CS684 Project Report

This report is intended to help us help you so please fill in this document carefully. This, like the other work products, is part of the project assessment. This report is to be submitted through Moodle BEFORE your prototype is assessed in the lab. Please fill ALL the sections given below –

Project Title: ***Making the Firebird V traverse along arbitrary curves***
Project Team number: ***21***
Team members (Name and roll numbers): ***Rakesh D 10305069, Revya Naik 09305078***

1. Introduction

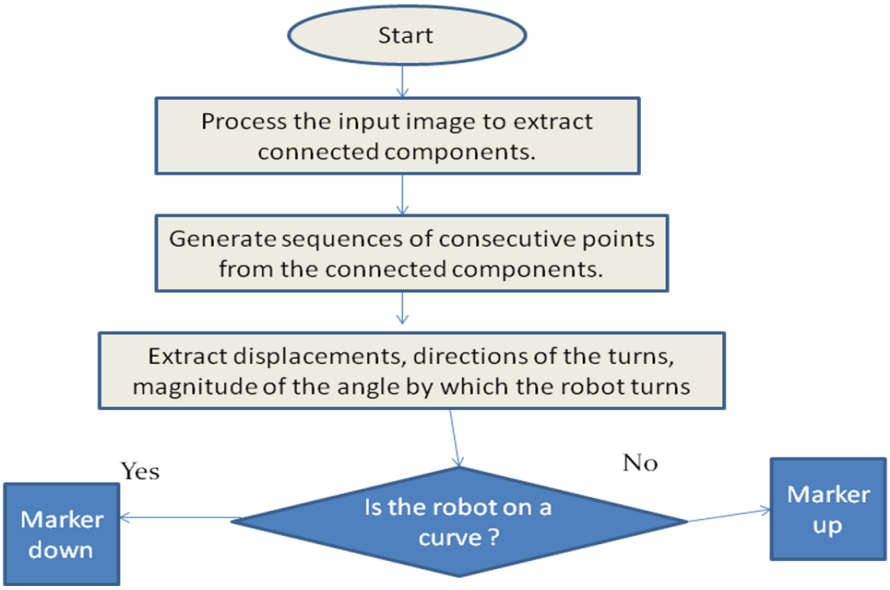
1.1 Problem statement

The aim of the project is to make the Firebird V robot traverse along arbitrarily shaped curved paths. A binary image of the curve(s) is taken as input and the Robot is expected to trace the path depicted in the image as faithfully as possible.
The degree of accuracy in tracing the path can be verified by attaching a Marker to the Robot so that it draws the curve as it traces the path.
The image may contain multiple instances of curves, in which case, the Robot is supposed to draw all the curves by lifting the Marker up and down at appropriate points.

1.2 Requirements Specification

1. Firebird V Robot
2. Servo Motor attached to the robot to hold the Marker in position
3. MATLAB to process the input images

1.3 System Design (use formalisms such as Statecharts, FSM, pseudo-code, etc.)



1.3 Assumptions and limitations

We assume that the input image doesn't contain curves crossing each other. If it has such curves, the robot arbitrarily selects one branch and traverses along that branch.

1.4 Setup and any extensions implemented on the robot

A servo motor rotating in the vertical plane is fixed on top of the robot, so that the Marker attached to it can be lifted up and down to draw multiple curves.

1.5 Additional hardware used

Servo Motor, Marker.

1.6 Issues specific to the project

Precise position control based on PWM control of the velocity of the wheels of the robot is not possible as the PWM resolution is very bad. Thus the Robot can only be made to move along a curve only by dividing the path into a piecewise linear curve. The degree of accuracy of the linear approximation is dependent on the length of the individual linear segments. Smaller the segments, better the approximation to the curve. However, the lengths of the individual segments of the curve can't be too small as the minimum distance the Robot can accurately measure is 5.38 mm, the difference in distance travelled by the robot when the input given is 10 and 12 can't be too large, thus affecting the accuracy of reproduction of the curve.

2. Present Status

2.1 A timeline based picture of project stating current status + requirements completed

The project at present traverses all non-intersecting curves fairly accurately. But it doesn't handle intersecting curves accurately. The accuracy of relative distances in images with multiple curves is not high. However, the robot is capable of drawing any image containing non-crossing curves.

2.2 If there are any delays, why they occurred? How you have overcome issues?

Initially we planned to control the position of the robot by precisely controlling the velocities (obtained by solving robot kinematics equations) of the left and right wheels so that the robot travels along the required curved trajectory. However, the available PWM resolution (8-bit) proved to be very restrictive for the precise control of velocities. Hence we shifted to an approach of approximating the given curve by a piecewise linear curve. This change in approach caused some delays.

2.3 Critical steps in your projects: hardware, interfacing, algorithmic complexity, etc.

1. Processing the input image:
   The input image which has multiple curves should first processed to separate the individual curves, which can be done using connected components algorithms. Each of the components is then thinned to get a one-pixel thick curve using morphological processing. The list of pixels is then transformed to a list of adjacent co-ordinates.

2. Generating useful data from the raw data of the image:
   The list of adjacent point co-ordinates is then down-sampled to generate a manageable number of points. These points act as the ends of linear segments into which we intend to divide the original curve. These points are now used to calculate the displacement along each individual segment, the angular turn at the end of each segment – both direction (Left Turn or Right Turn) and magnitude of the angle. An array of displacements, direction of angular displacement, and magnitude of angle is generated corresponding to each segment. This data is now used as the input for the C-file which implements the motion of the Robot.

3. Implementation of the actual steps in C
   The motion of the Robot is controlled according to the data generated in the previous step by precisely moving the Robot along pre-calculated displacements and angles. The servo motor makes sure that the Robot doesn't draw while it moves from one curve to another curve.

2.4 Individual roles and contributions

Rakesh – Design and implementation of the MATLAB functions, scripts to extract the data (displacements, angle of a turn, direction of a turn) from the given image. Extension of the algorithm to the case where there are multiple curves.

Revya – Implementation of the remaining part of code in C, (managing servo motor, actual commands to the hardware of the Robot)

2.5 How much time devoted to project so far - man-days

Approximately 70 ~ 80 man hours in the lab and additional 70 man hours working on MATLAB code processing the image, generating the data.

5. Final Roadmap of Project

5.1 The roadmap to completion – with milestones + deliverables

First week – Processing the input image, generating the list of segment ends

Second week – Processing the segment end points to generate linear, angular displacements, directions.

Third Week – Implementation of the C code which uses the generated data to finally realize the objective. Testing and modifications were carried out.

First Prototype was presented on 29th October, where the Robot was capable of handling one curve per image. Modifications (handling multiple curves) were suggested which were implemented in the sub-sequent week.

Final Version, presented on 9th November is capable of handling multiple curves per image.

5.2 State deadlines for each activity – Project completion, Documentation, code documentation etc.

First prototype completed by 27th October. Final Version and Documentation by 8th November.

6. Innovation, Creativity and Reusability Index of your Project

6.1 Innovations in project

The Robot is capable of handling arbitrarily complex curve instead of straight line images. Thus the Project can be used to enable the robot maneuver around arbitrarily shaped obstacles by drawing the trajectory it follows.

6.2 Reusability in project

The code can be broadly classified into two Parts – MATLAB (for generating the required data for the robot), C (for actually controlling the hardware based on the instructions from the data generated by MATLAB). There are 8 modularly designed MATLAB functions, each function performing a well-defined subtasks, interfaces as explained in the documentation. Thus the code is very much reusable.

7. Help us in improving the process

7.1 What you think can be improved in terms of project activities

Selection of problem statements could have been done earlier to facilitate easier planning

7.2 Any comments on the current schedule of events

All the deadlines have been scheduled close to the end of the semester making it difficult to manage.

7.3 Are you satisfied with the way the course activities have gone – specially the project ?

Yes

8. Bug Report

1. The relative position of curves in case of multiples curves isn't highly accurate due to the scaling used in the code. (Exact reproduction without scaling, would require a large space)
   Possible Fix: The accuracy can only be improved if the resolution of the position control or angle control is improved. (which stands at 5.38 mm)

2. The robot doesn't initially orient itself in position before drawing the curves. Hence the orientation of the resultant picture of the Robot is dependent upon the direction along which the first curve the Robot draws is oriented.
   Possible Fix: To try and align itself in required position by using a camera

9. Future Scope

1. The project can be extended to handle curves which cross each other.
2. It can be extended to remotely control the position of robot by communicating the images of the curves along which we intend to move the robot.
3. It can be extended to an online version, in which the Robot travels along a curve as we draw it on an input device like touch pad. It may be computationally intensive.

10. Learnings

1. Precise position control of the Robot is not possible by manipulating the velocities, as the PWM resolution isn't good.
2. Handling of curves – conversion into piecewise linear segments, extraction of angles
3. Image processing – Extraction of connected components, morphological processing etc.

CS684 Documentation – README file

CS684 – 2010 Project

Making the Firebird V Robot Traverse along arbitrary curved paths

Team 21 - details**Students:**

Name	Roll No.	Email
Rakesh Dhanireddy	10305069	rakesh@cse.iitb.ac.in

Revya Naik	09305078	revya@cse.iitb.ac.in

Project Objective

Introduction:

The aim of the project is to make the Firebird V robot traverse along arbitrarily shaped curved paths. A binary image of the curve(s) is taken as input and the Robot is expected to trace the path depicted in the image as faithfully as possible.

The degree of accuracy in tracing the path can be verified by attaching a Marker to the Robot so that it draws the curve as it traces the path.

The image may contain multiple instances of curves, in which case, the Robot is supposed to draw all the curves by lifting the Marker up and down at appropriate points.

Hardware Platform

The hardware required:

1. Firebird V ATMEGA2560
2. A Servo Motor to enable the lift-up motion of the Marker

Software

Software Used:

1. AVR Studio 4
2. Matlab 7.14

Code Description

Code Files.

File	Purpose	Executes on
TraverseCurve.c	Main Program	Robot
Winfb.h	Contains the abstractions of major operations.	Robot
image.h	Contains the data of the image	Robot
newpaths.m	Calculates the required data given a binary image as input	PC
extractpaths.m	Extracts the points out of the given binary image in required order	PC
calangle.m	calculates the angles by which the robot has to turn	PC
caldist.m	calculates the displacements by which the robot has to move at various points along the curve	PC
newnext.m	Called by extractpaths.m to extract successive points from the image.	PC

Folder Structure:

Deliverables

Filenamed	Contains
TraverseCurve-C-code.zip	SourceCode of programs to be burnt on Robot. Contains documentation of the code as well.
MATLAB-interface.zip	Contains Matlab files.
Documents.tar.gz	Contains Project related doc files.

Instructions

Instructions to run the project:

1. Create a 1 bit Monochrome bitmap image with Paint with the desired shapes of arbitrary curves, say, “curve.bmp”, save it in Matlab home directory.
2. Read the image into Matlab by the command “im=imread(‘curve’, ‘bmp’);”. Now the binary image is stored in a matrix ‘im’.
3. Edit the file path of ‘image.h’ (to which the image data is written by the Matlab file) in the newpaths.m according to the directory structure.
4. Run the command ‘newpaths(im);’ in the Matlab console. This file calculates all the required data points, angles etc. (with the help of other matlab source files) and populates the ‘image.h’ file with the required data.
5. Create a new project in AVR Studio named TraverseCurve and add the files ‘TraverseCurve.c’ and ‘image.h’ as source files and compile the code.
6. Burn the hexfile to the Firebird V and place it in a spacious location. The Robot traverses along the curve depicted in the image as required!