Student Name: Anne Harmeson
TA: Louis Brandy
William Dubel
Max Koessik
Instructor: A. A. Arroyo

University of Florida
Department of Electrical and Computer Engineering
EEL 5666
Intelligent Machines Design Laboratory
Spring 2004

Cyclops Special Sensor Report:

XCam2

Table of Contents

Special Sensor

Introduction **3**

Sensor Specs **3**

Sensor Integration **4**

Problems **5**

Sensor Suit

Infared (IR) **7**

Bump **7**

Radio Frequency (RF) **8**

References **10**

Introduction

The XCam2 is a wireless web camera, used with the XCam2 Battery Pack, Small Audio/Video Receiver, and USB Video Capture Adapter. With the camera mounted on the robot, this system allows the robot to capture video data and process the data externally on a Laptop. My goal is to use the XCam2 system to allow my robot to sense a bright color and then track it, much like the CMUcam does. The XCam2 system is special because the Laptop is a more powerful processing tool than the robot's micro controller or the CMUcam, and I get to write my own image processing code. My extra special goal is be able to track movement, not just a particular color.

Sensor Specs

The XCam2 system was purchased from X10.com. All the components are sold separately, but X10 offers a special wireless kit that includes a camera, receiver, and USB adapter for a special price of $130. In addition, X10 throws in a bunch of other wireless toys, because the kit is so special. I also received a Fire Cracker Computer Interface, Multi-Camera Remote, 2 Wireless Transceivers, and an XCam2 Battery Pack . Figure 1 is a picture of the Camera, Receiver, and USB adapter, all take from the X10 website.

Figure 1: XCam2, Video Receiver, USB Adapter

The kit comes with instructions on how to set up the camera on your computer and the USB driver information must be downloaded from the X10 website. The website also has a free program view the captured video, Xray Vision, but it was of no use to my project. The wireless transmitter/receiver frequency of the camera system is 2.4 GHZ. The battery pack of the camera requires 4 AA batteries.

The XCam2 captures video in Jpeg format files. Using a simple program to get the capture device info, I found the following formats available:

- size=320x240, encoding=yuv, maxdatalength=115200
- size=160x120, encoding=rgb, maxdatalength=57600
- size=176x144, encoding=rgb, maxdatalength=76032
- size=320x240, encoding=rgb, maxdatalength=230400
- size=352x288, encoding=rgb, maxdatalength=304128
- size=640x480, encoding=rgb, maxdatalength=921600
- size=160x120, encoding=yuv, maxdatalength=28800
- size=176x144, encoding=yuv, maxdatalength=38016
- size=176x144, encoding=yuv, maxdatalength=38016
- size=352x288, encoding=yuv, maxdatalength=152064
- size=640x480, encoding=yuv, maxdatalength=460800

Sensor Integration

The XCam2 is mounted onto the robot, so that it can "see" facing forward. The video receiver with USB adapter plugs into my Laptop. My laptop is an IBM with the Windows XP operating system. A Java application running on the Laptop captures the video data from the receiver. Another Java application analyzes the data and then produces intelligent commands to send back to the robot. Commands are sent back to the robot via RF sent through the laptop's parallel port.

I chose to develop the applications in Java and Windows because I am most familiar with the Java programming language and the Windows operating system. The Java 2 Platform Standard Edition, version 1.4, the Java Media Framework (JMF) version 2.1.1, and the Java Communications API are installed on my computer. The JMF API enables video-based media to be added to applications built on the Java technology. It can capture, playback, stream, and transcode multiple media formats. The Java Communications API contains support for IEEE 1284-parallel ports (not in Windows XP - see Conclusion section for details). Figure 8 shows a diagram of the communication structure of the XCam2 system.

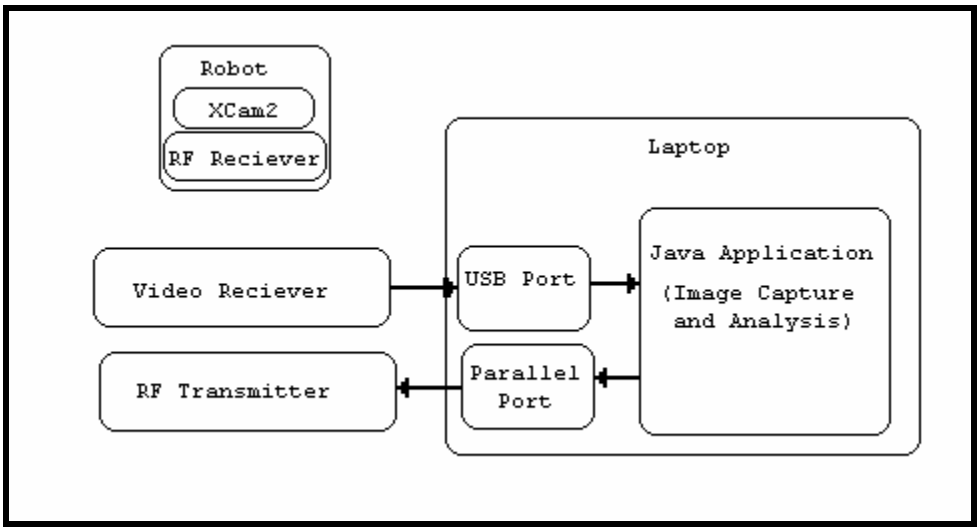

Figure 2: Sensor Integration Diagram

The following image is a sample of video data captured from the camera (in grayscale).

Figure 3: Video Data Sample

Problems

Since the wireless camera system uses a 2.4GHz frequency, the system can get interference from cordless phones and wireless Internet routers. The camera and receiver have channel select settings to allow you to pick a channel with the least amount of interference. Inadequate lighting ca also be a problem, causing the images to appear dark or gray. A good light source is

important. As you can see from the sample image, the images are dark and colors are not well defined.

Sensor Suit

The sensors on Cyclops are used for obstacle avoidance and to track a red object. Below is a summary of each sensor used and where it was purchased

IR

Two Sharp GP2D12 Detector Packages were used for as proximity sensors, mounted in the front with a crisscross view. The IR sensors were purchased from the Acroname website. I found that the cross sensor design could leave blind spots. This was overcome by writing a special IR calibration routine to store different threshold values for front and side obstacles. After using the calibration routine, the cross-view IR sensors are amazing and Cyclops hardly ever misses an obstacle. My IR setup and obstacle avoidance code is located in the AVR Code section of the Appendixes. Figure 4 is a picture of the Detector Package taken from the Acroname website.

Figure 4: Sharp GP2D12 Detector Package.

Bump

Three bump switches are mounted in the front and front-sides with a bump skirt. These are designed to stop Cyclops if the IR sensors fail to detect an obstacle. Originally, I had hard-coded threshold values for the IR sensors in my obstacle avoidance code – creating a need for a backup system. This code did not always work because front obstacles and side obstacles needed different threshold values. I added bump switches to help out the IR, but then later I added a calibration routine to give the IR better values. After calibrating the robot, the IR always works, and the bump switches are not really necessary, but the skirt looks really good. My code for the bump switches is located in the ARV Code section of the Appendixes.

RF

The RF transmitter/receiver combination provides communication between my laptop and Cyclops. An RF signal is only sent in one direction: from the laptop to the robot. A two-way connection is not necessary since the laptop receives video data from the wireless web camera. The RF receiver is setup on the robot, and the RF transmitter is setup and on the laptop.

The following RF components were purchased from the Rentron website, the remote Control Store section:

- Transmitter (TXLC-434 434MHz High Performance RF TX Module)
- Receiver (RXLC-434 434MHz High Performance RF RX Module)
- Two antennas (50-OHM Whip Style Antenna)
- 8-bit encoder (Holtek HT-640 8-Bit Encoder IC)
- 8-bit decoder (Holtek HT-648L 8-Bit Decoder IC)

The RF works extremely well; I had no problems with distance. An encoder/decoder setup provides error protection and I had no problems with invalid transmissions. Figures 5 and 6 show the circuits used to implement the Transmitter (TX) and Receiver (RX). The 8-bit encoder/decoder provides 2^8 different commands. I probably used only 15 commands. The error correction value on bits A9-0 ensures that a transmission is correct. The error correction value can be any value, but it must be the same value on both the encoder and decoder. The SN74LS244 on the TX-side is a chip containing 8 tri-state buffers to protect you from damaging pins on your parallel port.

The RF was very easy to put together. The TX and RX come with schematics showing exactly how to set it up. The hardest part was figuring out a way to write data to the parallel port. In order to access my parallel port I had to install the DriverLINX Port I/O, a driver to give port access under Windows XP, and download some .dll files. A link to this site is the Documentation section of this report. The code used to access the port is located in the Java Code section of the Appendixes.

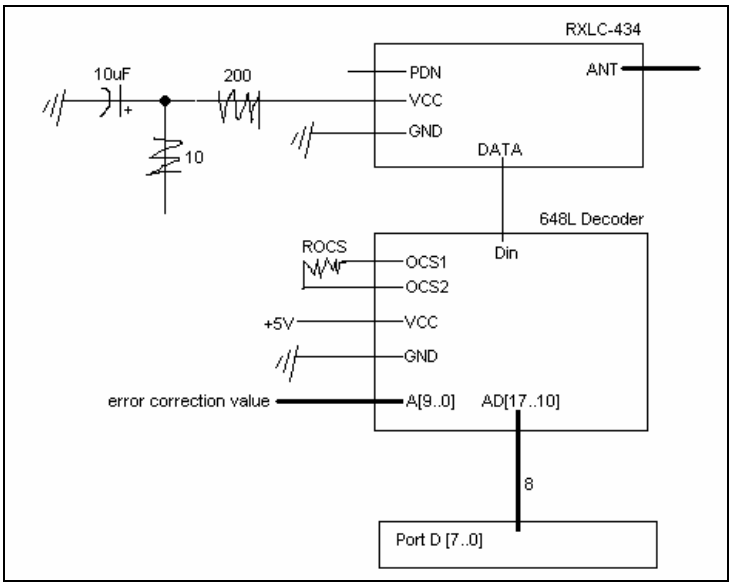

Figure 5: RX circuit.

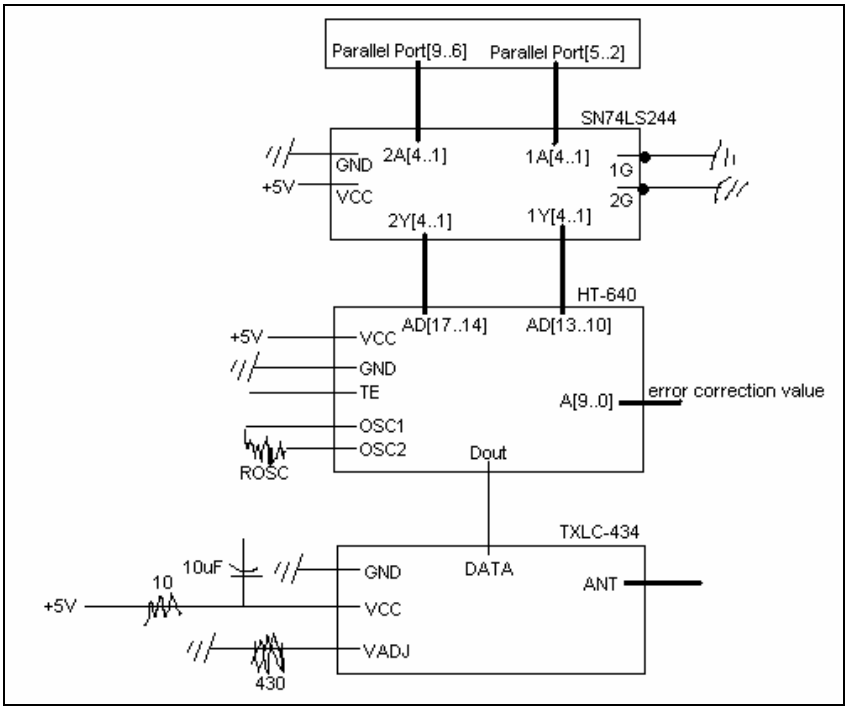

Figure 6: TX circuit

Sources for Parts

Acroname: http://www.acroname.com/index.html
- Parallax Cont Rotation Servo
- Red Wheel with Black Band
- Sharp GP2D120 IR Sensor

E-Plus: Gainesville
- Various stuff

Lowe's: Gainesville
- Small Caster
- Various Stuff

Radio Shack: Gainesville
- Various stuff

Rentron: http://www.rentron.com/PicBasic/RemoteControl.htm
- Transmitter (TXLC-434 434MHz High Performance RF TX Module)
- Receiver (RXLC-434 434MHz High Performance RF RX Module)
- Two antennas (50-OHM Whip Style Antenna)
- 8-bit encoder (Holtek HT-640 8-Bit Encoder IC)
- 8-bit decoder (Holtek HT-648L 8-Bit Decoder IC)

Spark Fun Electronics: http://www.sparkfun.com/shop/index.php?shop=1&cart=56589&cat=1&
- ATMega128 Mini Header Board
- Parallel Port Dongle Programmer for STK Port

X10: www.X10.com
- XCam2 package: XCam2, Video Receiver, USB adapter, and batter pack

References:

TA's: Louis Brandy, William Dubel, Max Koessik
Instructor: A. A Arroyo

David Fischer's Java Programming Examples: Capture Video from Logitech QuickCam Pro 3000 Camera with Java JMF.
http://www.mutong.com/fischer/java/usbcam/

Interfacing the Standard Parallel Port.
http://www.beyondlogic.org/spp/parallel.htm#2

Java Communications API by Sun Microsystems, Inc.
http://java.sun.com/products/javacomm/index.jsp

Java Media Framework API Guide.
http://java.sun.com/products/java-media/jmf/2.1.1/guide/index.html

JMF API and Sample Code by Sun Microsystems, Inc.
http://java.sun.com/products/java-media/jmf/index.jsp

Parallel Printer Port Access through Java.
http://www.geocities.com/Juanga69/parport/

Port IO - Simple user mode DLL and NT kernel driver to allow direct hardware I/O access. Works under Windows 95, 98, Me, 2000, and XP. This software is unsupported.
http://www.driverlinx.com/
(Port IO Driver)
http://www.geocities.com/dinceraydin/python/indexeng.html
(winioport module for Python: winioport.zip)

Programming in Java Advanced Imaging.
http://java.sun.com/products/java-media/jai/forDevelopers/jai1_0_1guide-unc/JAITOC.fm.html

Program Multimedia with JMF Tutorial by Budi Kurniawan.
http://www.javaworld.com/jw-04-2001/jw-0406-jmf1.html

RANZOR: IMDL robot by Cyrus Harrison.
http://www.mil.ufl.edu/imdl/papers/IMDL_Report_Summer_02/harrison_cyrus/ranzor.pdf

Rodriguez, Uriel: A Genius.