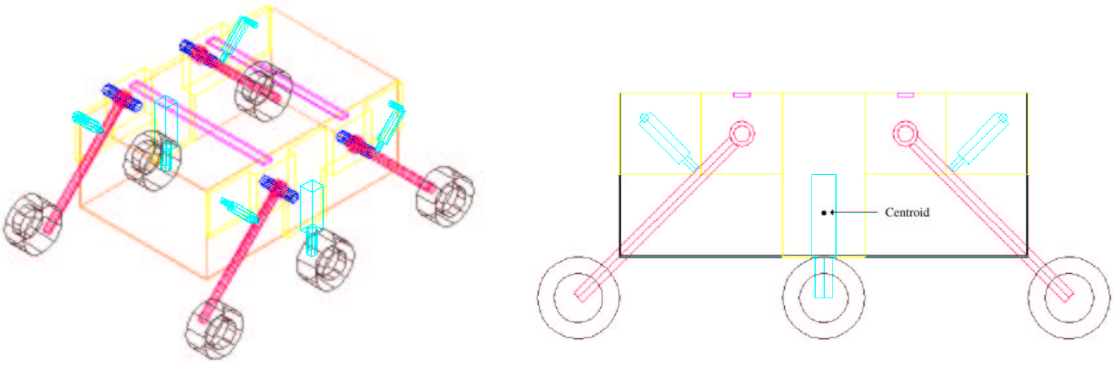


Figure 1. 3D and side views of RoboPET mechanical structure.

obstacle in its path, it will stop, analyze the object dimensions and reach a decision as to whether to overcome it or to divert from it. This process is made automatically, using a probabilistic model to assess obstacles and movements.

Commands, pictures and any other information exchanged between the robot and the operators are broadcasted through the Internet. This fact allows the robot operators to be far away from the place where the robot is running. Also, it permits that more than one site receives images from the robot, making its operation more flexible. On the other hand connections through Internet may have long lags which might put the robot in unsafe situations. For example, a “go ahead command” may have been given to the robot and it would continue going ahead until another command was sent; given the delays in the Internet, the image of a big obstacle or a deep hole in the way of the robot may have not been received by the operator. In this situation, the RoboPET could crash or fall, which could cause damages to the robot. To cope with this problem an automatic system that avoids RoboPET from getting into those dangerous situations was developed. Basically, there exists an image processing system coupled with an automatic decision making. The image processing system grabs imagery and processes it in order to detect obstacles. Once an obstacle is detected, it is measured and located in robot coordinates. The information is passed on to the decision making system that evaluates whether the robot should stop, deviate from or just ignore the obstacle.

Another feature of RoboPET is its ability to get panoramic views from the scene. The robot has a spherical mirror placed above it with a camera pointed out. The imagery grabbed from the mirror is processed and rectified. Thus, the operator receives panoramic pictures of the environment surrounding the robot.

This paper is organized as follows: Section 2 describes the main components of RoboPET. The mechanics, electronics and video devices are presented as well as some important features considered during the design. Section 3 presents the algorithms for collision avoidance and image processing. Section 4 shows how to obtain panoramic pictures from an environment and Section 5 concludes the article.

2. Robot Description

RoboPET was designed to operate in hazardous environments with rugged terrains. This fact makes the mechanical, electronic and sensor designs very important parts of the systems. This section will explain each one of these parts and will give details of the adopted methods.

2.1. Mechanical Design

The vehicle has 6 wheels, all of them with internal motors and gear heads. The suspension system is composed of 4 inclined suspension arms and 2 straight suspension arms. The diameter of the wheels is about 90 mm which, together with the suspension system, makes the robot able to climb obstacles of up to 60 mm height[1] . There are six 4.5 Watts DC motors from Maxon Motors assembled in the wheels with planetary gear heads of reduction of 1118:1. Those motors plus gear heads have a maximum torque of 2 N.m each and a maximum rotation of 12 rpm without load.

One of the main objectives of the mechanical design was to create a light but resistant robot, able to carry a load heavier than its own weight. This fact would decrease the energy needs and its internal batteries could be smaller for the same operation time. Thus, the material selection had an important role for the robot performance. After a careful analysis that took into account mechanical characteristics as well as chemical reactivity, it was decided that aluminum alloys would be used for most of the components. Aluminum alloys have the same mechanical strength resistance with lower mass and lower corrosion potential if compared to iron alloys.

Other desirable features are the capacity to transpose obstacles and to have good stability even in irregular terrains. This fact led to the creation of a vehicle whose centroid is as lower as possible and whose suspension system relieves the

---

[1] This height is for objects with straight lines, which make an angle of 90 degrees with the ground. In other words, it would be able to climb steps of up to 60 mm.

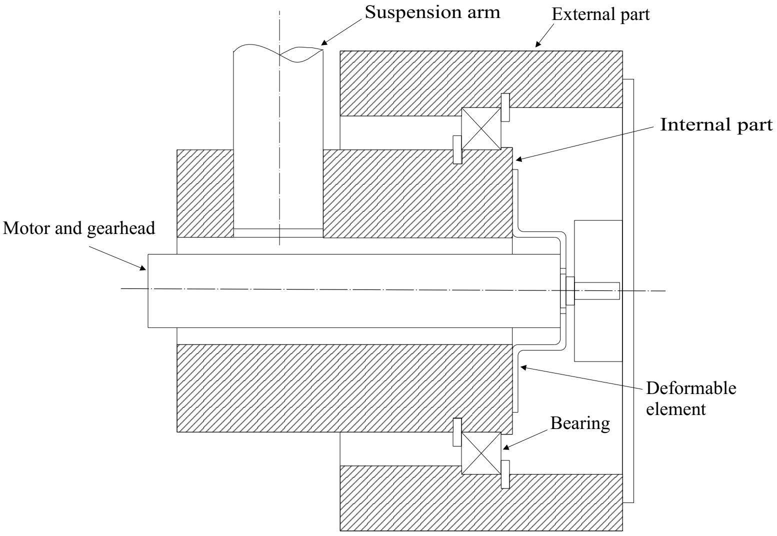


Figure 2. Schematic representation of the wheel.

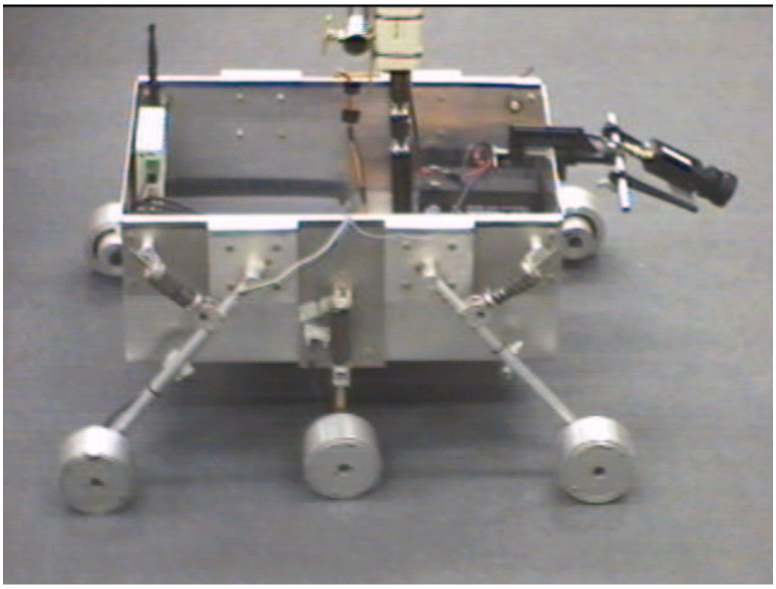


Figure 3. Side view of RoboPet.

inclination of the vehicle related to the ground, helping to keep the stability. The final solution can be seen in Fig.(1). The main body of the robot is hold by the suspension arms fixed at the upper part of body. This fact let the batteries, hard disk and other heavier components of the robot to be placed at the ground of the structure. As a consequence, the centroid is dislocated downwards which increases the whole stability. Figure (1) also depicts a side view of the robot and the centroid position below the point where the arms are fixed to the body.

Another important topic during the mechanical design was the design of the wheels. The wheels should be resistant enough to support the impacts and the stress from the irregular terrain but not transfer those impacts to the motors. As the motor is fixed to the external part by its axle and has its structure fixed to the internal part (see Fig (2)) , every impact or stress from the ground would be transmitted to the motor bearings which were not designed to resist such a load. Besides, every lack of eccentricity during the fabrication process would also increase the stress over the motor bearings that could cause damages or decrease its life expectancy. This problem is addressed by adding a deformable element (made of nylon), that fixes the motor to the internal part of the wheel, Fig (2). This element can transmit the motor torque to the wheel but it deforms if other forces were applied. Thus, axial and radial loads over the motor are smoothed out but the torque is transmitted directly.

RoboPET has four types of motions: to go ahead, to go backwards, to turn right and to turn left. Thus, it is possible to control 2 degrees of freedom: rotation and translation in one axle. However, the robot on an open surface has 3 effective degrees of freedom. To turn right or left, the robot sets the wheels of the same side into opposite directions. To go ahead or backwards, the robot sets the wheels in the same direction. Figure (3) shows the robot with the video and eletronic devices.

2.3. Power Suppply and Motor Drive

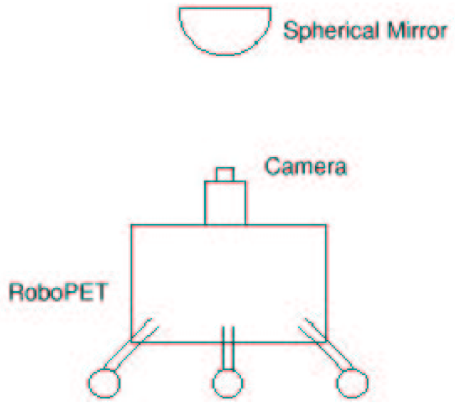


Figure 4. General schema for generating panoramic image.

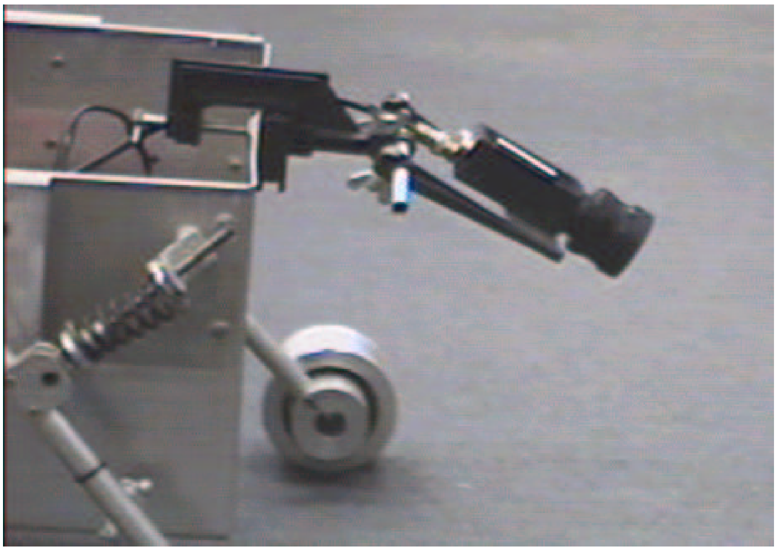


Figure 5. Detail of the camera used for collision avoidance.

The system has three batteries. The first one is responsible for supplying energy to the DC motors. It has a nominal voltage of 7.2V and capacity to store 3000 mAh. With such a capacity the robot is able to run up to 40 minutes without recharging. The battery is constituted by a series association of 6 single nickel-cadmium batteries of 1.2V each.

The second battery is a Lead-Acid battery responsible for supplying energy to the radio, cameras and the gates of the power MOSFETs in the electric driver. It has nominal voltage of 12V and capacity to store 7.2Ah which is enough for more than one hour of operation.

The third battery is exclusive for the robot's computer. It is a Li-ION battery pack of 10.8V nominal voltage and capacity of 2600mAh which guarantees an operation time of around one hour.

The motor drive is quite simple. It has two series of four power MOSFETs arranged in bridge. Each one is responsible for three motors of the same side. That disposition let the robot to go ahead and backward, and turning left or right - in this case, setting the series of motors in opposite directions. Commands are sent from the robot to the motor drive through the computer parallel port. The motor drive has a circuit that interprets the commands and drives the power MOSFETs.

2.4. Video Devices

The robot has two cameras. The first one is a Sony CCD Color with resolution of 640 x 480 pixels. It is pointed out to the spherical mirror above the robot (see Fig (4)) and it is responsible for getting imagery from the mirror which will be transformed into panoramic images. The camera signal is transmitted to the server through a Trango Systems VTX-2400 Eagle radio transmitter. The server has a radio receiver and an internal Pinnacle Studio PCTV that captures the imagery and transmits it to the operator computer.

The robot is also equipped with a CCD (Charge-Coupled Device) gray-scale camera whose lens point diagonally toward the floor, as is shown in Fig (5). With that configuration, the camera captures a rectangular image from the floor so that it is possible to measure the dimensions of the obstacle. The imagery is grabbed with a resolution of 320x240 pixels, in 8-bit gray-scale, through a frame grabber plugged to the embedded computer USB port. The grabber - a Pinnacle Studio PCTV USB - gets the imagery from the gray-scale camera and passes it the internal computer.

2.5. Computational Devices and Software

Basically, the robot system uses three computers: the embedded computer, the robot server and the operator's computer. The embedded computer - a PC K6-II 366Mhz with 32 Mb of RAM running linux (kernel 2.4.9) - is internal to the robot and responsible for receiving commands and passing them to the motor driver. It has a TCP/IP radio link that connects the robot to the local network.

The server - that is also a web server - is a PC desktop connected to the web and responsible for the communication to the robot and for the storage of the web pages and applets that will appear on the operator browser. It has a frame grabber and a radio receiver to capture imagery from the color camera.

The operator computer runs the applets that command the robot and does the image transformation into panoramic views. Depending on the connection and the performance of the operator's computer, one can also request that an applet show imagery from the gray-scale camera. Then, one can check the automatic decisions of the collision avoidance program.

The software of this project was mostly implemented in the Java programming language. The choice for Java was based on its portability and flexibility to work in different machines and operational systems without changes in the code. The embedded computer, server and operator's computer may have different operational systems and even in this situation the whole system would work normally. As Java was created to operate in the Internet, programming of applets and other agents through the Internet is very easy. Java has many methods to deal with security issues and its library has several classes to implement communication between two network machines using the TCP/IP protocol. The only methods implemented in a *native* language were those that command the parallel port of the embedded computer.

3. Collision Avoidance Program

3.1. Image Processing

The image processing technique used here is based on the work of Schneiderman [9] and Duda [2] and considers that the robot is present in a terrain with a single color, like an indoor environment.

The gray-scale camera lens points diagonally toward the floor. With this configuration, the camera captures a rectangular image from the floor so that it is possible to accurately measure the distance between an obstacle and the robot.

In order to reduce the noise in the images, a Gaussian filter is used as the first step in the recognition process. Then, the edges are detected using a Sobel convolution operator, which yields a fairly good result in the environment where the robot is present. The main drawback of the resulting image is the large width of the lines representing the detected edges (with more than one pixel). To further enhance the edge detection, an edge following algorithm is applied (described in algorithm 1.)

Algorithm 1 Edge Following.

```
private void getBound(Raster scr, int i, int j){
    int p,inew,jnew,k=0,i0=i,j0=j;
    int[] ioff={-1,-1,0,1,1,1,0,-1};
    int[] joff={0,-1,-1,-1,0,1,1,1};
    int[] nbd={6,0,0,2,2,4,4,6};
    int[] pixel={175,0,0};
    int[] pix=new int[3];
    int[][] tempResult = new int[76800][2];
    int[][] result;
    do{
      for(int n=0;n<8;n++){
        p=k+n;
        if(p>7) p-=8;
        inew=i+ioff[p]; jnew=j+joff[p];
        pix=scr.getPixel(inew,jnew,pix);
        if(pix[0]==255){
          k=p-1;
          if(k<0) k=7;
          k=nbd[k];
          i=inew; j=jnew;
          scr.setPixel(i,j,pixel);
          break;
        }
      }
    }while(((i!=i0)||(j!=j0)));
  }
```

Algorithm 2 Region Filling.

```
private void compLabel(Raster scr,int i, int j){
    int imax,jmax;
    int[] pixel={255,255,255};
    int[] red={175,0,0};
    int[] pix=new int[3];
    imax = scr.getWidth();
    jmax = scr.getHeight();
    scr.getDataBuffer(), new Point(0,0));
    if(i>=0&&j>=0&&j<jmax-1&&i<imax-1){
      pix=scr.getPixel(i,j,pix);
      if(pix[0]==0||pix[0]==255){
        scr.setPixel(i,j,red);
        compLabel(scr,i-1,j);
        compLabel(scr,i,j-1);
        compLabel(scr,i,j+1);
        compLabel(scr,i+1,j);
      }
    }
  }
```

Furthermore, region segmentation is obtained by filling the region within edges with the simplification that all found obstacles are concave. To obtain this, a point contained inside the edges is used and expanded until it fills the entire obstacle (algorithm 2). The expansion is done analyzing neighboring pixels and determining if the pixel should be inside or outside the obstacle. Figure (6) shows images of a sample image and how are the various stages of processing.

Among these steps, it is possible to obtain important information about the scene, like perimeter of the obstacle as well as its captured area. The obstacle's centroid is easily calculated using the points inside the obstacle at the time the expansion is done. This centroid is used as the obstacle's position in the decision-making process.

It should be noted that the vision system implemented does not use stereo image processing neither movement analysis to obtain 3D information about the scene, although data obtained from the robot's movement will be used in the decision-making process, as a source of data about the location of obstacles.

After the captured images were processed by the method described in the previous section, it is important to know how the obtained information will be used. The approach adopted was to use a Bayesian network [7] to decide the action to be taken by the robot. To guarantee that the network will calculate its inferences in real-time the Adaptative Conditioning [8] algorithm was used.

3.2. Decision Making Process

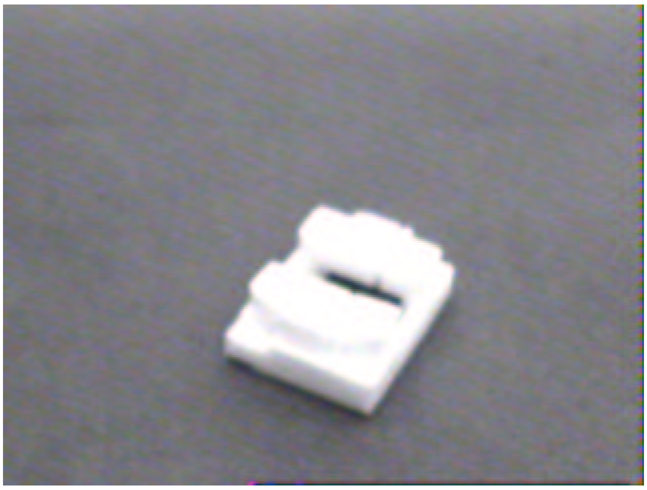
Many algorithms have been used for navigation and collision avoidance in mobile robotics such as [1, 3, 4, 5]. The algorithm presented here, on the other hand, has a different approach. It models the uncertainty related to the sensors and measures through a Bayesian network and draws decisions based on this model. A Bayesian network uses probabilistic models to produce inferences over a set of evidences. For RoboPET, the network should decide which movement to do, based in features (observed variables). The output classes are directions the robot will move, i.e., forward, backward, left, right or stop, the records are the scene descriptions given by the image processing and the features are the detailed information about each scene. The detailed information is comprised of obstacle's locations, mean distance and sizes, combined with data obtained from the previous analyzes. The choosen structure for the network was a Naïve-Bayes, because it would provide a simple model to work with, and also provides reasonable results in most cases. The main characteristic of this network is that there is only one parent node, and all other nodes are children of this node.

The network consider things such as the presence of walls as deterministic information and small obstacles as transposable, meaning that the robot does not need to avoid that obstacle. The diagram of the network is presented in Fig (7).

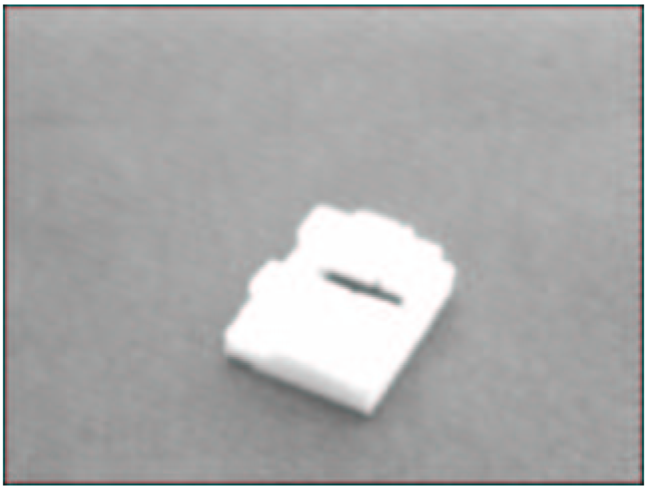
To quantify the probabilities of the network, the robot is trained with a database containing the possible configurations of features. After the training, the robot is ready to explore his environment and, if necessary, receive further training.

4. Panoramic Images

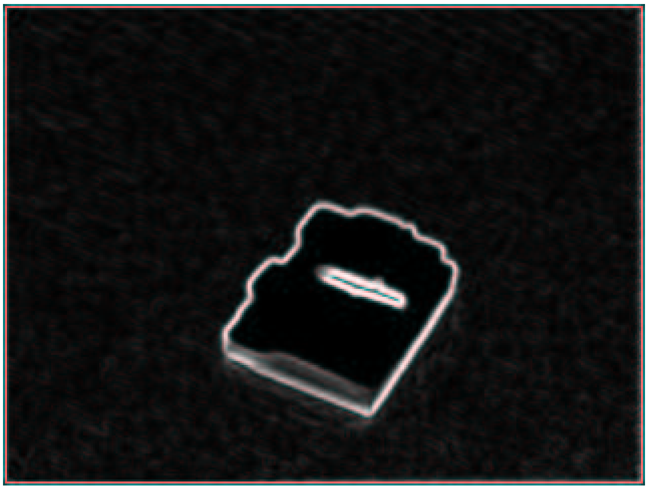
RoboPET is able to get panoramic images grabbed from a spherical mirror placed above the color camera. Those images are rectified to fit in a rectangle, which presents a 360-degree view of the scenery. This process is performed in the operator's computer on his request or it is possible to automatically send the images through the Internet for remote processing. The methods adopted are quite simple and basically involve the selection and translation of pixels from the mirror image into a flat rectangle. Figure (8) presents a general scheme of the system. As one can notice, there is a circular



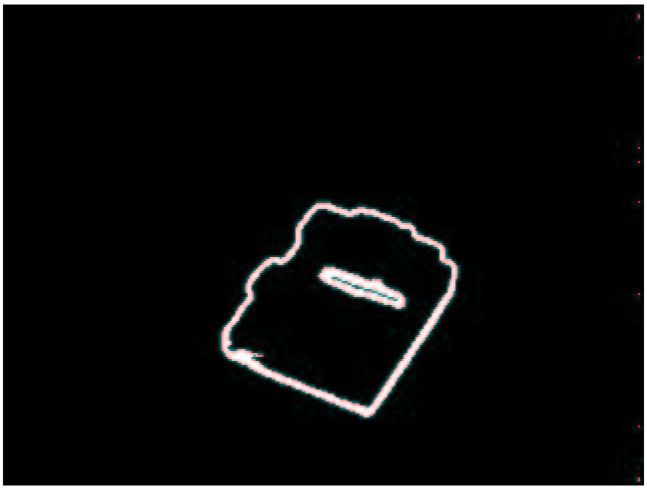
1.Original image



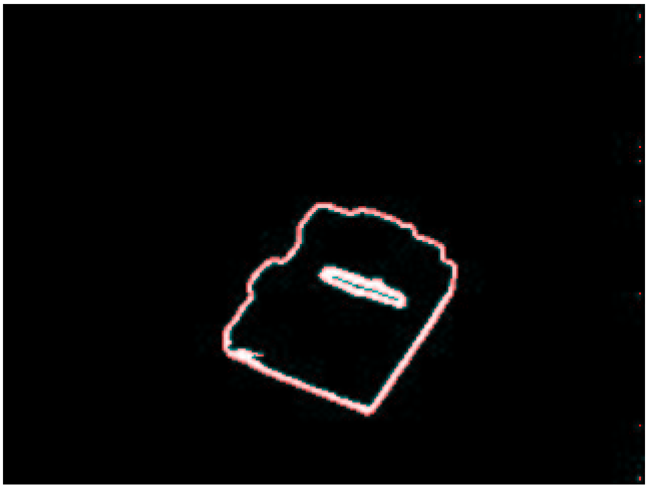
2.After Gaussian filter



3.After Sobel Convolution

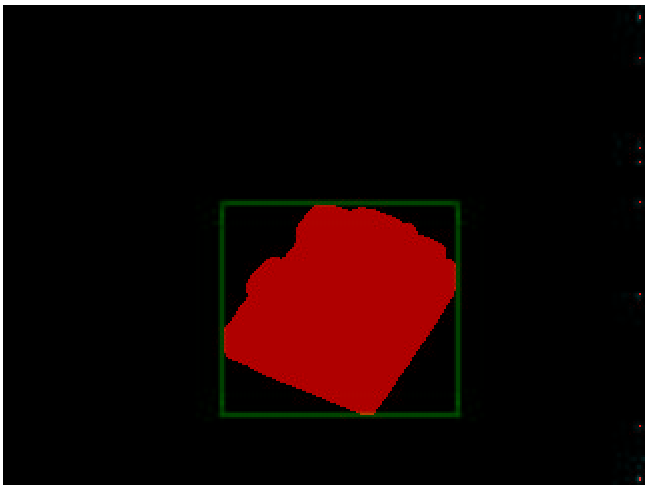


4.After Threshold



5.After Edge following

6.After Region segmentation



7.Final Image

Figure 6. Stages of the image processing.

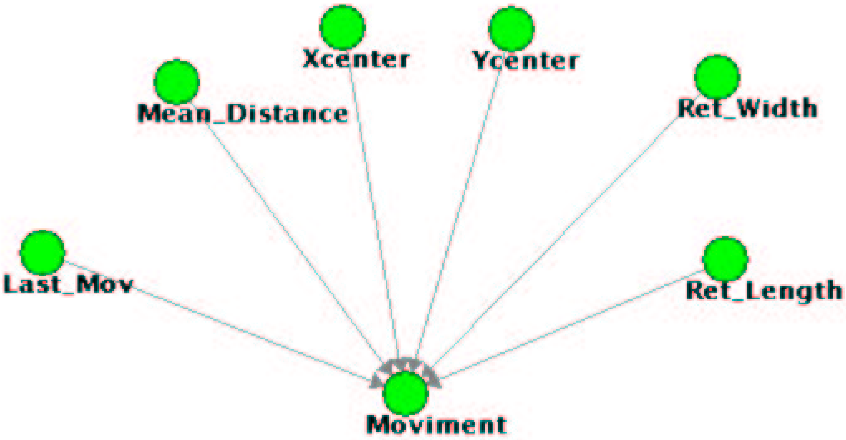


Figure 7. Bayesian network for decision making.

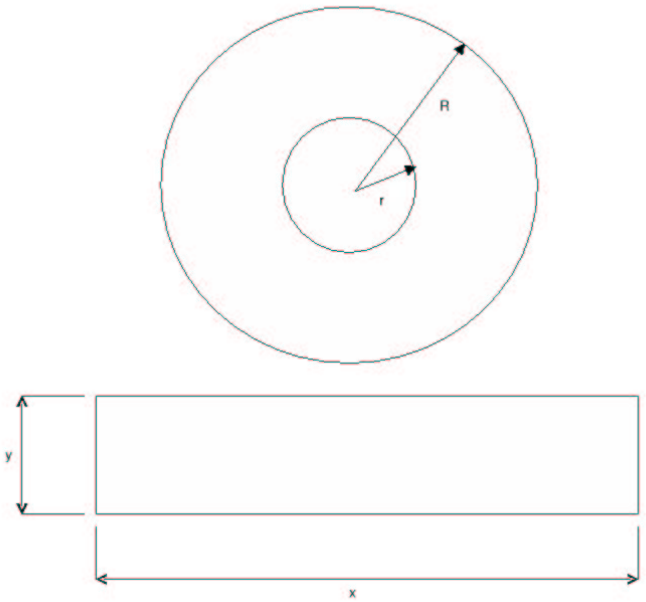
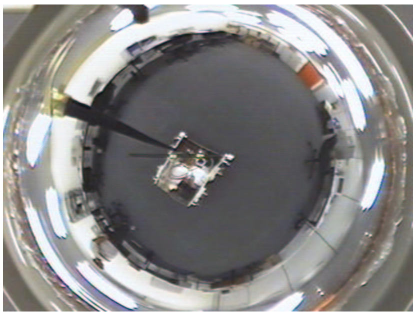


Figure 8. Transforming images into panoramic views.



(a)

(b)

Figure 10. a) The initial image; b) The panoramic after the rectification.

crown that represents a general view of the environment.

It is possible to transform the image into a rectangle, representing the landscape where the robot is. The rectangular image has x pixels of width and y pixels of height while the shape of the circular image can be determined by a radius r and an angle a. The algorithm implemented covers each pixel from the circular image and moves it into the new rectangular one. The frame grabber board used has resolution of 640 x 480 pixels. With this configuration, it was found that the appropriate radius for the crown was 150 pixels. This number of pixels is equivalent to the number of circular layers that will be used when rectifying the image and is the y dimension of the rectified image.

The angle used in the process was 360-degree, to obtain a full panoramic view of the environment. Based on this angle and the previous radius of the crown, it is possible to calculate the most appropriate x dimension of the final image. This calculation is done dividing the area of the crown by the x dimension, and the approximate value of 1000 pixels for the y dimension is used.

Although there are other types of geometries that provide less distortion, the choice of a spherical mirror was due to economical and practical aspects. The other types of mirrors were not handily available at the time of construction of the robot. The study of technical issues in using spherical mirrors was based in a work of Ollis [6].

The software implementation to rectify the image was coded in Java 2 SDK, Standard Edition, v1.3, together with the Java Media Framework v1.2(JMF). The JMF was used in the process of grabbing the images from the camera and later to send the images through the Internet.

Figure (10.a) shows an image from the spherical mirror obtained with the frame grabber and Figure (10.b) the rectified image.

6. Conclusion

This work presented a semi-autonomous robot suitable for operation in hazardous environments. The robot, named RoboPET, was mostly built in aluminum alloys and combines different techniques of image processing not only to provide a wide view of the environment to the operator but also to process imagery internally and to make its own decisions autonomously. The communication through the Internet makes RoboPET very flexible and able to be operated from all over the world.

A method for transforming an image from a spherical mirror into a panoramic view of the environment was implemented in Java. Using some special Java packages such as JMF, an applet running at the operator computer does the transformation and shows a 360-degree picture of the environment.

Other implemented computer vision method captures images from a camera and detects possible obstacles. When a obstacle is detected, their geometric measures are evaluated and that information is passed on to a probabilistic decision making algorithm. The experimental results shows that a probabilistic model such as Bayesian networks can yield accurate decisions even with the uncertainty of the vision system.

6. References

[1] Burl, M.C., Weber, M. and Perona, P. A probabilistic approach to object recognition using local photometry and global geometry. Lecture Notes in Computer Science vol. 1407, p. 628-641, 1998.

[2] Duda, R.O. and Hart, P.E. Pattern Classification and Scene Analysis. Published by John Wiley & Sons, New York, 1973.

[3] Kröse, B. J. A., Dev, A. and Groen, F. C. A. Heading direction of a mobile robot from the optical flow. Image and Vision Computing vol. 18, n. 5, p. 415-424, 2000.

[4] Kundur, S.R., and Raviv, D. Active vision-based control schemes for autonomous navigation tasks. Pattern Recognition vol. 33, n. 2, p. 295-308, 2000.

[5] Nourbakhsh, I.R., Andre, D., Tomasi, C. and Genesereth, M.R. Mobile robot obstacle avoidance via depth from focus. Robotics and Autonomous Systems vol. 22, n. 2, p. 151-158, 1997.

[6] Ollis, M., Herman, H. and Singh, S. Analysis and Design of Panoramic Stereo Vision Using Equi-Angular Pixel Cameras. Robotics Institute, Carnegie Mellon University, Tech. Rep. CMU-RI-TR-99-04, Pittsburgh, PA, January 1999.

[7] Pearl, J., Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Published by Morgan Kaufmann, San Mateo, California, 1998.

[8] Ramos, F.T., Cozman, F.G., Ide, J., Embedded Bayesian Networks: Anyspace, Anytime Probabilistic Inference. In: Proceedings of the AAAI/KDD/UAI-2002 Joint Workshop on Real-time Decision Support and Diagnosis systems, p. 13-19, AAAI Press, Edmonton, Canada, 2002.

[9] Schneiderman, H. and Kanade, T., Probabilistic Modeling of Local Appearance and Spatial Relationships for Object Recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR ’98), p. 45-51, July 1998.