2009 American Control Conference
Hyatt Regency Riverfront, St. Louis, MO, USA
June 10-12, 2009

Low Cost Mobile Robotics Experiment with Camera and Sonar Sensors

Huan Dinh and Tamer Inanc, *Member, IEEE*

Abstract – **The aim of the paper is to demonstrate a design of a low cost mobile robotics experiment using camera and sonar sensors. The experiment is designed to be used as one of laboratory sessions of a course on Fundamentals of Autonomous Robots. This *hands-on* course aims to *foster* students' interests from different fields to autonomous mobile robotics and improve the education in this area. The proposed experiment setup consists of a low cost LEGO Mindstorms Robotic Invention System kit, Handy Board microcontroller board, CMU camera and SRF04 sonar sensor.**

I. Introduction

This paper demonstrates a design of a low-cost mobile robotics experiment to teach fundamentals of mobile robot kinematics; sonar and vision sensors; computer vision; controller design and sensor fusion. The experiment is designed to be used as one of the laboratory sessions of a course on Fundamentals of Autonomous Robots. The proposed experiment uses a low-cost LEGO Mindstorms Robotic Invention System 2.0 kit (a robot building kit), Handy Board microcontroller board, CMUcam camera and SRF04 sonar sensor.

Students will perform the proposed experiment after performing five different experiments where they learn to use touch, light, IR (Infrared) sensors, to program with Handy Board, to design a PID controller, and to construct LEGO based mobile robots. This proposed experiment, the last structured lab experiment of the Fundamentals of Autonomous Robots course, is designed to introduce and teach students how to use vision sensors, fundamentals of computer vision, sonar sensors, how to combine vision and sonar sensors and how to design a controller to improve the efficiency of the system. Finally, students will use the information they learned in this lab experiment for the final robotic competition of the class. The competition, *"Robo-Pong"* originally designed at MIT [7, 17] consists of *unstructured* labs where interdisciplinary teams of three students collaborate with each other to design robots which can play robotic version of ping-pong game during the second half of the semester. Final robotic competition is open to public.

The paper is structured as follows. Section 2 introduces the "tools" that are used in the proposed lab. experiment, for instance, LEGO Mindstorms Robot building kit, Handy Board, sonar and vision sensors. Section 3 discusses modeling of the mobile robot. Section 4 gives information about experiments. Finally, section 5 concludes the paper.

Huan Dinh and Tamer Inanc are with the University of Louisville, Louisville, KY 40292 USA (phone: 502-852-7508; e-mail: t.inanc@louisville.edu).

II. Tools

A. LEGO Mindstorms Robotic Invention System 2.0

A LEGO Mindstorms Robotic Invention System 2.0 kit consists of 718 LEGO elements, one RCX microcomputer, two touch sensors, one light sensor, and two geared motors, enough to design complicated mobile robots. It costs about \$200. In our proposed experiment, Handy Board microcontroller unit [18] is used rather than the RCX microcomputer which comes with the Mindstorms kit. This is due to the limited number of sensor inputs (three) and motor outputs (three) of the RCX unit. Another problem with the RCX unit is that sensor interface for the RCX microcomputer is more complicated than the Handy Board sensor inputs. This very much limits use of simple, inexpensive, off-the-shelf sensors such as light and IR (infrared) sensors with RCX. It is possible to buy LEGO version of these sensors but they are not cost effective. On the other hand, it is easy to interface those sensors with the Handy Board.

B. Handy Board Microcontroller Board

Handy Board shown in Fig. 1 is a Motorola 68HC11 based microcontroller unit. It was originally designed for the 6.270 LEGO Robot Design Competition Course at MIT to be used to control small mobile robots [6, 8, and 17]. The board has several advantages such as; it is small (palm size) and relatively inexpensive, costs about \$300, it has built-in 4 motor outputs and 16 sensor inputs (7 analog and 9 digital),

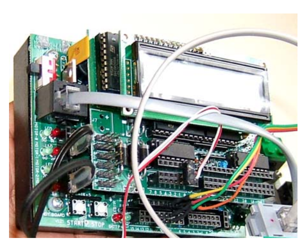


Fig. 1 The Handy Board

LCD display unit, and simple programming. The main disadvantage of the Handy Board is that it uses an 8-bit Motorola 68HC11 microprocessor running with a 2 MHz system clock and it has a very limited memory of 32k bytes. This memory is enough for most of the classroom experiments and small programs but it is not adequate for large programs, and run-time data collection.

Handy Board is programmed using a programming language called Interactive C or IC. IC was designed and implemented by Randy Sargent [12] with the assistance of Fred G. Martin. IC is a subset of C including local and global variables, control structures, pointers, arrays, etc.

C. Vision Sensor: CMUcam Camera

CMUcam is named for a series of 2 versions of camera sensors that have been developed by the Carnegie Mellon University as shown in Fig. 2. The CMUcam's vision system technology achieves onboard real time image processing that is accurate enough to perform educational object tracking experiments. The CMUcam is low-cost, small in size and convenient to interface to microcontrollers and personal computers. We chose to use the CMUcam1 version for our experiments because it is less expensive than the CMUcam2 version and its use has been well documented [16].

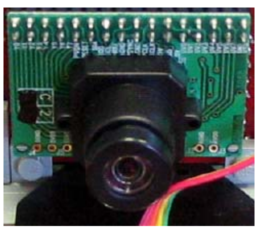


Fig. 2 CMUcam camera, Vision Sensor

1) Features of the CMUcam1:

Readers can refer the camera's full features in its user manual [16]. The main features are as following: track a known object by its color at 17 frames per second; find the location of the object by determining the center of the object's color block; have a capability to track a first object it finds upon start up; have one output port to automatically drive a servomotor to track an object; catch and transfer a raw image.

The CMU camera uses a firmware which can be downloaded through a SX-Key downloader. Normally, the camera firmware is downloaded by suppliers. It is capable to communicate with another processor by a level Shifted serial Port or a TTL Serial Port. With a purpose to use it with a Handy Board, the camera might be modified to use a TTL Serial Port to communicate with either a computer or a Handy Board through a dongle switch.

2) Connection of the CMUcam with the Handy Board:

Connection of the CMUcam with the Handy Board is illustrated in Fig. 3. The dongle switch is an adapter that flips the serial port on the CMUcam between the Handy Board and a computer. Details can be found in [16].

The camera has a set of built-in commands that can be used on its own interface CMUcamGUI, or with a programmable controller. After interfacing the camera with the Handy Board, one can use the IC's commands for the camera. For example, the command `trackRaw(rmin,rmax,gmin,gmax,bmin,bmax)` is calling the command TC on the camera with a custom color in the (Cr,Y Cb) color system. Similarly, the functions `track_size()`, and `track_x()` will return the approximate area and the *x* position of the object, which has been defined by the command `trackRaw()` above.

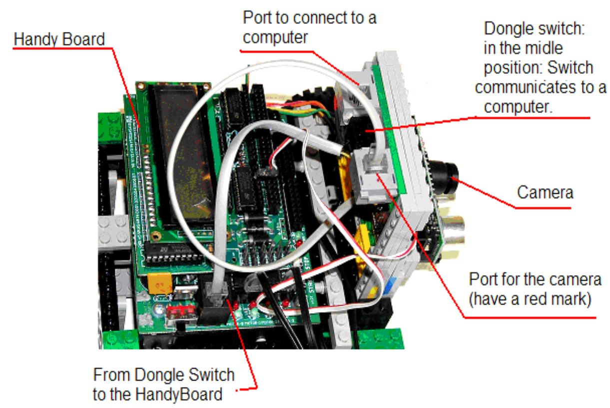


Fig. 3. The Handy Board connected with the CMUcam.

In our experiments with the camera, we practically use those functions to track an object which is an orange color ball as described in the following sections. The appropriate color data of the Orange are acquired by using the interface CMUcamGUI. The built-in functions of IC, `track_size()` returns the area *A* of the ball, and `track_x()` function will return the horizontal displacement *y'* of the ball.

D. SRF04 Sonar Sensors (Ultrasonic Rangefinders):

Sonar sensor is a timed analog sensor which sends a ping of high-pitched sound and listens for an echo. This high-pitched sound wave pulse travels through the air at about 1.125 feet per millisecond (the speed of sound). When it hits an object then it bounces back. The Handy Board commands the sonar to send a "ping" and measures the time it takes to receive an "echo". From the time difference between the ping and echo, and from the speed of sound, distance to the closest object in the field of sonar's view can be calculated. The Devantech SRF04 sonar sensors shown in Fig. 4 are used in the proposed experiment since they are reliable, easily connected and more cost effective sensors compare to their alternatives, such as Polaroid 6500 sonar sensor.

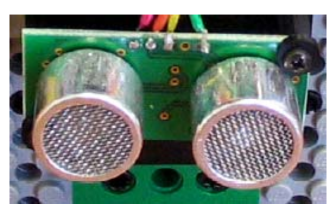


Fig. 4 The Devantech SRF04 Sonar Sensor

The range of SRF04 sonar sensors is approximately 1.2 inches to 3.3 yards (3 cm to 3 m). The field of view of the sonar is about +22.5 and -22.5 degrees around the centerline from the sonar. More details and how the sonar is connected to the Handy Board can be found at `www.robot-electronics.co.uk/htm/srf04tech.htm` and at `www.acroname.com/robotics/info/examples/srf04-3/srf04-3.html`.

SRF04 sonar sensors or range finders are not ideal devices. They have some disadvantages such as they are limited in resolution, range, and the size of object they can detect. They also might detect false echoes or distance returned by the sensor which may not correspond to the actual distance to the object. This is especially true for indoor environments where the "ping" sound wave might get reflected from multiple objects. Simple solution to this problem is to average several sonar readings.

III. A Model Of The System: The Mobile Robot And The Camera Sensor.

The proposed low-cost differential drive autonomous mobile robot is shown in Fig. 5.

A. *The Robot's Movement*

This section will briefly review few basic simple calculations for the differential driving system that have been reported before by many authors [2, 13].

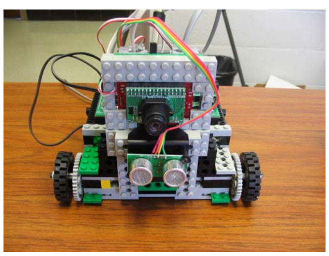


Fig. 5 Low-cost Autonomous Robot with Sonar and Vision Sensors.

The robot has two wheels placed in front and a caster wheel at the back side. These two front wheels are powered and controlled independently to perform the robot's movement. Fig. 6 shows a diagram of the two front wheels with its axle. Distance between the two wheels is denoted by *b*. A local coordinate system has its origin *O* at the middle point of the axle. The *y* axis is along the axle to the right wheel. The *x* axis is perpendicular to the axle, and it points out from the front side of the mobile robot.

s_L, s_R, s_O denote the displacements of the left wheel, the right wheel, and the middle point *O* of the robot, respectively. Thus, the displacement of the robot is s_O, which is an average value of s_R and s_L as given in Eqn. (1).

$$s_O = \frac{s_L + s_R}{2} \qquad (1)$$

If $s_L \neq s_R$, the robot will turn with an angle θ. From Fig. 6 we have:

$$\tan\theta \cong \frac{s_R - s_L}{b} \qquad (2)$$

Assuming that θ is small, then $\theta = \frac{s_R - s_L}{b}$

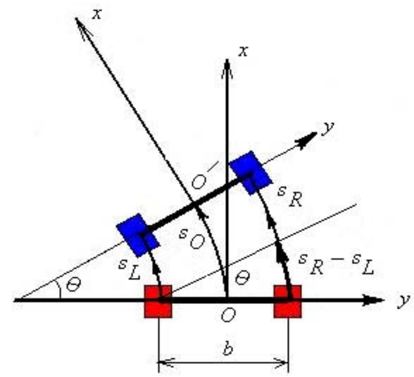


Fig. 6 The differential drive robot's movement diagram.

B. *The "Tracking of an Object" Problem*

Assume that there is an object placed on the *x* axis at a location *B*, with a distance *L* to the origin *O, the middle point of the robot*. If the object moves to a location *B'*, from the point of view of the robot at *O,* the new location of the object has a distance *L*' to the origin *O,* and the angle between OB' and the *x* axis is θ', measured counter clock wise direction from the x axis. Now, to track the object, the robot needs to make a displacement s_O to new location *O'* to keep a distance *L* to the object as shown in Fig. 7.

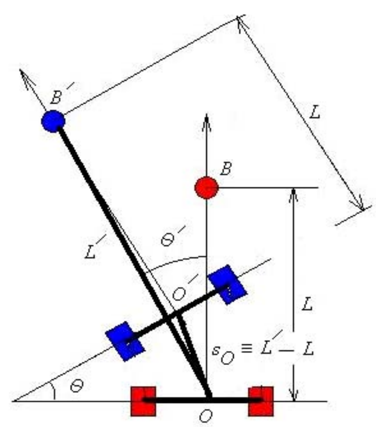


Fig. 7 The robot's tracking object diagram. Red color is the prior position of the robot. Blue color is the future position after robot moves.

For a small turn of the robot, since θ is small enough; described the robot motion can be approximated as "point-and-shoot" movement [2]. Displacement s_O equivalent to the difference between *L'* and *L:*

$$s_O = L' - L \quad \text{and} \quad \theta = \theta' \qquad (3)$$

From Eqns. (1), (2) and (3):

$$\frac{s_L + s_R}{2} = L' - L \quad \text{and} \quad \frac{s_R - s_L}{b} = \theta' \tag{4}$$

After solving Eqn. (4) for s_R and s_L:

$$s_R = \frac{b}{2}\theta' + (L' - L) \quad \text{and} \quad s_L = -\frac{b}{2}\theta' + (L' - L) \tag{5}$$

Thus, if we set up the distance to the object L *(desired distance between the robot and the object),* and the distance between two wheels b, then a camera sensor can be used to acquire L' and θ', which is explained in details below. The needed displacements for the robot's wheels to track the object can be calculated.

C. Using A Camera Sensor to Acquire L' and θ'

The camera sensor CMUcam returns two values from the object as mentioned in the above sections:

- the horizontal displacement of the object y' which ranges from -40 to 40 pixels (the image from the camera has resolution of (80 x 143) pixels) and
- the area A of the object which is the number of pixels occupied by the object in the camera's image.

An orange ball is used as the object of interest. The movement of the object is limited in 2D. The horizontal displacement y' along the y-axis is known by the returned value of the built-in function `track_x()` of the CMUCam camera as shown in Fig. 8. The depth displacement is approximated from the calculation of the area A which is returned from the function `track_size()` of the camera as illustrated in Fig. 9.

1) Calculation of θ' from y':

The camera sensor is placed at the origin O of the wheels' axle. An object will be placed on the x axis at a distance L. From simple experiments, we can get the maximum angle, θ_{max}, that the camera sensor could still see the object. Thus, L and θ_{max} are approximately known constant numbers. If we move the object parallel to the y axis at a distance y' as in Fig. 8, we have:

$$\frac{y'}{L} = \tan\theta' \quad \text{and} \quad \frac{40}{L} = \tan\theta_{\max} \tag{6}$$

The constant number 40 is the maximum angle θ_{max} in pixels in the horizontal direction.

$$\text{From Eqn. (6)} \Rightarrow \theta' = \tan^{-1}\left(\frac{y'.\tan\theta_{\max}}{40}\right) \tag{7}$$

Note that the unit of L is not the same as the unit of y'. The L' unit is the length unit (mm), and the unit of y' is pixel. However, this unit (pixel) will be cancelled out with 40 (pixels) in the denominator, the units' difference will not affect the result of θ'.

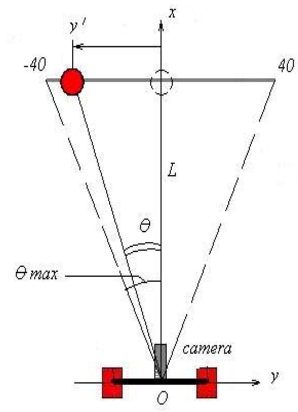


Fig. 8 Calculate θ' from y'.

2) Calculation of (L'-L) from the Area A:

An area of an object, which is acquired from the CMUcam camera, is inversely proportional to the distance between the object of interest to track and the camera. From Fig. 9, it is seen that the object, was initially placed at a distance L to the camera. The camera acquired the object's area A in its image1. When the object move to a distance L', then the camera acquired the object's area A' in its image2. Again, note that the unit of those areas is pixel. Then, if we project the area A from the image1 into the image2, the area seems to be changed by a coefficient which is a ratio of the two distances, becomes the area A_1 as shown in the Fig. 9. In fact, the camera will see the value of A_1 in the image2 same as A in the image1.

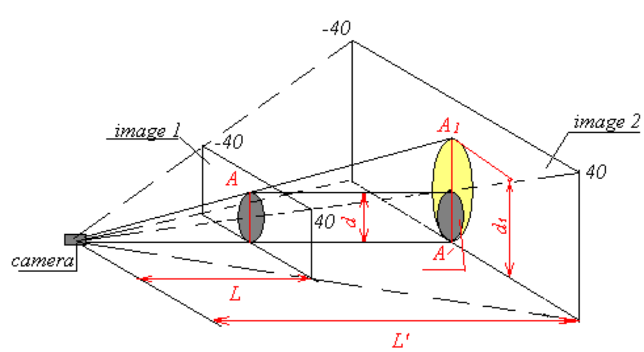


Fig. 9 Calculate *(L'-L) from the area A.*

$$A_1 = A \tag{8}$$

Call d and d_1 are diameters of A and A_1 , respectively. From Eqn. (8) we have:

$$\Rightarrow \frac{A'}{A_1} = \frac{\pi\left(\frac{d^2}{4}\right)}{\pi\left(\frac{d_1^2}{4}\right)} = \left(\frac{d}{d_1}\right)^2 = \left(\frac{L}{L'}\right)^2 = \frac{A'}{A}$$

$$\Rightarrow \frac{L}{L'} = \sqrt{\frac{A'}{A}} \Rightarrow L' = L\sqrt{\frac{A}{A'}}$$

$$\Rightarrow L' - L = \left(\sqrt{\frac{A}{A'}} - 1\right)L' \qquad (9)$$

D. Using the Camera Model in IC Programming.

Finally, the tracking of an object problem is modeled by the system of the linear equations given in Eqn. (5).

$$s_R = \frac{b}{2}\theta' + (L' - L) \quad \text{and} \quad s_L = -\frac{b}{2}\theta' + (L' - L)$$

where s_R and s_L are the displacement functions for the displacements of the right wheel and the left wheel, respectively. Θ' and (L'-L) are the variables that are obtained from the equations (7) and (9).

The constants are b, L, A, and Θ_{max}. The data, that are obtained from the camera sensor, are y' and A'. The following commands are used in the program to set the powers applied for the motors:

```
Right_Motor_Power = Ktheta*(b/2)*theta
+Kl*(DeltaL);

Left_Motor_Power = - Ktheta*(b/2)*theta
+ Kl*(DeltaL);
```

where, `theta`= Θ', `DeltaL` = L'-L. `Ktheta` and `Kl` are the proportional controller gain constants which are adjusted by trial and error. It is important to mention that the students taking the class are not required to have a background in control theory. Therefore, motor powers are calculated in a very simple way. Obviously, one can add more control theoretic background to the experiments given below with quantifying the system, sensor, actuator dynamics and designing the controllers such as PID, LQR, LQG or any other type of modern feedback controllers.

IV. Experiments

In our programs, the camera and the sonar sensors are initialized and controlled by using built-in IC functions. The functions for the CMUcam camera are stored in the `cmucamlib.ic` in the IC library, thus the program has to declare using of that library at first;

```
#use "cmucamlib.ic"
```

Then, the program should include these following commands to initialize the sensors:

```
init_camera();
clamp_camera_yuv();
sonar_init();
```

First, `init_camera()` function is called to initialize the camera. Then, `clamp_camera_yuv()` function is called to automatically set the camera white balance for the lighting conditions in the environment. To declare the object, the `trackRaw(ranges of the object’s color)` function is called to track an object of interest which is an orange ball in the following experiments. This function tracks a blob with orange color and returns a confidence level. A confidence level of 80 and up is a good indication that there is an orange color object in the scene. (Students will be given information about fundamentals of image processing, color images, thresholding, blob tracking so that they just do not end up using the already built-in code of CMUcam, instead they understand the concept).

Three different experiments were performed to test the proposed experiment with camera and sonar sensors. It is important to mention that in all of these experiments, the primary movement of the robot is *not* “pivot and forward” type movement to change its direction. The robot changes its direction continuously according to the sensor information. Therefore, the robot is able to turn and change its direction smoothly.

Experiment 1: Use of the camera sensor only to track an object.

In this experiment, information only from the camera sensor used to track an object. Sonar sensor is not used. The motion of the object in 2D, both along sideways (robot moves toward its right and its left) and along in depth (robot moves toward front and back) were measured. As shown in Fig. 10 the robot was able to track the object successfully.

Experiment 2: Use of both of the camera and sonar sensors to track an object.

In this experiment, both camera and sonar sensors were used to track the object. Sonar is used to calculate the depth distance from the object and hence it is helping the camera to track the object. The motion of the object along sideways was measured and tracked by using camera sensor information. On the other hand, the motion of the object along in depth was measured and tracked by using sonar sensor information. This way, a better tracking under poor lighting condition than the experiment 1 was achieved. It is also difficult to find depth information using a single camera. Snap shots of this experiment are shown in Fig. 11. As it is seen in this figure, robot moves closer toward the ball at t=27sec., then it corrects its orientation, and gets closer at t=29sec. and finally it corrects the distance between the robot itself and the object according to sonar readings at t=30sec.

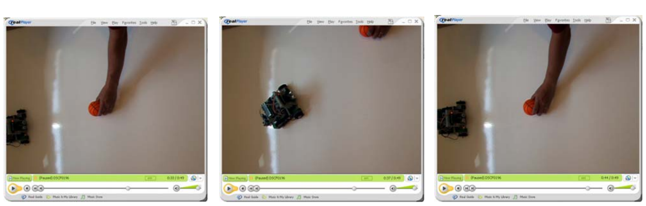
Experiment 3: Use of both of the camera and sonar sensors to track an object and avoid a transparent obstacle.

In this experiment, object tracking and avoiding a *transparent* obstacle was performed. We deliberately chose a transparent object so that camera can still see the object behind the obstacle but it cannot distinguish much the transparent object itself. Tracking in sideways and depth directions are both done using only the information from the CMUcam. Detection and avoidance of the transparent obstacle, which is hard to be noticed by the camera sensor, was performed by using the SRF04 sonar sensor information. As shown from the snap shots of the experiment in Fig. 12, robot stops in front of the transparent obstacle at t = 8 sec. even though sensor information from the camera tells it to go forward. Then, robot backs-up according to the sonar sensor information to avoid obstacle at t = 9 sec. At t = 10 sec., *robot estimates the direction of the movement of the object* from the previous frames and makes a decision to turn toward left to avoid obstacle and still be able to find and track the object. During this time, the object is out of view of the camera sensor mounted on the robot. Then robot makes a right turn at t = 12 sec. and avoids the obstacle. It finds the object at t = 13 sec.

V. Conclusions

We have demonstrated design of a low cost experiment combining sonar and vision sensors which will be used as one of the laboratory experiments of our Fundamentals of Autonomous Robotics course.

After performing the proposed experiment, students working in multidisciplinary teams will understand concept and difficulties of using and combining sonar and vision sensors and implementing them on mobile autonomous robots. Hence, they will be exposed to several important engineering subjects such as mobile robot kinematics; sonar and vision sensors; computer vision; controller design, importance of well structured software programs, and sensor fusion.

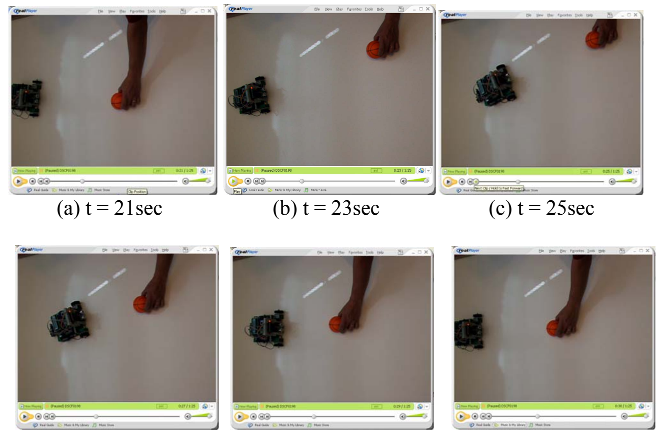


a) t = 33sec. (b) t = 37sec (c) t = 44sec

Fig. 10 Experiment 1: Use of the camera sensor only to track an object.

Acknowledgements

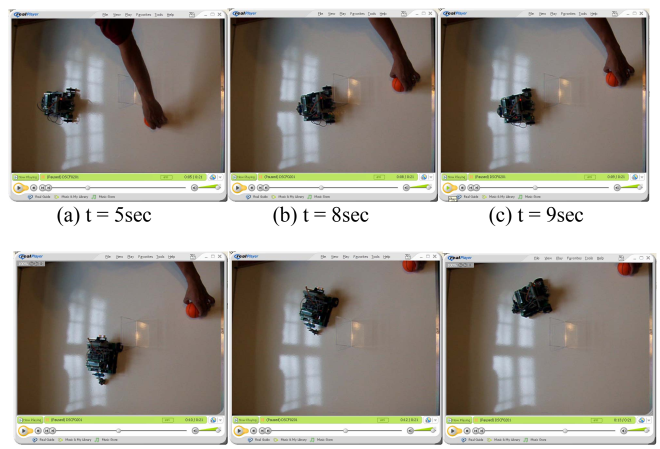
This work is supported by University of Louisville Research Initiation grant.



(a) t = 21sec (b) t = 23sec (c) t = 25sec

(d) t = 27sec (e) t = 29sec (f) t = 30sec

Fig. 11 Experiment 2: Use of both of the camera and sonar sensors to track an object.



(a) t = 5sec (b) t = 8sec (c) t = 9sec

(d) t = 10sec (e) t = 12sec (f) t = 13sec

Fig. 12 Experiment 3: Use of both of the camera and sonar sensors to track an object and avoid a transparent obstacle.

References

[1] J. Borenstein, H. R. Everett, and L. Feng. *Where am I? Sensors and Methods for Mobile Robot Positioning*. The Univ. of Michigan, 1996.
[2] F. G. Martin. *Robotic Explorations*. Upper Saddle River, New Jersey: Prentice Hall, 2001.
[3] F. G. Martin, *"IDEAL AND REAL SYSTEMS: A Study of Notions of Control in Undergraduates Who Design Robots"*, Y. Kafai and M. Resnick (Eds.), Constructionism in Practice: Rethinking the Roles of Technology in Learning. Presented at the National Educational Computing Conference, Boston, MA, June 1994. MIT Media Laboratory, Cambridge, MA.
[4] F. G. Martin. The 6.270 Robot Builder's Guide. MIT Media Lab, Cambridge, MA, 1992.
[5] R. Sargent. Interactive C Software, 2004. www.kipr.org/ic/
[6] R. Siegwart and I. R. Nourbakhsh. *Introduction to Autonomous Robotics*. A Bradford Book. The MIT Press, Cambridge, MA, 2004.
[7] CMUcam camera. www.cs.cmu.edu/~cmucam/
[8] 6.270 Autonomous Robot Design Competition. web.mit.edu/6.270/
[9] Handy Board microcontroller board. www.handyboard.com