Jate Sujjavanich
EEL 5666C
IMDL Final Report
Thursday, August 7, 2003

Oscar the Grouchy Garbageman
Specical Sensor:
CMUcam

Introduction

The special sensor chosen for my robot was the CMUcam from www.seattlerobotics.com. This is a camera and microcontroller combination. Seattlerobotics.com provides it pre-assembled and tested for just $10 over the un-assembled price (total $109 less shipping). The camera comes with the ability to do simple vision recognition tasks with color. It is self-contained and has a simple user interface.

Power

The camera comes with a dropout regulator and takes an input between 6 and 9 volts. This allowed it to run off the raw battery pack of my robot. The camera also comes with a transformer which allows it to be tested more easily. Note that this

Interface

The interface for the camera is a serial one. There is a choice with communication via the level shifted port or the TTL/logic level pins. I chose the TTL/logic level pins for power conservation. It is possible and recommended to use the on-board level-shifted port for initial testing on a computer. I removed the RS-232 buffer to and used the TTL lines instead.

Software

Seattlerobotics.com provides a CD that contains useful interface/testing software. There is a Visual Basic and a Java version. The Java version requires that you have java installed. It also seemed more robust that the visual basic version. A screenshot appears below.

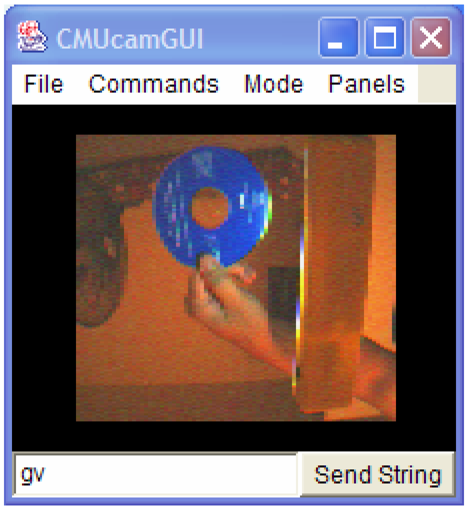

Figure 1 - Frame Dump

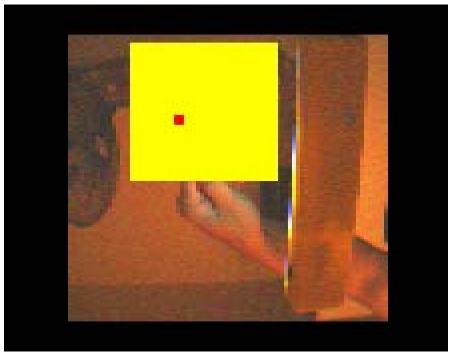
Figure 2 - Color tracking aid

Command Interface

The commands were all text based. They typically were a text prefix followed by a text suffix. For example:

```
"TC 100 240 16 50 16 50\r\0"
```

The TC stands for track color. The next three sets of numbers {{100, 240}, {16, 50}, {16, 50}} indicate the min and max values for red, green, and blue, respectively. Other commands follow a similar format as described in the manual.

After each command is sent, a reply "ACK" or "NCK" is received.

Performance

As you may be able to see in the above frame dump (depending on whether you are viewing a color version of this report), the camera has a tendency to become overly saturated with infrared light. The CD in the above photo was lit with a small fluorescent light, so it appeared to be its proper color. The robot performed well when during demo day under fluorescent lights. Indirect sunlight also gives good color balance.

When using the camera, the documentation suggests that you engage the auto white-balance and auto-gain functions so it is properly adjusted to the ambient light. This process takes about five seconds.

I was successful in using the Track Color command to center in on a color. The number used for this is the middle mass coordinate.

Appendix

The following code formed the interface for the camera:

```
const char uart1_data[][40] PROGMEM = {"\r\0",
                                       "CR 18 44 17 5 19 33\r\0", // Auto on
                                       "CR 18 40 17 5 19 32\r\0", // Auto off
                                       "PM 1\r\0", // Poll Mode On
                                       "TC 100 240 16 50 16 50\r\0", // Track Red
                                       "ACK M 53 103 49 82 58 111 5 34", // dummy data
                                       "TC 10 240 100 240 16 50\r\0" // Track Blue
                                       };
void cam_cmd(int num)
{
        strcpy_P(uart1_buf_out, uart1_data[num]);
        uart1_string_out();
        uart1_string_in();
}
void uart1_tx(char byte)
{
	while (!(UCSR1A & (1 << UDRE1))); // loop till UDR1 is empty
	UDR1 = byte;
}

char uart1_rx()
{
	char byte;
	while (!(UCSR1A & (1 << RXC1)));
	byte = UDR1;
	return byte;
}

void uart1_string_in()
{
	/* This function reads in a string from uart1 terminated by \r */
	/* String is input into a global string array */
	int i = 0;
	char byte = 0;
	while ((byte != ':') && (i < 40))
	{
		byte = uart1_rx();
		uart1_buf_in[i++] = byte;
	}

	/* Set last byte to be null termination */
	uart1_buf_in[i] = 0;
}

void uart1_string_out()
{
	int i = 0;
	while ((i < 40) && (uart1_buf_out[i] != 0))
	{
		uart1_tx(uart1_buf_out[i++]);
	}
}
```