Visual Perception Based Behaviors for a Small Autonomous Mobile Robot

Scott Jantz[†] and Keith L Doty[‡]

[†]Machine Intelligence Laboratory
Department of Electrical and Computer Engineering
University of Florida, USA
Tel. (352) 392-6605
Email: < scott@mil.ufl.edu>
URL: <www.mil.ufl.edu>

[‡]Mekatronix, Inc.
Gainesville, FL
Tel: (352)376-7373
Email: <doty@mekatronix.com>
URL: < www.mekatronix.com>

2000 Florida Conference on Recent Advances in Robotics
May 4-5, 2000, Florida Atlantic University

Abstract

In this paper we describe three vision-based behaviors for mobile autonomous robots. The first behavior is vision based collision avoidance. The second behavior is a tracking behavior, which allows the study of autonomous robots moving in a room. The third behavior is the use of the tracking program to communicate to the robot where it is and how to get to its goal, a type of visual servoing. All of these behaviors are implemented on the TJ Pro™ robot with the Argos™ pan-tilt head. [1] The PC is connected to the robot through RF serial and video connections.

Introduction

The TJ Pro™ [Doty 1999a][Doty 1999b] with an Argos platform [Doty 2000] (Figure 1), coupled with RF data and video links provides an excellent test bed for visual behaviors in autonomous robots. By removing the video processing from the robot we gain several advantages for our research. The first advantage is cost. The robotic platform is low cost and small because it does not have to support the video processor. Another advantage is the ease of processor scalability. If a faster PC becomes available, it can be interfaced to the robot with no changes in hardware or software. Although not covered in this paper, a possible future application is to interface the robot to a distributed network of computers supporting high speed parallel computation for advanced vision processing.

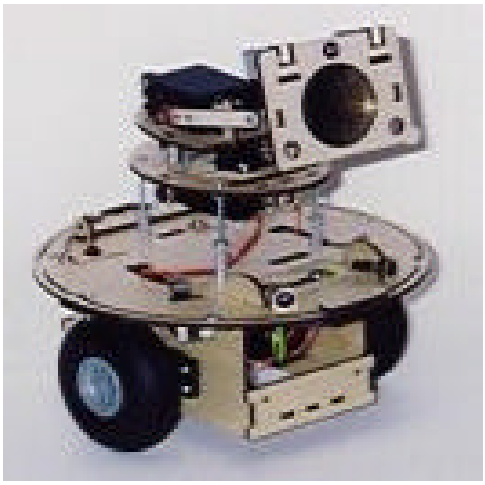

Figure 1: TJ Pro robot with Argos pan-tilt head.

Our use of a high-level computer language (Visual Basic) allows several advantages. The use of Visual Basic first solves the hardware interface problem; any video capture device is

[1] Trademark of Mekatronix, Inc.

automatically detected and used to capture robot video making the robot vision system platform independent. The use of a high level programming language also allows the algorithms to be easily changed for experimentation purposes.

The visual tracking behaviors described here involve a fixed camera mounted to provide a bird's-eye view of the robot and its environment. The visual control loop flows from the camera to the PC and then to the robot via RF links. These behaviors allow the PC to track the robot and either send this information to the robot for use as an extra sensor or to track the behavior of a robot under its own control.

Platform

The basic TJ Pro robot contains a MC68HC11 processor with 32Kbytes of SRAM. The robot has 2 servomotors for locomotion, 2 IR proximity sensors and 4 bump switches, 3 in front and one in back, for contact detection. Robot power derives from 6 AA NiCd batteries. Physically the robots are 3.5 inches high and 6.8 inches in diameter. The Argos pan-tilt head has 2 servos to move the sensor head. One pans the sensor head and the other determines the tilt. On the sensor head there is a single chip color CCD camera, a sonar sensor and 2 IR sensors. Panning of the first servo moves the sensor head in the horizontal plane 180°. Tilting the second servo moves the sensor head in the vertical plane about 120°. On the back of the robot is the RF box, which contains the RF serial, and RF video links.

Hardware Implementation

The video from the color CCD camera is sent to the computer from the mobile robot through a 2.4 Gigahertz analog NTSC four-channel modulator. The voltage required for the camera and the video modulator is 12V. In order to provide 12V from the 6 pack of batteries (7.8V nominal) we use a switching power supply. Regulated 5 VDC is also provided from this board to the RF serial link. The RF serial link is a 20Kbaud spread spectrum 900Mhz radio modem. The choice of 2.4 GHz for video and 900Mhz for data is important. The data channel would interfere with the analog video, if they were in the same band. The spread spectrum data link is noise immune. The analog video, however, employs simple FM modulation and would show noise each time the data channel sends a data packed. The data channel is scalable for multiple robots. Within the packet system there are thousands of channels on which the robots can communicate. In the experiment reported here, there is only one robot on one channel.

The hardware for the 3rd person view experiments differs from the mobile version only in that the camera and video link are not on the robot.

On the computer side, the hardware consists of a data link connected to the serial port. The data link requires no special computer adaptations since the serial link resembles a serial cable between the robot and computer. Software that works with the robot connected with a serial cable will work with the data link. For the video, a down converter is used to demodulate the RF video back into the NTSC signal that came from the camera in the first place. Again the link resembles a direct video cable connection making for ease of testing and implementation. To capture the video we use a Snappy video

snapshot parallel port video capture device.

Software Implementation

Hardware independence was crucial when looking at the software development. In order to accomplish this goal the software calls all hardware through standard windows APIs and controls [Reiser, M. 1999]. The ezVidCap [Ray Mercer 1999] provides a seamless software connection to any video-grabbing device connected to the computer. Connection to the RS-232 port is accomplished with the Visual Basic standard MSCOMM control. By using these features of Visual Basic the code is completely transportable.

Computations begin by reading images into an array. These arrays are then manipulated in matrix operations to the desired end of the algorithm.

Behaviors

Robo-centric Collision Avoidance

For robo-centric behaviors the camera is mounted on the Argos head attached to the TJ Pro platform. We base this behavior on simple frame differences. The justification for working with differentials comes from biology [Carlson 1994] and robotics [Fu, Gonzalez, & Lee 1987]. To implement difference frames, subtract each element in the first frame $f(x, y, t_1)$ from each element in the second and succeeding frames,

$$Df(x, y, t,t_1) = | f(x,y,t) - f(x,y,t_1)| .$$

This simple calculation determines changes in the computer view from each frame to the next from the initial reference frame. If the scene is static, except for the moving robot, the changes represent the current location of the robot and the possible collisions. In the delta frame, $Df(x, y, t,t_1)$, which is referenced to the frame at fixed time t_1, we use several assumptions to simplify the calculations. In the camera image we assume that the bottom half of the frame is *floor* and the top half is *background* due to the robot's close proximity to the floor and the camera position on the Argos pan-tilt head. This assumption allows us to ignore a complex and changing background and focus on objects in the foreground, which are possible obstacles. If the bottom half of the robot's view starts changing then there is probably an object coming towards the robot. To derive information from the floor-half of the image in a form that makes decision

$$sum_{left} = \sum_{x=0}^{X/2} \sum_{y=0}^{Y/2} \Delta f(x, y, t, t_1)$$

$$sum_{right} = \sum_{x=X/2}^{X} \sum_{y=0}^{Y/2} \Delta f(x, y, t, t_1)$$

making easy at time *t*, the robot performs the image calculations

Where X and Y are the limits of the image and sum_{left} and sum_{right} are the resultant numbers, which show where the movement (and hence the object) is most prevalent. The robot simply moves in the direction of least movement, i.e., left if $sum_{right} > sum_{left}$ and right if $sum_{right} < sum_{left.}$ This simple comparison allows the robot to avoid most obstacles. Figure 2 shows the collision avoidance system in action.

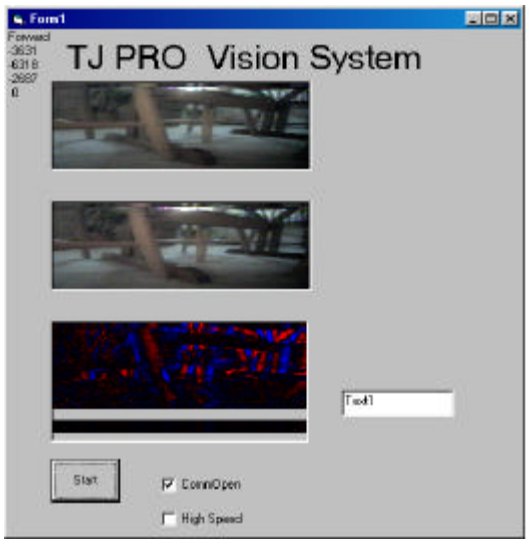

Figure 2: TJ Pro vision system in action first 2 pictures show successive frames. 3rd Image shows the differential image.

Fixed Camera View

Tracking an autonomous robot

The algorithm for tracking a robot in a room scene starts out much like the collision avoidance behavior with a frame differential calculation

$$Df(x, y, t, t_1) = | f(x,y,t) - f(x,y,t_1)| .$$

From the frame differential, the program must locate the robot in the scene. The algorithm assumes only the robot moves in the scene. The algorithm finds the coordinates of the largest blob change in the scene,

$$x(t) = MAX_{j=0}^{j=Y-1}\left(\sum_{i=0}^{i=X-5}\left(\sum_{y=j}^{y=j+2}\sum_{x=i}^{x=i+2}\Delta f(x,y,t,t_1)\right)\right)$$

$$y(t) = MAX_{i=0}^{i=X-1}\left(\sum_{j=0}^{j=Y-5}\left(\sum_{y=j}^{y=j+2}\sum_{x=i}^{x=i+2}\Delta f(x,y,t,t_1)\right)\right)$$

Where X and Y designate the limits of the frame and the function MAX returns the maximum value evaluated between its limits. For use as a visual tracker dots are placed on the reference image where the program has found the robot. Later to make better sense of the data, the program draws colored lines between points to show the direction of travel. Sample outputs are shown in Figure 3, Figure 4, and Figure 5.

Figure 3: Early tracking program output.

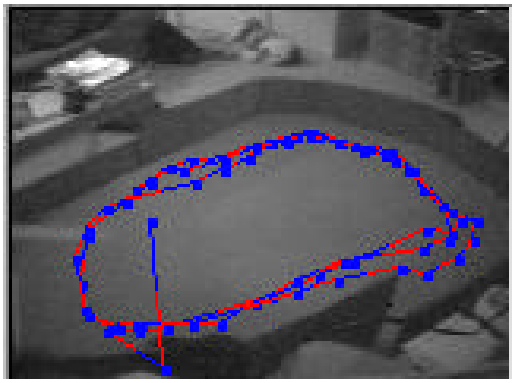

Figure 4: Tracking a robot in an enclosed area executing on-board collision avoidance.

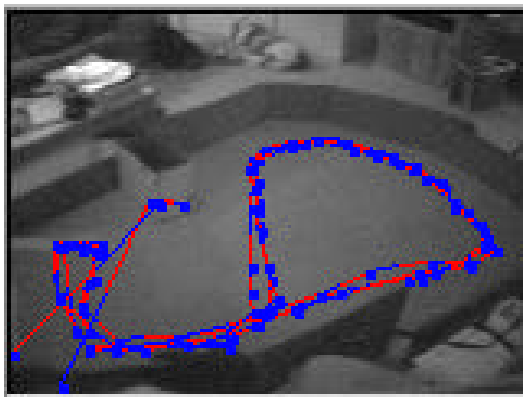

Figure 5: Same robot, same arena, just an added obstacle.

Fixed Camera as Robot Sensor

With the data from the tracking program and the RF serial link to the robot, the robot will now use the fixed camera as a global sensor. For this behavior the behavior algorithm must calculate the distance to a goal point (blue spot for human observers). If either the distance in the x direction is increasing

$$|x_{current} - x_{goal}| > |x_{last} - x_{goal}|$$

or the distance in the y direction is increasing

$$|y_{current} - y_{goal}| > |y_{last} - y_{goal}|$$

then the robot is instructed to turn always right (or left). If the distance in both directions is closing (i.e., the robot is heading in the right direction), then the robot keeps moving straight. Figure 6 and Figure 7 illustrate sample outputs.

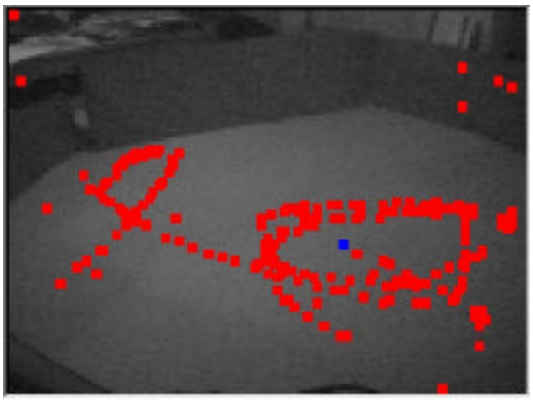

Figure 6. Robot path depicted in red as it moves toward its goal (blue).

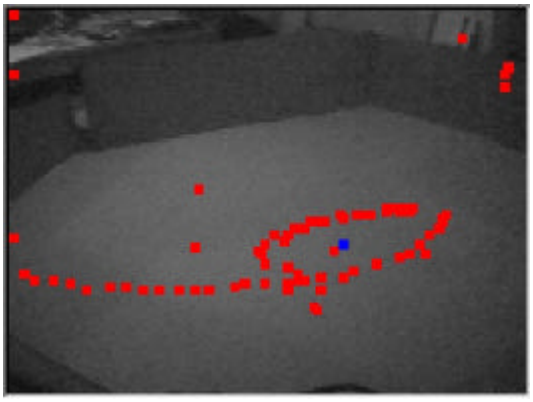

Figure 7: Different robot starting position but same control as shown in Figure 6.

Results

Collision avoidance

Visual collision avoidance worked well in high contrast environments with a featureless floor. This algorithm failed when the robot approached an object with the properties of a long featureless wall where the camera records little to no visual change as the robot approaches. The robot could sometimes see the featureless wall if it caught the intersection of the floor and wall. Some tweaking was still required to set the threshold of avoidance. If this threshold was set too low then the robot would run from every slight shadow, too high and it runs into objects. This threshold also depends on the lighting (contrast) and patterns on objects.

Fixed Camera View Tracking

This program produced superb results. The fixed camera system continuously tracked the robot in a complex environment. In the arena we also observed the performance of robots executing collision avoidance behavior. Researchers can use this program to study the performance of autonomous robot behaviors by recording the information with a color camera and analyzing the data either dynamically, or off-line, with an RF linked computer.

Fixed Camera View with Goal

Visual servoing on manipulators often requires sophisticated mathematics and control. For mobile robots working on a locally flat terrain, implementation of visual servoing based on the view of a "satellite" camera turned out surprisingly simple and direct. The algorithm to perform the robot control was robust, exceedingly simple and worked well.

This program held another surprise, an interesting emergent behavior. The robot would overshoot the goal and circle in like a shark coming in for the kill (Figure 6 and Figure 7). In multiple experiments this biological circling behavior emerged every time when the simple 2-dimensional control was only based on whether the x and y distances to target were increasing or decreasing.

Future Work

Speed optimization of the Visual Basic code will greatly increase performance. Several techniques using WINDOWS APIs and video card hardware should be able to speed this program up significantly. Continuous differential calculation should also improve the tracking programs by changing the reference frame to the immediate past frame instead of a fixed frame at startup time t_1. This approach would increase the program's immunity to light changes and other environmental disturbances.

A clear future goal would be to allow multiple robot tracking and perhaps the playing of robot sports. Multiple robot tracking could be accomplished by using different colored LEDs, or different LED configurations, on top of each robot to identify the robot.

Another future enhancement would combine the vision data from a fixed, bird's-eye view camera with the data from an onboard robot camera to provide the robot with exceptional visual information about the environment for navigation, mapping and control.

References

Carlson, N. R. (1994). *Physiology of Behavior* (5th ed.). Needham Heights, MA: Allyn and Bacon.

Doty, Keith L. (2000). *Argos Assembly and Users Manual*. Mekatronix, Inc.

Doty, Keith L. (1999a). *TJ Pro Assembly Manual*. Mekatronix, Inc.

Doty, Keith L. (1999b). *TJ Pro Users Manual*. Mekatronix, Inc.

Fairhurst, M. C. (1988). *Computer Vision for Robotic Systems: an Introduction.* New York: Prentice Hall.

Fu, K. S., Gonzalez, R. C., & Lee, C. S. G. (1987). *Robotics: Control, Sensing, Vision and Intelligence*. New York: McGraw Hill.

Mercer, R. (1998). EzVidCap [On-line]. Available: http://www.shrinkwrapvb.com/

Petroutsos, E. (1998). *Mastering Visual Basic 6*. San Fransisco, CA: Sybex.

Reiser, M. (1999). *PC-eye Bot / Toro.* [On-line]. Available: www.mil.ufl.edu/imdl