Using Cameras for Robotic Tracking

Howard Li

Robotics and Automation Group
Department of Electrical and Computer Engineering
University of New Brunswick

http://www.ee.unb.ca/howard

May 21, 2009

Introduction

- A lot of highway accidents are caused by deviation from the road when driving a car.
- A lot of tasks need to be performed in environments which are unreachable by human beings.
- Autonomous road following has potential in preventing accidents caused by human fatigue and performing tasks in environments unreachable by human beings.

Current Technologies

- Continuously detection and tracking of road features.
- Steering and speed control based on road features.
- Computer vision for road feature extraction and analysis.
- Use of PID, neural-networks and reinforcement learning algorithms for control rules of autonomous road following.

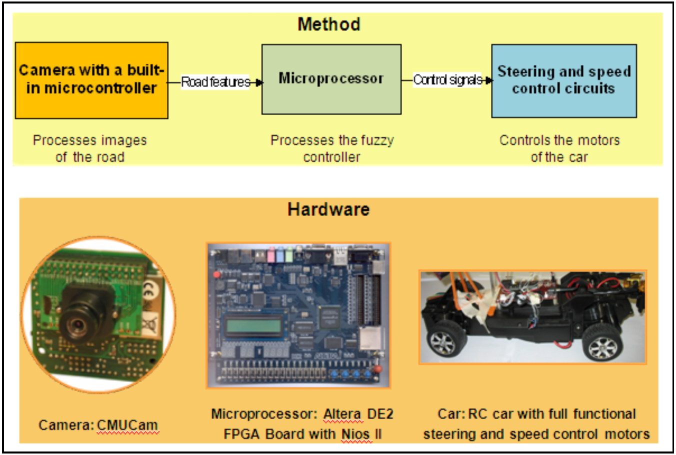
Objectives

- To develop an embedded system for vision based autonomous road following.
- To design a fuzzy logic controller (FLC) for autonomous road following.
- To analyze the stability of the proposed system.

Hardware Setup

- The objective of the project is to design a fully automated prototype system which can be used to control a motor vehicle using an on board camera.
- This is implemented in a way that the vehicle can remain between two colored lines while simulating highway motion.
- The idea is to utilize a simple system which processes the camera output and implements a control scheme which can quickly and accurately control the vehicle.
- The hardware system includes a camera associated with a frame-grabber.
- The camera is then connected to a microprocessor which controls the vehicle motion.

Architecture

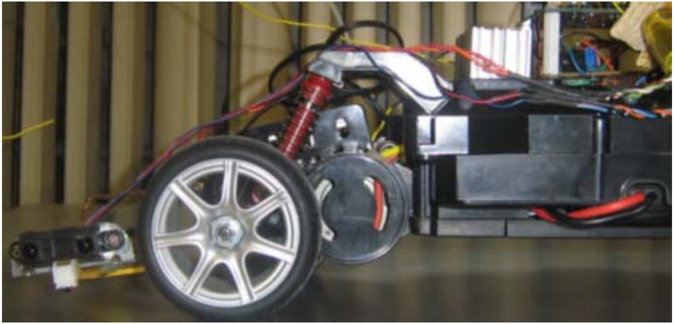


UNB

Remote Control Car

- The Audi S4 RC car platform is chosen because it offers a fully functional steering and drive system.
- The control signal for the steering and drive system is generated by the microcontroller which emulates the car's existing control signals.
- The speed and steering of the car are controlled by varying the PWM signals.
- The steering mechanism of the RC car is shown in the next slide.

Remote Control Car - continued



Camera

The CMUCam3 camera is chosen for the project. The camera uses a built in microcontroller and frame-grabber to allow for post processing on each captured image. Using a multiple color tracking algorithm, we can track each color individually. The centroid of two colored lines is then used to calculate the position error which is sent to the microprocessor. Since the vehicle is supposed to remain in the center of the two colored lines, error can be calculated if the centroid is not at the center of the image.

Camera, continued

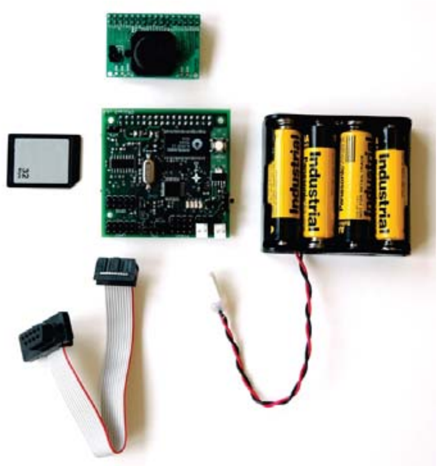


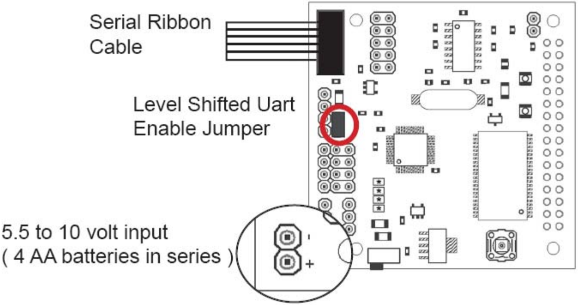
Figure: The CMUCam3 camera.

Camera, continued

The CMUcam3 can be loaded with various firmware images. These images may perform different specific tasks. For example, the "cmucam2.hex" file will allow your CMUcam3 to emulate the functionality of the CMUcam2. When you develop your own CMUcam3 applications or would like to experiment with new firmware images, it will be necessary to flash the microprocessor. Flashing the microprocessor can be done over a standard serial port using the downloading utility.

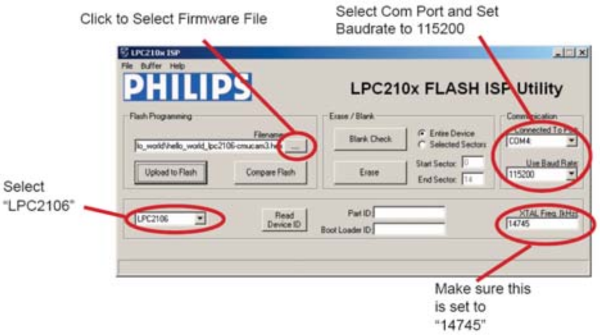
How to Program the CMUCam3 Camera

Install the flash utility, connect the CMUcam3 to your serial port as shown below



How to Program the CMUCam3 Camera, continued

Go to the Start Menu, “All Programs”, “LPC210x ISP” and execute the “LPC210x ISP” application. You should eventually see a screen similiar to this:

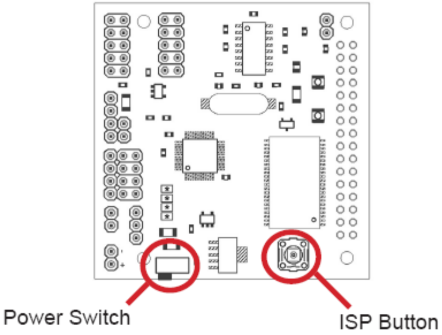


How to Program the CMUCam3 Camera, continued

Click on the "..." box next to "Filename" to select a hex file to download. As an example, select the "hello_world_lpc2106-cmucam3.hex" file located in the CMUcam3 hello_world project directory. Set the correct Comm Port and Device Selections. Click on the "Upload to Flash" button. You should see a dialog box like below.

How to Program the CMUCam3 Camera, continued

While holding down the “ISP” button on the camera, turn on the camera power. Make sure the button is held in for about a second. If the camera is already on, turn it off and on again while holding in the “ISP” button. If the button is depressed upon startup, the camera will enter its bootloader mode.



How to Program the CMUCam3 Camera, continued

Press "OK" and wait for the image to download. If it fails check the connectors, reset the power and try again. The Flash Utility will say "File Upload Successfully Completed!" when the download is complete.
The code we have developed to detect the centroid of the road for calculating the error as the input to the controller is listed in the table shown in the next slide.

Computer Vision

Table: Error Calculation Chart

Red Centroid	Yellow Centroid	Error Calculation
$RedCent! = 0$	$YellowCent! = 0$	$Error = 3\times (Average - Center + 20)$
$RedCent! = 0$	$YellowCent = 0$	$Error = 3\times (175 - RedCentroid)$
$RedCent = 0$	$YellowCent! = 0$	$Error = 2\times (6 - YellowCentroid)$
$RedCent = 0$	$YellowCent = 0$	$Error = Error$

Microprocessor

- The PIC18F4431 microcontroller, programmed using the C language, is used to interface with the camera and car's existing circuitry.
- As comparison studies, the Altera DE2 FPGA board is programmed to interface with the camera and the car's circuitry.
- The microprocessors receive the incoming position error from the RS232 port located on the CMUCam3 board. The microprocessors then use the error to calculate the appropriate steering signal, using a fuzzy controller.
- The steering signal is then output to the RC car's circuit.

Kinematic Model of the Vehicle

The robot's configuration space has four dimensions, two for translation, one for rotation and one for the steering angle. Let $(x,\ y,\ \phi,\ \theta)$ denote the configuration of the robot, parameterized by the location of the front wheels. The kinematic model of the robotic vehicle can be represented as

$$\begin{aligned} \dot{x} &= u_3 \cos\theta, \\ \dot{y} &= u_3 \sin\theta, \\ \dot{\theta} &= \frac{u_3}{l} \tan\phi, \end{aligned} \tag{1}$$

where u_3 corresponds to the forward velocity of the vehicle and the angle of the vehicle body with respect to the horizontal line is θ, the steering angle with respect to the vehicle body is ϕ, $(x,\ y)$ is the location of the center point of the front wheels, l is the length between the front and the rear wheels.

Kinematics, continued

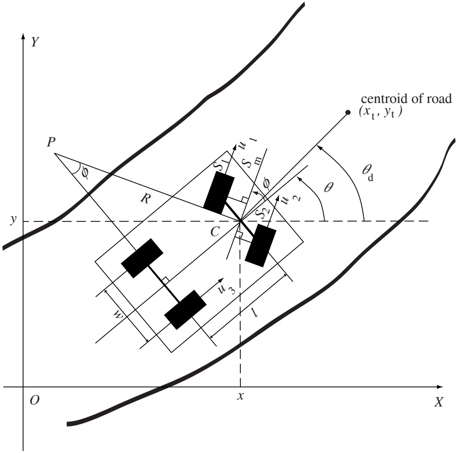


Figure: The model of the 4-wheeled robotic vehicle.

Kinematics, continued

Since the turn radius of the robot is quite large compared with the radius of the wheels, we have the following relations:

$$\begin{aligned} S_1 &= (R - \frac{w}{2}\cos\phi)\phi, \\ S_m &= R\phi, \\ S_2 &= (R + \frac{w}{2}\cos\phi)\phi, \end{aligned} \tag{2}$$

where S_1 and S_2 give the displacement (distance traveled) of the front left and front right wheel respectively, R is the turn radius of the center point of the front wheels, w is the distance between wheels (from center-to-center along the length between the two font wheels or two back wheels), and ϕ is the angle of the turn in radians. S_m is the displacement at the center point of the front wheels.

Kinematics, continued

Once we have established the simple geometry for the robotic vehicle system, it is easy to develop algorithms for controlling the robot's steering angle ϕ, thus controlling the robot's orientation θ. By adjusting the steering angle ϕ, we can control the velocity of both the front wheels

$$\begin{aligned} \dot{S}_1 &= \dot{\phi}(R - \frac{w}{2}\cos\phi) + \frac{w}{2}\phi\dot{\phi}\sin\phi, \\ \dot{S}_2 &= \dot{\phi}(R + \frac{w}{2}\cos\phi) - \frac{w}{2}\phi\dot{\phi}\sin\phi, \end{aligned} \quad (3)$$

so that

$$u_2 - u_1 = \dot{\phi}w(\cos\phi - \phi\sin\phi), \quad (4)$$

where u_1 and u_2 correspond to the forward velocity of the front left wheel and the front right wheel respectively.

Fuzzy Logic Controller

The error is represented by $\theta_d - \theta$. To reduce the error to zero, the steering angle should be equal to ϕ. ϕ is determined by the fuzzy controller. $\theta_d - \theta$ is calculated in the camera, and then transmitted from the camera through the serial port to the microprocessor. The error and the change in error are calculated and fed into the fuzzy controller as shown in the figure.

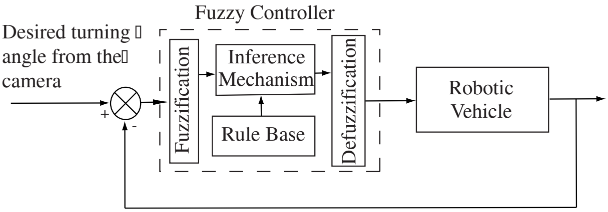


Figure: Block diagram of the fuzzy logic controller.

Fuzzification

In this study, we use five membership functions for both error $e = \theta_d - \theta$ and change in error $\dot{e} = \dot{\theta}_d - \dot{\theta}$.

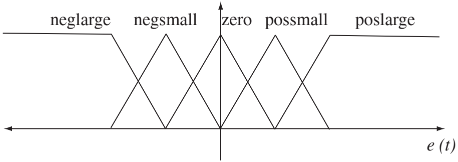


Figure: Input membership functions for error e.

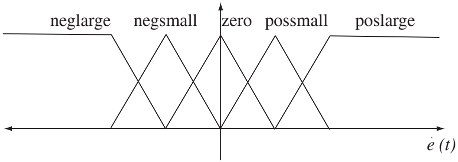


Figure: Input membership functions for change in error $\dot{e}$.

Fuzzy Rule Base

The rule base stores the rules governing the input/output relationship of the proposed fuzzy controller. The inference mechanism is responsible for decision making in the control system using approximate reasoning

Table: Rules For the Steering System

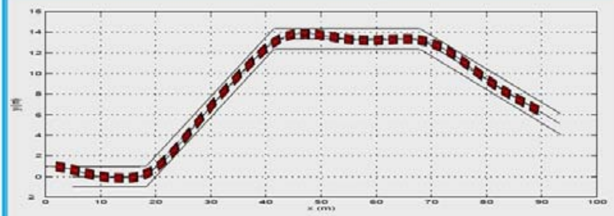
	Φ	$\dot{e}$ pos large	pos small	zero	neg small	neg large
	poslarge	PL	PL	PL	PS	ZO
	possmall	PL	PL	PS	ZO	NS
e	zero	PL	PS	ZO	NS	NL
	negsmall	PS	ZO	NS	NL	NL
	neglarge	ZO	NS	NL	NL	NL

Defuzzification

The defuzzification procedure maps the fuzzy output from the inference mechanism to a crisp signal. We use the "center of gravity" (COG) defuzzification method to combine the recommendations represented by the implied fuzzy sets from all the rules. Let b_i denote the center of the membership function of the consequent rule (i) and $\int \mu_{(i)} dx$ denote the area under the membership function $\mu_{(i)}$. The COG method computes the crisp value μ^c to be

$$\mu^c = \frac{\sum_i b_i \int \mu_{(i)} dx}{\sum_i \int \mu_{(i)} dx}. \quad (5)$$

Simulation



Following A Road in Cartesian Space

Experiments - 1

Experiments - 2

UNB

Conclusion

- It is discussed how to use a CMUCam 3 camera and an RC car for robotic tracking.
- We have discussed the hardware setup with the camera, the microprocessor, the RC car, and the fuzzy logic controller for the project.
- Simulation and experimental results have been presented.