利用樹莓派建立模型基準化設計之車道偵測系統

張志翰	林嘉洨	田昆育	林楷能
華創車電技術中心	台北大學	台北大學	台北大學
股份有限公司科長	電機工程學系教授	電機工程學系	電機工程學系
chih-han.chang@haitec.com.tw	jclin2014@gmail.com	keant010203@gmail.com	tobbyyo@gmail.com

摘要

本論文是以模型基準化之設計完成能夠偵測車輛是否行進在車道中間的車道偏移警示系統，以利駕駛人在行進間集中精神。本系統選用樹莓派的鏡頭去截取即時影像，為了減少運算處理的時間，僅截取畫面下三分之二處，在經過 2D 濾波與最大類間方差法去消除雜訊與轉成二值化影像，緊接著利用霍夫轉換與卡爾曼濾波器分別做車道線的選取與追蹤。當車輛偏離車道中間時，為了做到警示駕駛人的效果，在追蹤出的兩條車道線間決定左右之車道線，並算出兩條車道線與車道中心的水平距離，藉此判斷為左偏或右偏，並在畫面中發出警告給駕駛者。

關鍵詞：樹莓派、霍夫轉換、模型基準化設計

Abstract

This paper attempts to address a model-base lane departure warning system which can detect whether the automobile deviates from the left lane or right lane line in order that it is able to provide the driver with a alert.

This system utilizes Pi Camera from Raspberry Pi to capture the immediate image. First of all, the image is separated into bottom and top parts instead of using whole image and only the bottom part is used in order to reduce time for calculating and boost the precision. Second, the 2-D FIR Filter will eliminate unnecessary clutters. Next, the Otsu's Method makes the whole system proceed under different luminosity by automatically converting an intensity image to a binary image. Besides, Hough Transform and Kalman Filter are two crucial parts needed to detect where the lanes are and the deviation, respectively. At last, the system will warn the driver of the visual signal if the lane departure signal is activated. As a result, the model-based method makes it possible to validate numerical behaviors of Raspberry Pi in Simulink. A prototype has been implemented and the experimental results are commented.

Keywords: Raspberry Pi, Hough Transform, Model-Based Design.

1. 前言

1.1 研究動機與目的

在生活型態逐漸增快的這個世代，車輛已成為了我們生活當中不可或缺的一個部分，國人擁有的車輛數目也於 103 年 9 月突破 750 萬輛[1]，車輛事故的發生次數也伴隨著持續增加，因此對於車輛安全系統的研究更加受到重視。除了平時眾所皆知的被動安全系統，例如：安全帶、安全氣囊，對於主動安全系統的研發更是重要，如：電子穩定系統(EBS)、防鎖剎車系統(ABS)，而在本論文中，欲使用由樹莓派作為平台，結合 simulink 之中系統工程的概念去完成車道偏移系統(LDWS)，提供駕駛人與用路人一個安全的環境。

1.2 文獻回顧

在偵測車道線中，許多的前輩都對這個部分有許多深刻的討論，例如說在 2005 年王昭祥先生的「即時車道偵測與追蹤」[2]，就是利用霍夫轉換加上邊緣偵測的技巧，成功將車道線偵測出來，在 2009 年范耿豪先生的「以霍夫轉換為基礎之智慧型快速車道線偵測」[3]便是利用將 ROI 選取在車輛前方的方法，加上車道線的亮度會比周圍高的特點達到準確的偵測出車道的效果，並且因為 ROI 的選取，將判斷的像素點的大量減少，而在結合嵌入式系統去達成車道偵測的效果，有郭宗彥先生在 2008 年的「以嵌入式平台實現車道偵測系統」的研究[4]，用 PXA255 為平台，使用 Linux 架構下，在藉由切割圖片的方法去加速系統完成偵測的功能，而於 2010 年時，洪國振先生則是完成了「基於 WinCE 平台實現嵌入式車道偏移警示系統」[5]，不同於前者，此論文使用 Linux 架構，主要的作業系統為 Windows CE，並且利用 PXA270 作為平台，實踐了車道偏移系統，在 2012 年，魏仟豪先生的「在嵌入式異質雙核心平台上整合開發夜間車燈與車道追蹤偵測與事件記錄功能之駕駛輔助系統」[6]在軟體方面是用 Ubuntu 去完成，而不同於上述的論文，在硬體方面是利用雙核心系統，分別為 ARM 和 DSP 兩種系統，ARM 的核心負責影像控制、程式控制、車道線截取的工作，而 DSP 的核心則掌管影像轉換、車燈

截取、行車紀錄影像的動作，而在本論文中，希望能夠結合前人對於車道線偵測的研究加上追蹤，最後移植到目前(2015)體積最小的單板機的 Raspberry Pi 去完成車道偏移系統。

2. 主要系統

2.1 系統架構

首先本文藉由 Pi camera 即時性接收道路資訊後，一開始進來的是 RGB 的圖像，所以先轉成光度影像後，再切割出有興趣的部分(ROI)，因為過於下面會是引擎蓋的部分，而遠處會照到非道路的部分(天空或樹)，所以主要的範圍是圖像的下三分之二，接著利用最大類間方差法決定閥值的大小後，轉換成二值圖，之所以利用最大類間方差法是希望增加系統的穩定度，減少外界光暗的影響，到這裡算是做完影像的前處理。

接著就是用霍夫轉換找出整個圖像上的直線，轉換出來的結果中，對於θ有一個限定是要在 11 度到 90 度之間與-11 到-90 度之間，原本霍夫轉換出來的結果會在-90 度到 90 度之間，去除的部分是幾乎為水平線段的可能，之所以這樣做是為了避免車輛停止線這類橫向的線導致本系統的誤判，考慮到偵測的過程中仍然會有誤判的可能性，所以加入了卡爾曼濾波器去做修正，在迭代的過程中找出兩條最適合的車道線。

在得到兩條車道線以後，將其轉換回原本的卡氏座標，接著判斷何為左線、何為右線以及設定車道中心與左右線該保持的距離範圍，以作偏移警示之依據，最後再加上原本的圖像資訊，將紅線及偏移警示標示於畫面並輸出。

2.2 色彩模型

本論文中主要用到的色彩空間有三種，分別為灰階型態、RGB，灰階型態主要是讓圖片僅存在有 1 與 0 兩種訊號，分別代表黑與白兩色，主要是做為霍夫轉換的前處理，RGB 的圖像主要是為了儲存原始圖像，以利在完成處理後進行影像的疊合。

2.3 霍夫轉換

霍夫轉換是用來偵測圖形中幾何圖形的方法，因此利用這方法去搜索在圖形中的直線作為車道線，在卡氏座標中，任何的線都可以表示成

$$y=mx+b \quad (1)$$

式中 m 為斜率、b 為 y 截距。

在鉛垂線的情況，m 會趨近於無限大而無法表示，所以如下圖 1 所示，可知

$$b=\rho\cos\theta \quad (2)$$

$$m=\cot\theta \quad (3)$$

藉由這樣我們可以轉換從 x,y 關係式改成ρ和θ的關係式，而通過任一點(x0,y0)隨著斜率與 y 截距的不同會對應到許多組(ρ,θ)。

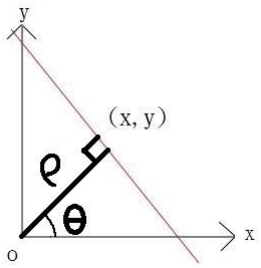

圖 1.卡氏座標與極座標轉換對照圖

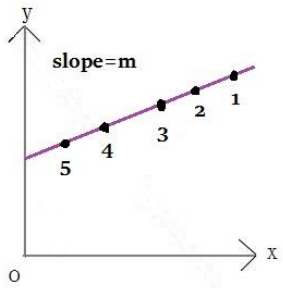

圖 2.卡氏座標圖

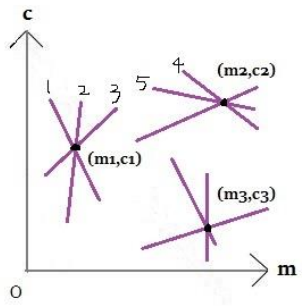

圖 3. 卡氏座標轉換至 m-c 卡式座標

在極座標的情況下，霍夫轉換找尋交集最多的點作為想要找的線段，因為直線是由無限多的點所構成的，線上的點都可以找到共同的(ρ,θ)，下圖便是將線轉化成極座標的情況。

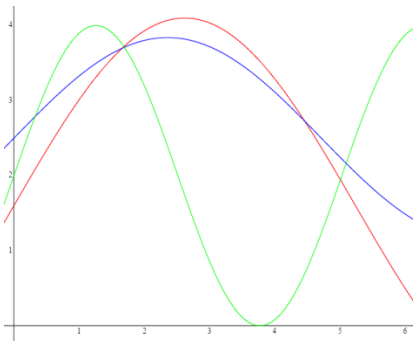

圖 4.卡氏座標轉極座標

2.4 卡爾曼濾波器

卡爾曼濾波器是一種利用高效的遞迴方法來估計輸入訊號的過去與當前狀態，甚至是預測未來的狀態，整個卡爾曼濾波器的概念主要建立在五大公式之上，並且利用系統預測值與測量值來估算當下的狀態，對於系統的預測值會採用式(4)、(5)來計算，而測量值則是透過式(6)、(7)、(8)計算所得，其中 Kg(k)為卡爾曼增益。

$$X(k|k-1)=AX(k-1|k-1)+B\ U(k) \quad (4)$$

$$P(k|k-1)=AP(k-1|k-1)A^T+Q \quad (5)$$

$$X(k|k)=\ X(k|k-1)+Kg(k)\ (Z(k)-H\ X(k|k-1)) \quad (6)$$

$$Kg(k)=\ P(k|k-1)H^T/\ (H\ P(k|k-1)\ \ H^T+\ R) \quad (7)$$

$$P(k|k)=(\ I-Kg(k)\ H\ )P(k|k-1) \quad (8)$$

3. Raspberry Pi

3.1 Raspberry Pi

在本論文中採用的板子為 Raspberry Pi(樹莓派)model B，下表 1 為其規格:

表 1 Raspberry Pi 硬體規格

名稱	Raspberry Pi Model B
SoC	Broadcom BCM2835
CPU	700MHz , ARM1176JZFS Core
GPU	Broadcom Video Core IV ; OpenGL ES 2.0 ;1080p 30 h.264/MPEG-4 AVC
RAM	512MB
網路介面	10/100 乙太網路介面
USB 2.0 介面個數	2
儲存裝置	SD / MMC / SDIO 卡插槽
音訊輸出	3.5mm jack ; HDMI
視訊輸出	RCA Terminal ; HDMI
額定功率	700mA
電源輸入	5V
總體尺寸	65x56.5x10 毫米
重量	23g

3.2 Pi camera

鏡頭的部分是使用專屬於 Raspberry Pi 的 Pi camera，利用軟排線與板子作連接，大小為 25mm x 20mm x 9mm、擁有五百萬像素(2592×1944 pixels)，支援 1080p30 與 720p60 錄影功能。

4. 模型化的基礎設計

4.1 系統流程

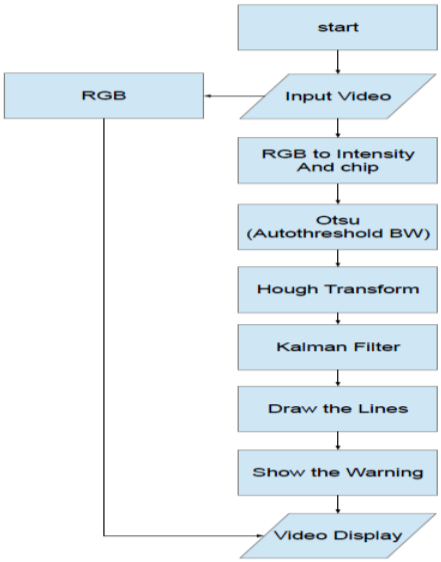

圖 5.系統總流程圖

4.2 Model-Based Design

本論文的系統架構是以 Matlab / Simulink 建立而成，這套軟體的特色在於能將任何系統模組化，利用積木的概念，將系統逐步建立起來，因此，本系統簡單分成 5 個部分，分別為輸入、車道偵測、車道追蹤、偏移警示、輸出:

輸入的部分，利用樹莓派的鏡頭將畫面讀取進來，再將 RGB 轉成光度影像，接著擷取畫面下三分之二後丟進下一個子系統做處理與判斷。

車道偵測的部分，先利用 2-D FIR 濾波器將前一個子系統的輸出資料做去除雜訊的動作，隨後再將畫面二值化，送入霍夫轉換，選出兩條可能直線的θ 跟 ρ值。

車道追蹤的部分，利用迭代的計算方式與卡爾曼濾波器修正θ 跟 ρ值。

偏移警示的部分，將前一個子系統輸出的兩組θ 跟 ρ值用霍夫線的模塊轉換到卡氏座標上，每一組θ 跟 ρ值產生兩組(x,y)座標，因此形成一條直線，接著利用兩條線靠近畫面下方的點座標去判定左線或右線，設定車道中心與左右線該保持的距離範圍，一旦某側小於之，便必須顯示偏移警示。

輸出的部分，透過畫直線的模塊將標線畫在畫面中，另外，若有偏移警示的訊號輸入此部分，則在畫面左上顯示左偏或右上顯示右偏。

5. 實驗結果

5.1 模擬實驗介紹

我們在實驗室中建構一個等比例縮小的車道環境，去模擬實際的情況；因此我們量測 Raspberry Pi 在車輛上的情況，底下的表 2 為小客車與模擬場地的對應情況。

表 2 實驗模擬與實地測量長度對照圖 (10:1)

	實地測量	實驗模擬
鏡頭離地垂直高度(公分)	110	11
鏡頭到車道線水平距離(公分)	370	37
鏡頭畫面最大寬度(公分)	300	30
車道線寬(公分)	10	1
車道寬(公分)	225	22.5

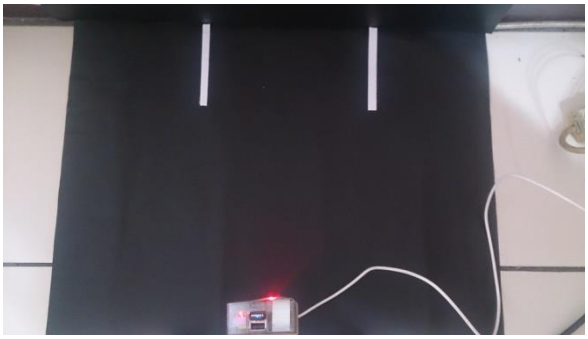

圖 6. 實驗場地鳥瞰圖

圖 7. 實驗場地側視圖

圖 8. 鏡頭架設在車子上的畫面

5.2 模擬結果與分析

在前面的系統架構有提到，經過系統所處理的畫面是截取原畫面下三分之二的部分，並且經過二值化再送入霍夫轉換。偵測可能直線的θ 跟 ρ 值，下面三張圖分別是原畫面、擷取畫面以及二值化的畫面:

圖 9. 原畫面白色實線由鏡頭輸入的畫面

圖 10. 原輸入畫面擷取下三分之二後的畫面

圖 11. 二值化的畫面

接著考量道路上的車道線有實線、虛線、不同顏色的車道線，所以我們將實驗分別對車道的顏色、類型、光度情況、車輛是否有左右偏移的情況分別當作變因加入討論，以下是各實驗結果圖。

5.2.1 實驗一: 車輛偏移與否的車道線偵測

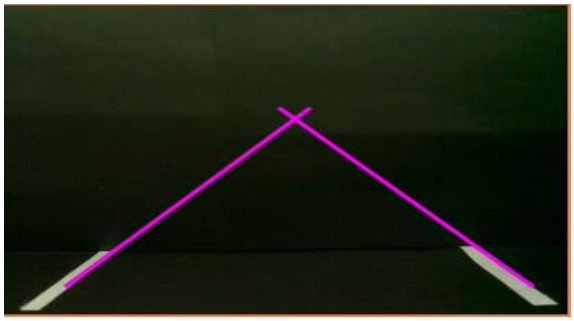

圖 12. 車輛不偏移的畫面

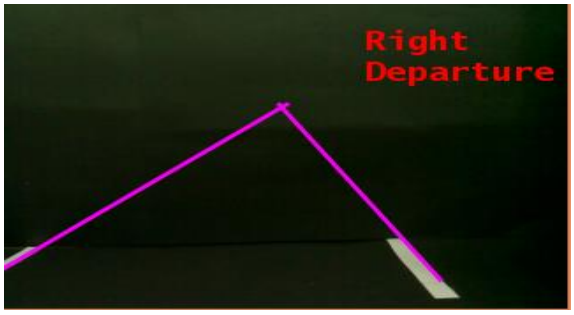

圖 13. 車輛右偏的畫面

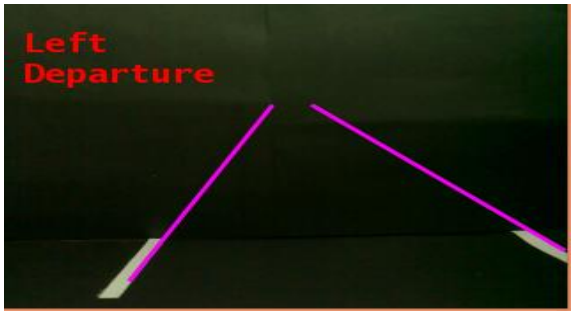

圖 14. 車輛左偏的畫面

5.2.2 實驗二:不同顏色之車道實線偵測

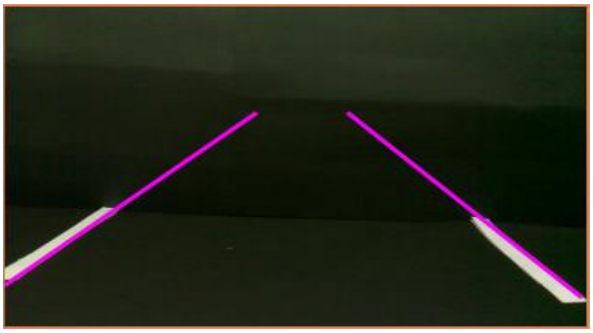

圖 15.偵測白色實線的畫面

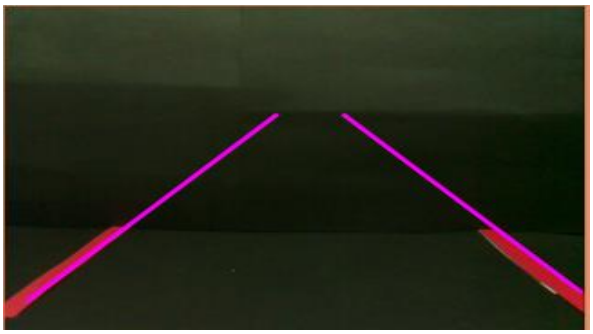

圖 16.偵測紅色實線的畫面

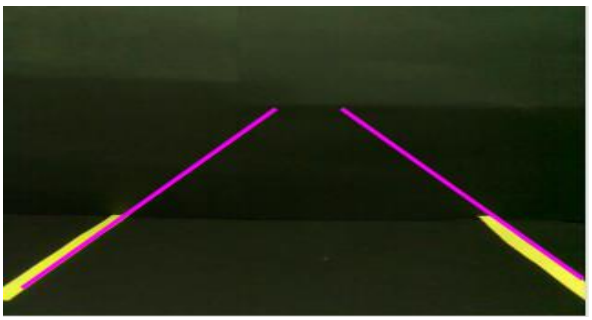

圖 17.偵測黃色實線的畫面

5.2.3 實驗三:不同顏色之車道虛線偵測

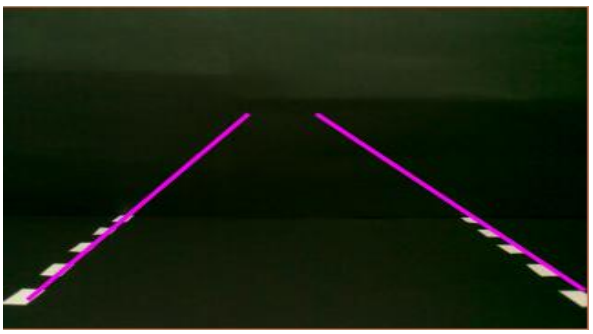

圖 18.偵測白色虛線的畫面

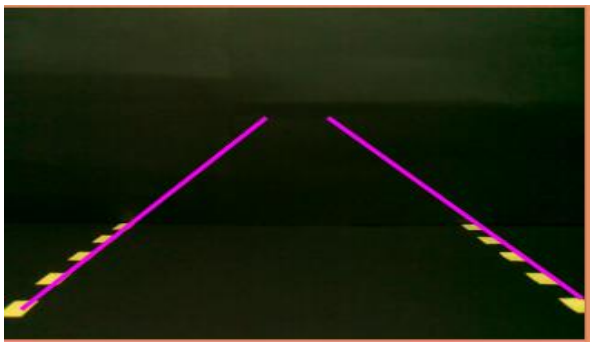

圖 19.偵測黃色虛線的畫面

5.2.4 實驗四:在不同光度下的車道實線偵測

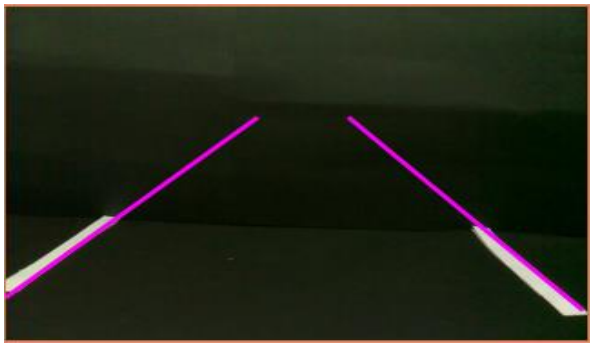

圖 20. 白色實線在光線較亮時的畫面

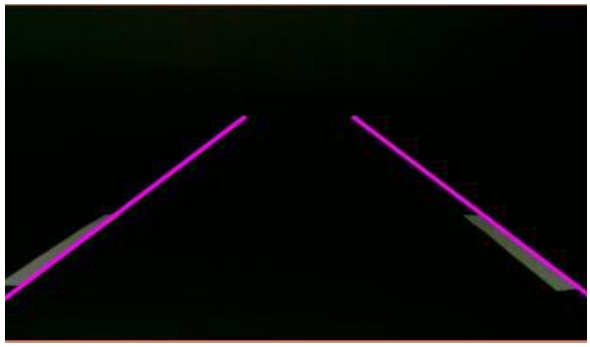

圖 21. 白色實線在光線較暗時的畫面

5.2.5 實驗五:彎道之車道偵測

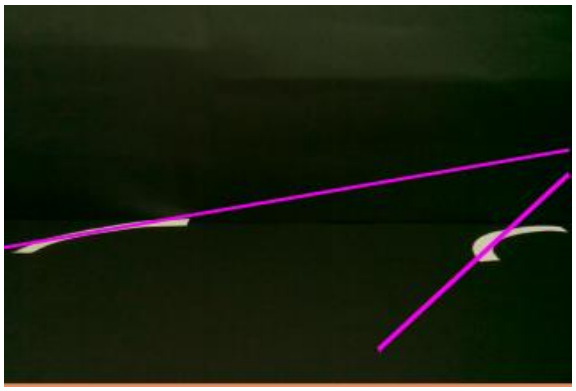

圖 22. 白色彎道實線的偵測畫面

5.3 休旅車的實際道路測試

由上方實驗可以知道對於一般模擬的道路狀況都能清楚有效的辨別出來，但是在是外需要抵抗更多外界的干擾，如行道樹、人行道、其他車輛等額外干擾，因此在狀況許可的情況下將測試移至戶外。但不同於室內以小客車為範本，本次實驗以較高的休旅車為實驗車輛。

此次實驗中分為車速與處理週期做為探討

5.3.1 車輛速度對偵測準確度的影響

表 3 以車速為變因之實驗數據
(處理週期 200ms)

車速(km/hr)	20	30	35	40
判斷正確(條)	1521	1033	797	451
判斷錯誤(條)	345	279	275	179
精確度(%)	81.5	78.7	74.3	71.6

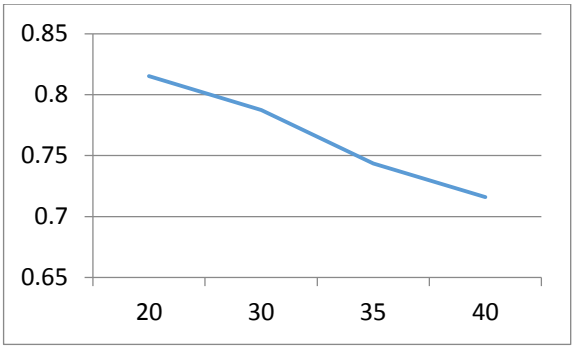

圖 23.車速與準確度折線圖
(處理週期 200ms)

5.3.2 處理週期對偵測準確度的影響

表 4 以車速為變因之實驗數據
(處理週期：100ms)

車速(km/hr)	20	30	40
判斷正確(條)	1501	967	921
判斷錯誤(條)	367	361	673
精確度(%)	80.4	72.8	57.8

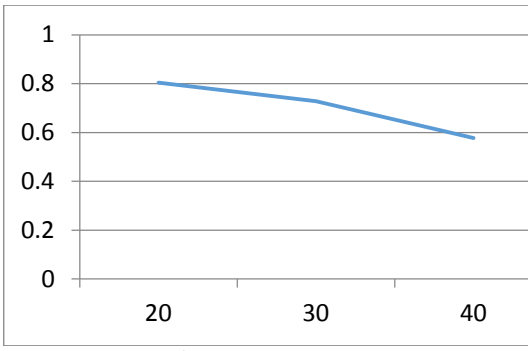

圖 23.車速與準確度折線圖
(處理週期 100ms)

6. 結論

在此次論文中，藉由 Raspberry Pi 主機板體積小的優勢，在利用 Matlab/simulink 模組化的方法將車道偏移警示系統完成。

在車道線為直線的情況下，無論是實線、虛線、各種顏色的車道線、或者因為光源影響導致車道線沒有很清楚時，本研究的車道偏移警示系統都能夠非常的清楚顯示出線的位置。但是在對於彎道的情況下，由實驗結果可以知道，彎道的偵測由於是直接使用直線式的霍夫轉換來近似，所以只會標記出圓弧上的弦做為警示。而在實車測試的情況也了解到車速越快或處理時間縮短時會導致穩定度下滑，尤其當處理時間縮短下滑的效率非常驚人。

雖然有一些缺失，但是經過這次嘗試可以知道在樹梅派上以模組化概念建立車輛偏移警示系統是可行的。緊接著會針對系統速度較慢跟對於非晴天的情況做測試與修改，以期達到反應快速且準確的警示系統。

參考文獻

[1] 交通部統計查詢網.線上檢索時間：2015 年 4 月 12 日取自：http://stat.motc.gov.tw/mocdb/stmain.jsp?sys=100.

[2] 王昭祥. 即時車道偵測與追蹤. 義守大學電子工程學研究所碩士論文, 2005.

[3] 范耿豪. 以霍夫轉換為基礎之智慧型快速車道線偵測. 國立臺灣師範大學應用電子科學系碩士論文, 2009.

[4] 郭宗彥. 以嵌入式平台實現車道偵測系統. 長庚大學電子工程學研究所碩士論文, 2008.

[5] 洪國振. 基於 WinCE 平台實現嵌入式車道偏移警示系統. 國立高雄大學電機工程學系碩士論文,2010.

[6] 魏仟豪. 在嵌入式異質雙核心平台上整合開發夜間車燈與車道追蹤偵測與事件記錄功能之駕駛輔助系統. 國立台北科技大學資訊工程研究所碩士論文, 2012.