中華民國第 62 屆中小學科學展覽會

作品說明書

高級中等學校組　電腦與資訊學科

第一名

052507

它罩得住我

學校名稱：高雄市私立中山高級工商職業學校

作者：	指導老師：
職二 朱育陞 職二 吳國維 職二 孫加恩	楊鎮澤

關鍵詞：內輪差、物件識別、盲區偵測

得獎感言

換個角度看問題，創造更佳的用路安全裝置

很高興今天我們有機會進入全國科學展覽的比賽，得名應該是每個高中生的目標，我們在視訊評審的階段每天不斷的練習，希望比賽的當天不要出錯，經過這一段時間的訓練，我們的口才也慢慢變好了，很幸運的我們在比賽當天沒有因為人為的失誤讓整體報告進度拖延。

我們在過程中碰到Yolo物件辨識程式無法正常辨識，我們搜尋相關資料及上網爬文後，找到了問題，更換資料集及更改參數後系統才能夠達到我們需要的辨識速度與準確度；在雷達感測器的實驗過程中，我們也碰到了無法使用單一個nano板來進行雷達的連接，必須透過串接另一塊開發板才有辦法解決這個問題；在程式語言的溝通上也碰到了一大難題，不同的語言要執行同一個動作的用法有些差異，我們也花了很多時間來研究這個問題。

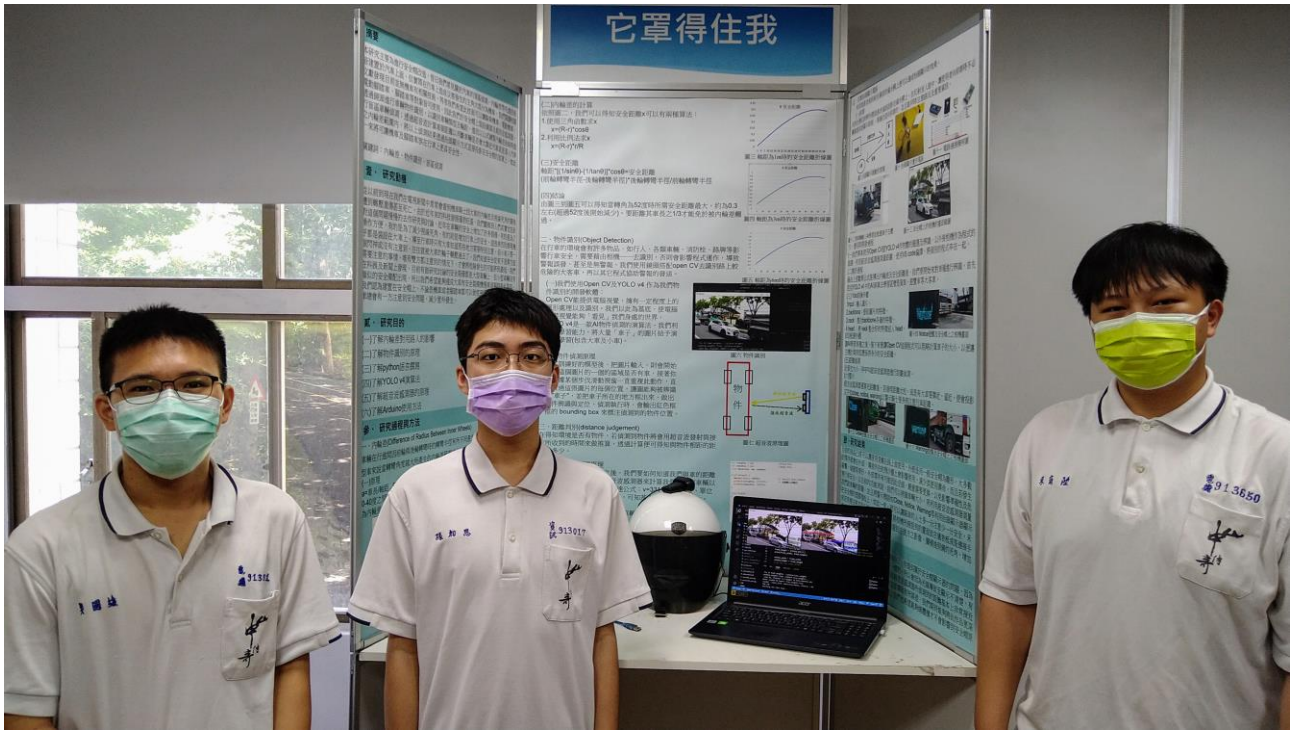
在參與科展過後，我們得到以下幾個啟發：

一、學習到如何更精確地蒐集資料，不再侷限於中文論壇，使用英文去尋找相關資訊，確保資訊正確無誤。

二、程式邏輯以及撰寫，讓我們可以運用 C 及 Python 控制雷達及抬頭顯示器並整合與電腦溝通。

三、學習到如何幫助隊員練習演講，自己也學習到演講的技巧，使團隊可以應對評審所提出的問題，讓我們的研究成果可以使評審完全了解。

最後，我們也學會如何做更妥善的時間運用，因為比賽準備過程中多次遇到段考以及其他不同事情，讓我們焦頭爛額，所以選擇將較不重要的事情排除後，再進行段考讀書、科展以及其他重要事項的時間分配，才不至於顧此失彼。另外，感謝老師給予我們機會可以參與此次的比賽，也很感謝隊友的互相扶持及鼓勵，感謝實驗室學長的教導，因為他們，我們才有機會踏上榮譽的殿堂。



展板前練習

程式測試

程式及安全帽測試

摘要

本研究主要為進行安全帽改造，昔日我們常見關於汽車的盲區偵測、內輪差等先進技術皆建置於汽車上面，但實際在行車上造成災害發生的主角大部分為機車，我們閱讀相關文獻發現目前並無機車有相關技術，再者我們希望此技術可以讓騎乘機車、電動機車、電動腳踏車、腳踏車等對象皆可使用，因此我們於安全帽上裝設鏡頭及雷達感測器，透過鏡頭進行車輛物件識別，以識別車輛類型及輪距，進而以公式繪製內輪差曲線與進行盲區車輛偵測；透過雷達計算車側距離以判斷車輛是否會太靠近汽車或是落入汽車之內輪差範圍內，將以上偵測結果透過抬頭顯示方式直接投影在安全帽的面罩上，如此一來將可讓機車及腳踏車族在行車上更具安全性。

壹、前言

一、研究動機

從以前到現在我們在電視新聞中常常會看到機車騎士因大車的內輪差及視線死角的關係遭到輾壓重傷甚至死亡，由於近年來的科技發展蓬勃旺盛，我們觀察到人們其實也在針對這個問題慢慢的去作研究與討論，近年在車輛的安全上增加了許多裝備，有的是為了操作方便，有的是為了減少視線死角，有的則是增加行車上的安全，這些東西我發現幾乎都是裝設在大車上，甚至行車時只有大車知道那些地方需要特別注意，但機車只要一個閃神或沒有注意到，往往就被大車的輪子輾壓過去了。我們知道安全從來不是單方面需要注意的事情，唯有雙方都注意到的情況下，才會將危險發生的可能將到最低。我們在歷屆科展及新聞上發現，目前有做研究討論的安全裝備都是大型車輛上，但機踏車沒有類似的安全備配出現，所以我們希望能夠達成大車有安全裝備，機慢車或腳踏車也要有，我們認為建置在安全帽上，不論是機車或者腳踏車都可以做使用，這樣不管是大車或小車總會有一方注意到安全問題，減少意外發生。

本研究主題運用到的學校課程有以下幾門課：

A、可程式控制實習：此課程中我們學習到了該如何去畫出流程圖，然後透過編排及運用電腦邏輯的方式去找出答案，書中我們可以學習到最多的就是邏輯觀念了，要設計出能夠獨立運作的程式，那邏輯必須要很厲害，把整個問題統整出來進一步的抽絲剝繭找到答案，那是多麼不容易的事情。當然課程最吸引我們的是在邏輯釐清的部分，讓我們不斷的動腦提問，最後找到答案，當找到解答時那是非常有成就感的事情。

B、單晶片微處理器實習 ：我們學校學到的是 Arduino 開發板，因此我們利用它來

當成我們程式實現的對象，我們學習透過各種感測元件與電路板溝通，讓電路板做出對應的動作，透過 Arduino IDE 做程式編輯並把程式上傳，將最終的結果呈現在 Arduino 開發板上 。

C、基本電學實習：本課程我們運用到了電源供應器的操作、麵包板的使用以及元件規格的識別與相關特性，讓我們在測試時不用常常把線重複焊接或更換接線方式，只要透過統整線路後測試完成，最後再把成品定型，這樣不但省時又省力，減少錯誤的發生，讓我們在開發時更順暢。

二、目的

我們希望透過此次製作，讓安全帽擁有其他智慧功能，讓注意者從大車司機轉為騎士，甚至是雙方都有注意到的情境下。使騎士在車輛眾多的大路上，不會因為某一方的沒注意或判斷錯誤而導致騎士或家屬憾慟一生。

我們主要想研究的目標如下：

(一)了解內輪差對用路時的影響。

(二)了解物件識別的原理。

(三)了解 python 語言撰寫。

(四)了解 YOLO v4 演算法。

(五)了解雷達感測器的原理。

(六)了解 Arduino 使用方法。

三、文獻探討

(一) 智慧影像，「One」視平安~大型車轉彎安全偵測警示系統

程式最後佈署於 NVIDIA Jason Nano Ubuntu 系統下並安裝於大型車輛。能明確得知人車種類，明確標定位置，並能立即判定有無危險，更立即以語音警告駕駛，請不干擾駕駛的視覺注意力。(智慧影像，「One」視平安~大型車轉彎安全偵測警示系統。2021)

由此我們可以得知目前的討論幾乎都是在大型車輛上，而這些配備也將被大型車輛使用，我們認為不只是大車需要注意這類的安全問題，若大型車與機車都有安裝這類的配備，將可以大大減少用路人的視線死角問題，增加生存機會減少傷亡產生。

(二)「友善超車，請給單車 1.5 公尺」在台灣可行嗎？

單車 1.5 公尺的安全距離是從德國開始的，而那時法規早已規範了此事，而現今歐洲多國也都立法明訂了超越單車的橫向安全距離，如單車王國荷蘭超越單車必須保留至少 1m 的安全距離。(The News Lens 關鍵評論。2019)

由此我們可以得知，機慢車與車輛的最大安全距離約為 1.5M，但就目前臺灣道路現況來說，要保持 1.5M 的距離確實有些難度，我們只可以盡量的遠離大型車輛，才能確保自身的安全，避免被內輪差或大型車行進時的吸力給捲進了車底造成傷亡。

貳、研究設備及器材

硬體	
筆記型電腦	 圖 1 筆記型電腦(圖片來源：Google)
雷達感測器	 圖 2 雷達感測器(圖片來源：TFmini 官網)
鏡頭	 圖 3 鏡頭(圖片來源：作者自攝)
OLED	 圖 4 OLED(圖片來源：作者自攝)
安全帽	 圖 5 安全帽(圖片來源：作者自攝)

軟體	
OpenCV(電腦視覺)	 圖 6 Open CV(圖片來源：OpenCV 官網)
Yolov4(偵測模型)	 圖 7 YOLO(圖片來源：YoLo 官網)
Python	 圖 8 Python(圖片來源：Python 官網)
Arduino DE	 圖 9 Arduino IDE(圖片來源：Arduino 官網)

參、研究過程或方法

一、內輪差(Difference of Radius Between Inner Wheels)

車輛在行進間因前輪與後輪轉彎時的轉彎半徑有所不同產生偏移所導致的盲區，對大型車來說當轉彎角度越大所產生的內輪差範圍越大，不止大車行車上有危險，對於騎機車、腳踏車甚至是人行道上的行人都產生了很大的威脅。

圖 10 內輪差(圖片來源：自由時報)

(一)原理

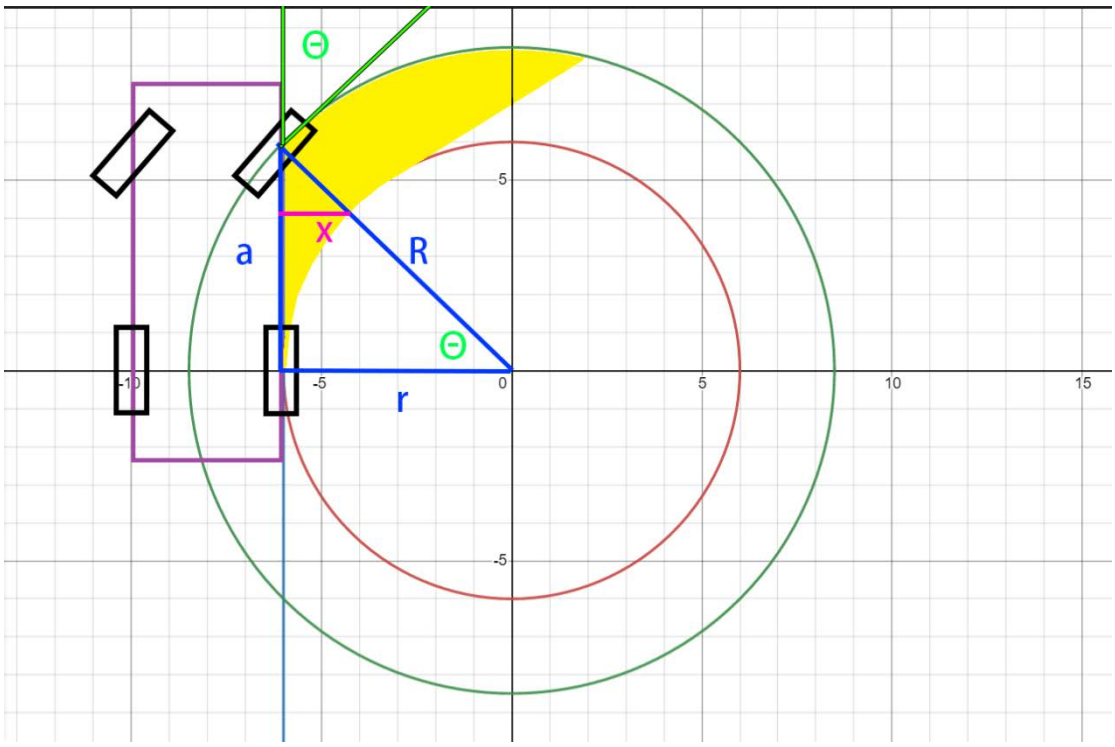


圖 11 內輪差關係圖(圖片來源：使用 desmos 繪製)

a=車長 or 軸距，R=前輪轉彎半徑，r=後輪轉彎半徑，x=安全距離，θ=轉彎角

度(θ 值會在 0-40 度之間，因不同車種而有所差異，有些特殊的賽車甚至可以轉至 55 度)(黃色區域即為內輪差的面積)

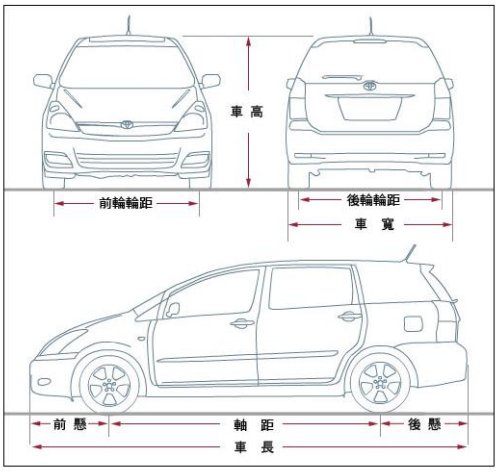


圖 12 車子的各部位名稱(圖片來源 https://classroom.u-car.com.tw/classroom-detail.asp?cid=33)

(二)內輪差的計算

依照圖 11，我們可以得知安全距離 x 可以有兩種算法：

1.使用三角函數求 x

圖中有以 r 為底的三角形和以 x 為底的三角形，當我們只知道車長或軸距時，可以透過 a/sinθ =R、a/cosθ=r 得知前後輪轉彎半徑，並且(R-r)為以 x 為底的三角形之斜邊，即可以藉由車長或軸長以及轉彎幅度來判斷其安全距離為何。由上可以得知：x=(R-r)*cosθ

2.利用比例法求 x

圖中有以 r 為底的三角形和以 x 為底的三角形，依照 aa 相似原理可以得知此兩個三角形的邊長是成比例關係的，當我們只知道車長或軸距時，可以透過 a/sinθ=R、a/cosθ=r 得知前後輪轉彎半徑，且(R-r)為以 x 為底的三角形之斜邊，就可以得知： r : x = R : (R-r)，因此 x=(R-r)*r/R。

(三)安全距離

軸距*[(1/sinθ)-(1/tanθ)]*cosθ=安全距離

圖 13 三角函數求安全距離 excel 計算公式

(前輪轉彎半徑-後輪轉彎半徑)*後輪轉彎半徑/前輪轉彎半徑

圖 14 比例法求安全距離 excel 計算公式

利用三角函數與比例求安全距離，軸距為 1 時求安全距離

三角函數求安全距離

a 軸距	R 前輪轉彎半徑	r 後輪轉彎半徑	R-r內輪差	角度	X 安全距離
1	57.2986885	57.28996163	0.008726868	1	0.008725539
1	28.65370835	28.63625328	0.017455065	2	0.017444432
1	19.10732261	19.08113669	0.026185922	3	0.026150035
1	14.33558703	14.30066626	0.034920769	4	0.034835704
1	11.47371325	11.4300523	0.043660943	5	0.0434948
1	9.566772234	9.514364454	0.052407779	6	0.052120684
1	8.205509048	8.144346428	0.06116262	7	0.060706723
1	7.185296534	7.115369722	0.069926812	8	0.069246289
1	6.392453221	6.313751515	0.078701707	9	0.077732758
1	5.758770483	5.67128182	0.087488664	10	0.086159514
1	5.240843064	5.144554016	0.096289048	11	0.094519947
1	4.809734345	4.704630109	0.105104235	12	0.102807456
1	4.445411483	4.331475874	0.113935608	13	0.111015446
1	4.133565494	4.010780934	0.122784561	14	0.119137335
1	3.863703305	3.732050808	0.131652498	15	0.127166548
1	3.627955279	3.487414444	0.140540835	16	0.135096521
1	3.42030362	3.270852618	0.149451001	17	0.142920703
1	3.236067977	3.077683537	0.15838444	18	0.150632554
1	3.071553487	2.904210878	0.167342609	19	0.158225545
1	2.9238044	2.747477419	0.176326981	20	0.165693163
1	2.79042811	2.605089065	0.185339045	21	0.173028905
1	2.669467163	2.475086853	0.194380309	22	0.180226284
1	2.559304665	2.355852366	0.203452299	23	0.187278829
1	2.458593336	2.246036774	0.212556562	24	0.194180081
1	2.366201583	2.144506921	0.221694663	25	0.200923599
1	2.281172033	2.050303842	0.230868191	26	0.207502956
1	2.202689265	1.962610506	0.240078759	27	0.213911741
1	2.130054468	1.880726465	0.249328003	28	0.22014356
1	2.06266534	1.804047755	0.258617584	29	0.226192036
1	2	1.732050808	0.267949192	30	0.232050808

比例求安全距離

a 軸距	R 前輪轉彎半徑	r 後輪轉彎半徑	R-r內輪差	角度	X 安全距離
1	57.2986885	57.28996163	0.008726868	1	0.008725539
1	28.65370835	28.63625328	0.017455065	2	0.017444432
1	19.10732261	19.08113669	0.026185922	3	0.026150035
1	14.33558703	14.30066626	0.034920769	4	0.034835704
1	11.47371325	11.4300523	0.043660943	5	0.0434948
1	9.566772234	9.514364454	0.052407779	6	0.052120684
1	8.205509048	8.144346428	0.06116262	7	0.060706723
1	7.185296534	7.115369722	0.069926812	8	0.069246289
1	6.392453221	6.313751515	0.078701707	9	0.077732758
1	5.758770483	5.67128182	0.087488664	10	0.086159514
1	5.240843064	5.144554016	0.096289048	11	0.094519947
1	4.809734345	4.704630109	0.105104235	12	0.102807456
1	4.445411483	4.331475874	0.113935608	13	0.111015446
1	4.133565494	4.010780934	0.122784561	14	0.119137335
1	3.863703305	3.732050808	0.131652498	15	0.127166548
1	3.627955279	3.487414444	0.140540835	16	0.135096521
1	3.42030362	3.270852618	0.149451001	17	0.142920703
1	3.236067977	3.077683537	0.15838444	18	0.150632554
1	3.071553487	2.904210878	0.167342609	19	0.158225545
1	2.9238044	2.747477419	0.176326981	20	0.165693163
1	2.79042811	2.605089065	0.185339045	21	0.173028905
1	2.669467163	2.475086853	0.194380309	22	0.180226284
1	2.559304665	2.355852366	0.203452299	23	0.187278829
1	2.458593336	2.246036774	0.212556562	24	0.194180081
1	2.366201583	2.144506921	0.221694663	25	0.200923599
1	2.281172033	2.050303842	0.230868191	26	0.207502956
1	2.202689265	1.962610506	0.240078759	27	0.213911741
1	2.130054468	1.880726465	0.249328003	28	0.22014356
1	2.06266534	1.804047755	0.258617584	29	0.226192036
1	2	1.732050808	0.267949192	30	0.232050808

圖 15 計算總表---1 度到 30 度(圖片來源：excel 截圖)

1	1.941604026	1.664279482	0.277324544	31	0.237713531		1	1.941604026	1.664279482	0.277324544	31	0.237713531
1	1.887079915	1.600334529	0.286745386	32	0.243173878		1	1.887079915	1.600334529	0.286745386	32	0.243173878
1	1.836078459	1.539864964	0.296213495	33	0.24842554		1	1.836078459	1.539864964	0.296213495	33	0.24842554
1	1.78829165	1.482560969	0.305730681	34	0.253462222		1	1.78829165	1.482560969	0.305730681	34	0.253462222
1	1.743446796	1.428148007	0.315298789	35	0.258277647		1	1.743446796	1.428148007	0.315298789	35	0.258277647
1	1.701301617	1.37638192	0.324919696	36	0.262865556		1	1.701301617	1.37638192	0.324919696	36	0.262865556
1	1.661640141	1.327044822	0.33459532	37	0.267219704		1	1.661640141	1.327044822	0.33459532	37	0.267219704
1	1.624269245	1.279941632	0.344327613	38	0.271333862		1	1.624269245	1.279941632	0.344327613	38	0.271333862
1	1.589015729	1.234897157	0.354118573	39	0.275201819		1	1.589015729	1.234897157	0.354118573	39	0.275201819
1	1.555723827	1.191753593	0.363970234	40	0.278817375		1	1.555723827	1.191753593	0.363970234	40	0.278817375
1	1.524253087	1.150368407	0.373884679	41	0.28217435		1	1.524253087	1.150368407	0.373884679	41	0.28217435
1	1.49447655	1.110612515	0.383864035	42	0.285266571		1	1.49447655	1.110612515	0.383864035	42	0.285266571
1	1.466279186	1.07236871	0.393910476	43	0.288087884		1	1.466279186	1.07236871	0.393910476	43	0.288087884
1	1.43955654	1.035530314	0.404026226	44	0.290632145		1	1.43955654	1.035530314	0.404026226	44	0.290632145
1	1.414213562	1	0.414213562	45	0.292893219		1	1.414213562	1	0.414213562	45	0.292893219
1	1.390163591	0.965688775	0.424474816	46	0.294864984		1	1.390163591	0.965688775	0.424474816	46	0.294864984
1	1.367327461	0.932515086	0.434812375	47	0.296541327		1	1.367327461	0.932515086	0.434812375	47	0.296541327
1	1.34563273	0.900404044	0.445228685	48	0.29791614		1	1.34563273	0.900404044	0.445228685	48	0.29791614
1	1.325012993	0.869286738	0.455726256	49	0.298983325		1	1.325012993	0.869286738	0.455726256	49	0.298983325
1	1.305407289	0.839099631	0.466307658	50	0.299736785		1	1.305407289	0.839099631	0.466307658	50	0.299736785
1	1.286759566	0.809784033	0.476975533	51	0.300170429		1	1.286759566	0.809784033	0.476975533	51	0.300170429
1	1.269018215	0.781285627	0.487732589	52	0.300278165		1	1.269018215	0.781285627	0.487732589	52	0.300278165
1	1.252135658	0.75355405	0.498581608	53	0.300053902		1	1.252135658	0.75355405	0.498581608	53	0.300053902
1	1.236067977	0.726542528	0.509525449	54	0.299491545		1	1.236067977	0.726542528	0.509525449	54	0.299491545
1	1.220774589	0.700207538	0.520567051	55	0.298584994		1	1.220774589	0.700207538	0.520567051	55	0.298584994
1	1.206217949	0.674508517	0.531709432	56	0.297328141		1	1.206217949	0.674508517	0.531709432	56	0.297328141
1	1.192363293	0.649407593	0.5429557	57	0.295714868		1	1.192363293	0.649407593	0.5429557	57	0.295714868
1	1.179178403	0.624869352	0.554309051	58	0.293739045		1	1.179178403	0.624869352	0.554309051	58	0.293739045
1	1.166633397	0.600860619	0.565772778	59	0.291394523		1	1.166633397	0.600860619	0.565772778	59	0.291394523
1	1.154700538	0.577350269	0.577350269	60	0.288675135		1	1.154700538	0.577350269	0.577350269	60	0.288675135

圖 16 計算總表---31 度到 60 度

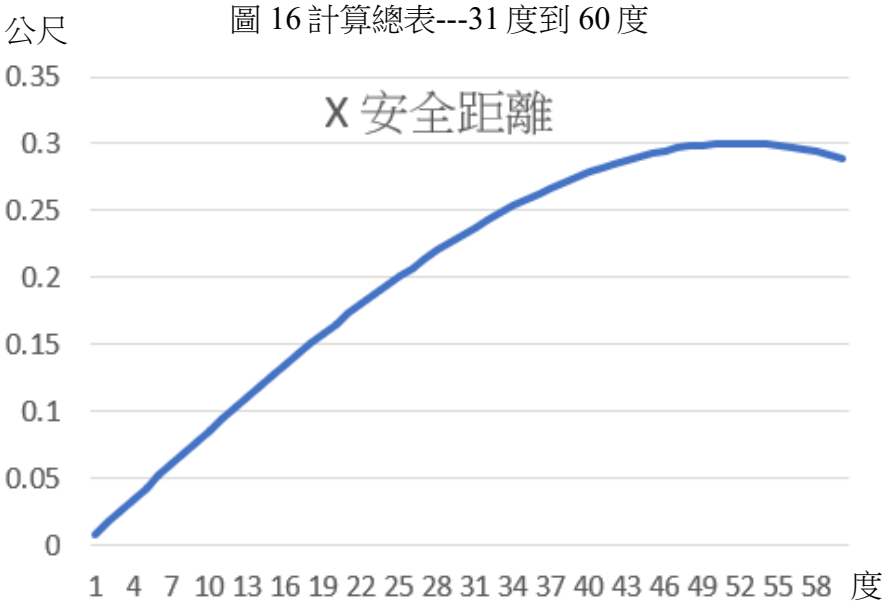


圖 17 圖 15 和圖 16 資料之折線圖

依上述圖表我們可以得知車輛轉彎角度為 52 度時，所需保持的安全距離最大。

(四)不同車長/軸距比較

接下來我們利用 excel 計算不同軸距下應保持之安全距離

利用三角函數與比例求安全距離，軸距為 3 時求安全距離

a 軸距	R 前輪轉彎半徑	r 後輪轉彎半徑	R-r內輪差	角度	X 安全距離
3	34.42113974	34.29015691	0.130982829	5	0.1304844
3	17.27631145	17.01384546	0.262465991	10	0.258478542
3	11.59110992	11.19615242	0.394957493	15	0.381499643
3	8.7714132	8.242432258	0.528980942	20	0.497079488
3	7.098604749	6.433520762	0.665083988	25	0.602770797
3	6	5.196152423	0.803847577	30	0.696152423
3	5.230340387	4.28444402	0.945896367	35	0.774832942
3	4.667171481	3.575260778	1.091910703	40	0.836452126
3	4.242640687	3	1.242640687	45	0.878679656
3	3.916221868	2.517298894	1.398922974	50	0.899210355
3	3.662323766	2.100622615	1.561701152	55	0.895754981
3	3.464101615	1.732050808	1.732050808	60	0.866025404

圖 18 軸距為 3 時的安全距離

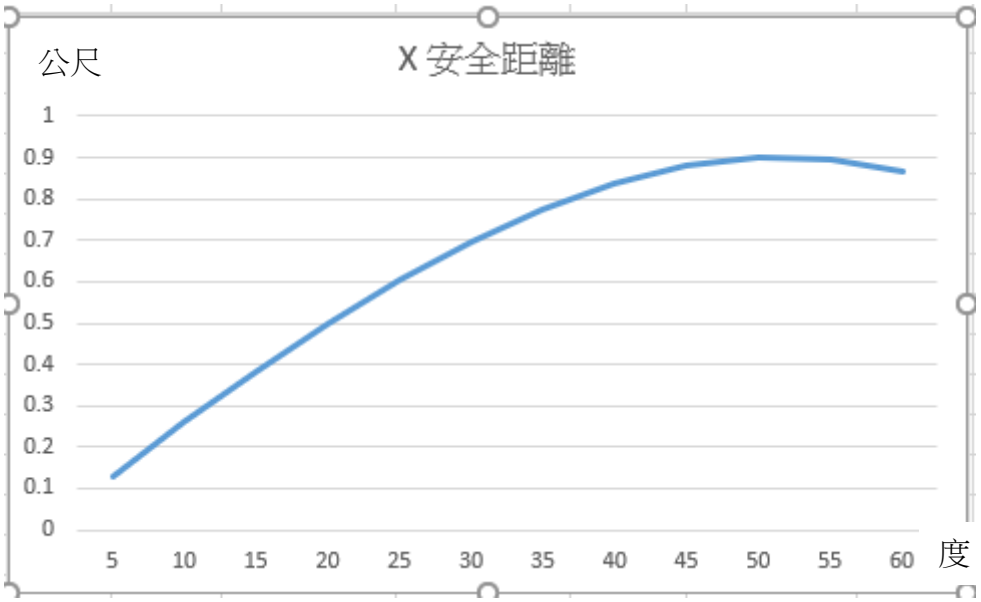


圖 19 軸距為 3 時的安全距離折線圖

利用三角函數與比例求安全距離，軸距為 6 時求安全距離

a 軸距	R 前輪轉彎半徑	r 後輪轉彎半徑	R-r內輪差	角度	X 安全距離
6	68.84227947	68.58031382	0.261965657	5	0.260968799
6	34.5526229	34.02769092	0.524931981	10	0.516957085
6	23.18221983	22.39230485	0.789914986	15	0.762999285
6	17.5428264	16.48486452	1.057961884	20	0.994158976
6	14.1972095	12.86704152	1.330167976	25	1.205541595
6	12	10.39230485	1.607695155	30	1.392304845
6	10.46068077	8.56888804	1.891792733	35	1.549665885
6	9.334342961	7.150521556	2.183821406	40	1.672904253
6	8.485281374	6	2.485281374	45	1.757359313
6	7.832443736	5.034597787	2.797845949	50	1.79842071
6	7.324647533	4.201245229	3.123402303	55	1.791509962
6	6.92820323	3.464101615	3.464101615	60	1.732050808

圖 20 軸距為 6 時的安全距離

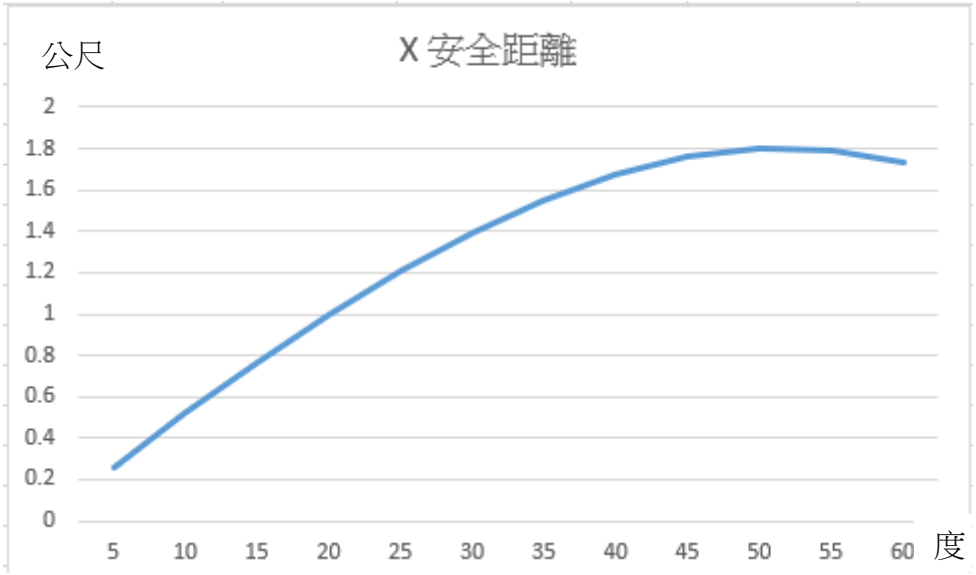


圖 21 軸距為 6 時的安全距離折線圖

(五)結論

由圖 15、圖 16 可以得知當轉角為 52 度時所需安全距離最大，約為 0.3 左右(超過 52 度後開始減少)。在圖 18、圖 20 可以得知軸距為 3m 時安全距離最大為 0.89m；軸距為 6m 時最大安全距離為 1.79m，皆約為軸距的 1/3。由此可知在面對大多數車輛時，要距離其車長之 1/3 才能免於被內輪差輾過。

二、物件識別(Object Detection)

在行車的環境會有許多物品，如行人、各類車輛、消防栓、路牌等影響行車安全，需要藉由相機一一去識別，否則會影響程式運作，導致警報誤發、甚至是無警報，我們使用鏡頭搭配 open CV 去識別路上較危險的大客車，在以其它程式協助警報的發送。

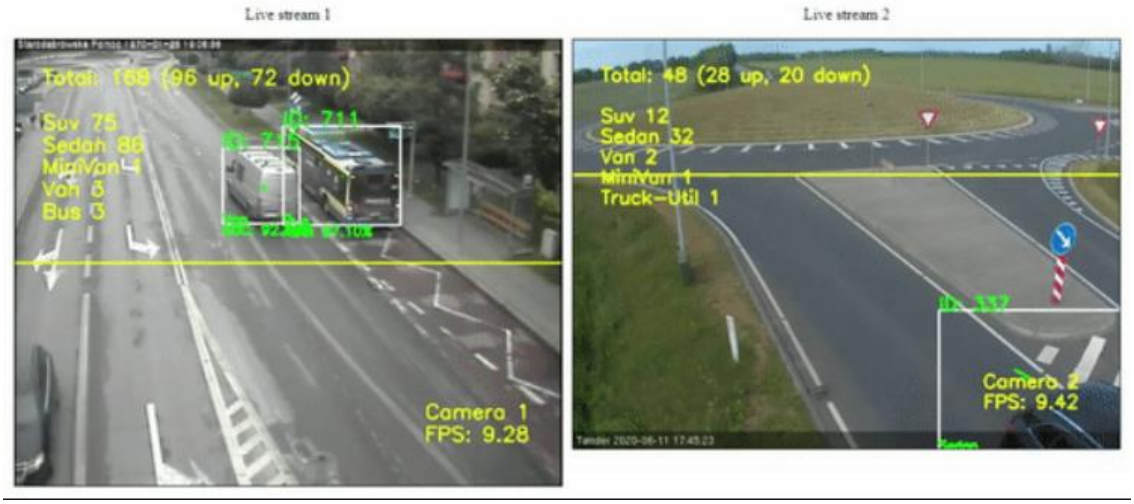


圖 22 物件識別(圖片來源：GitHub LeonLok)

(一)我們使用 Open CV 及 YOLO v4 作為我們物件識別的開發軟體：

Open CV 能提供電腦視覺，擁有一定程度上的圖形處理以及識別。我們以此為基底，使電腦具備視覺能夠「看見」我們身處的世界，讓他有判別物品的能力，並且其開源性使我們可以搜尋到更多資源做為參考，使程式的運作更為流暢。

YOLO v4 是一款 AI 物件偵測的演算法，我們利用其學習能力，將大量「車子」的圖片給予演算法學習(包含大車及小車)，使其在 Open CV 的基礎上有更進一步的物件判別，並且由於其處理速度快且精確，所以並不太會出現延遲而導致意外發生，也因為他是開源軟體，且有前人的經歷可做參考，使我們可以更簡易的去開發他。

(二)物件偵測原理

有了訓練好的模型後，把圖片輸入，則會開始偵測這個圖片的一個的區域是否有車，接著你會根據某個步伐滑動視窗一直重複此動作，直到滑過這張圖片的每個位置，讓圖能夠被辨識出"車子"，並把車子所在的地方框出來，做出物件辨識與定位，偵測執行時，會輸出紅色框框的 bounding box 來標注偵測到的物件位置。

三、距離判別(distance judgement)

在得知環境是否有物件，若偵測到物件將會用雷達發射與接收所收到的時間來做推算，透過計算便可得知與物件相距的距離為多少，再經過系統判斷句離是否太近，若

太近將發送警報給予使用者知道。

四、雷達感測器原理

在可以判別大車與小車之後，我們要如何知道我們與車的距離呢？所以我們使用兩個雷達感測器來計算我們和左側車輛以及後方車輛的距離。根據光速為 3 X 10^8 m/s，所以便可以利用光的反射時長來計算出距離：距離 =（光速 x 反射時間）/ 2，以此得知我們和車的距離為幾公尺，方便做出警訊。

雷達主要是用來感測是否有車輛在使用者左側與後方，以避免視線死角導致事故發生的機會，TF-MINI 雷達模組最遠可以探測到最遠 12 公尺最近 10 公分，運作原理為發射光波再利用公式計算出物體跟雷達本身的距離。

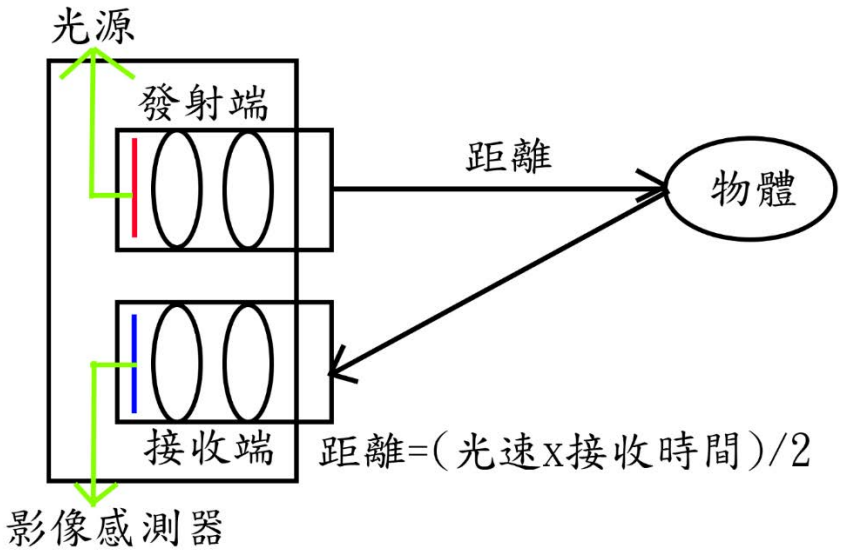


圖 23 雷達原理圖

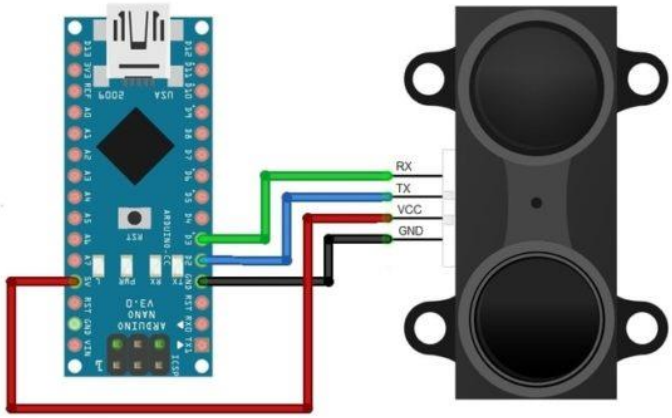


圖 24 雷達感測器與 Arduino 板連接

五、自製抬頭顯示電路

利用小型螢幕並將其固定在安全帽上側，使騎士可以很方便的讀取警訊。

(一) 原理

使用 0.96 OLED 並將其固定在安全帽上，以 45 度角將影像反射於安全帽鏡片的偏光膜上，讓使用者向前看時不必轉頭看其他顯示面板，視線仍是向前看的，並且能同時注意路況及重要資訊。

圖 25 抬頭顯示器動作原理

(二) 抬頭顯示電路製作

1.繪製電路簡明圖，將 Arduino nano、OLED、鋰電池及藍芽晶片以杜邦線進行連接。

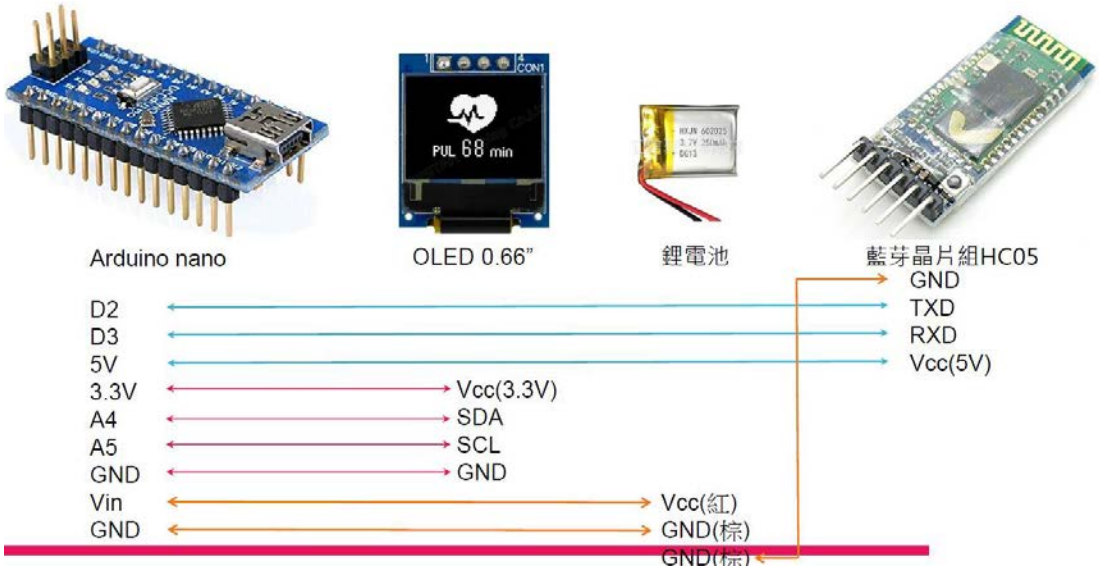


圖 26 電路連接簡明圖

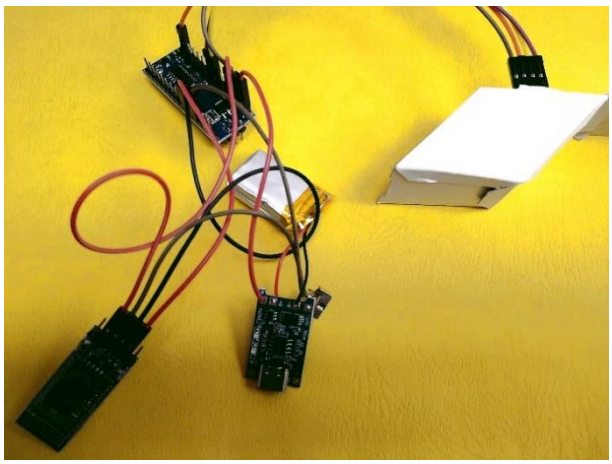


圖 27 抬頭顯示實作電路

2.以 Arduino IDE 撰寫程式，將程式匯入晶片進行測試。

```
void notice() {
  display.clearDisplay();
  display.setTextSize(2); // Draw 2X-scale text
  display.setTextColor(SSD1306_WHITE);
  display.println(F("notice"));
  display.setCursor(10, 0);
  display.display();
  delay(100);
}

void closed() {
  display.clearDisplay();
  display.setTextSize(2); // Draw 2X-scale text
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(10, 15);
  display.println(F("closed"));
  display.display();
  delay(100);

}
void warning() {
  display.clearDisplay();
  display.setTextSize(2); // Draw 2X-scale text
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(10, 15);
  display.println(F("warning"));
  display.display();
  delay(100);
```

圖 28 抬頭顯示程式碼

六、應用與開發過程

(一)我們專案使用 Open CV 跟 YOLO v4 作物體的圈選及辨識，以外接鏡頭作為程式的眼睛，利用雷達感測器測量距離，使用 VS code 編譯，將個別的程式串在一起。其中參考了以下資料：

˙YOLO v4 模型訓練實作。

˙如何以 YOLOv3 訓練自己的資料集。

˙LabelImg 影像標註工具使用教學。

˙Tfmini Lidar。

˙交通安全宣導-內輪差。

我們搜尋以上資料來幫助我們了解認識內輪差，以及其他軟體及程式的使用，使硬體及軟體可以配合。

(二)製作過程

藉由圖 10 上述數學公式推導出內輪差及安全距離後，我們要開始來對車種進行辨識，首先使用 YOLO v4 中的 AI 演算法學習甚麼是貨車、遊覽車等大客車，先使用圖片來幫助程式學習並框出其外框，如圖 29、圖 30 以及圖 31。讓 AI 學習大車的輪廓，再藉由工具轉換為 XML 檔案以便程式判讀。

(三) Yolo 訓練步驟

1.input：輸入圖片 。

2.backbone：提取圖片的特徵。

3.neck：整合 backbone 各層的特徵。

4.head：將 neck 整合好的特徵送入 head。

以下我們使用 3 張我們在路邊拍攝的大車圖片讓其辨識

圖 29 將車輛輸入 AI 學習並框選車子主體

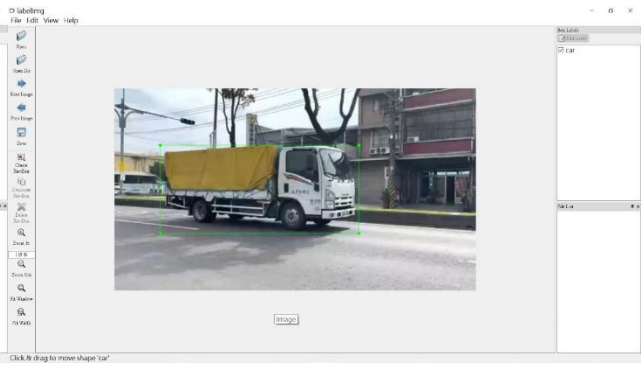


圖 30 將車輛輸入 AI 學習不同車種

圖 31 將車輛輸入 AI 學習並框選

sample > 0051.xml

```xml
<annotation>
  <folder>sample</folder>
  <filename>0051</filename>
  <path>D:/project/yolo_project/sample/0051.jpg</path>
  <source>
    <database>Unknown</database>
  </source>
  <size>
    <width>1920</width>
    <height>1440</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>car</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <bndbox>
      <xmin>86</xmin>
      <ymin>486</ymin>
      <xmax>1896</xmax>
      <ymax>1065</ymax>
    </bndbox>
  </object>
</annotation>
```

圖 32 將圖片標註後存檔即會轉換成__.xml 檔案供程式做判讀，此為其一判讀程式

(四)框選外觀

讓 AI 學習車種之後，接下來要讓 Open CV 這個程式可以粗略計算車子的大小，以便讓主機計算到底應保持多少的安全距離。

```
if len(cnts_up1 and cnts_up2 and cnts_side1 and cnts_side2) != 0:
    key = cv2.contourArea

    c1 = max(cnts_up1, key = cv2.contourArea)#找到面積最大的輪廓
    c2 = min(cnts_up2, key = cv2.contourArea)#找到面積最小的輪廓

    c3 = max(cnts_side1, key = cv2.contourArea)#找到面積最大的輪廓
    c4 = min(cnts_side2, key = cv2.contourArea)#找到面積最小的輪廓

    UP1 = cv2.moments(c1)
    UP2 = cv2.moments(c2)
    SIDE1 = cv2.moments(c3)
    SIDE2 = cv2.moments(c4)

    center_up_1 = (int(UP1["m10"]/UP1["m00"]), int(UP1["m01"]/UP1["m00"]))#計算質心
    center_up_2 = (int(UP2["m10"]/UP2["m00"]), int(UP2["m01"]/UP2["m00"]))#計算質心

    center_side_1 = (int(SIDE1["m10"]/SIDE1["m00"]), int(SIDE1["m01"]/SIDE1["m00"]))#計算質心
    center_side_2 = (int(SIDE2["m10"]/SIDE2["m00"]), int(SIDE2["m01"]/SIDE2["m00"]))#計算質心

    pointAx = int(UP1["m10"]/UP1["m00"])#(Ax,Ay)Redpoint
    pointAy = int(UP1["m01"]/UP1["m00"])
    pointBx = int(UP2["m10"]/UP2["m00"])#(Bx,By)Yellowpoint
    pointBy = int(UP2["m01"]/UP2["m00"])

    pointCx = int(SIDE1["m10"]/SIDE1["m00"])#(Ax,Ay)Redpoint
    pointCy = int(SIDE1["m01"]/SIDE1["m00"])
    pointDx = int(SIDE2["m10"]/SIDE2["m00"])#(Bx,By)Yellowpoint
```

圖 33 使用程式框出車輛

圖 34 利用安全帽上的相機的畫面截圖，其有框出每輛車的外觀

(五)距離偵測

計算完大小，再呼叫雷達感測器進行距離偵測主機

```
int radi1(){
  if (Serial.available()){
    if (Serial.read() == HEADER){
      uart[0] = HEADER;
      if (Serial.read() == HEADER){
        uart[1] = HEADER;uart2[1] = HEADER;
        for (i = 2; i < 9; i++){
          uart[i] = Serial.read();
        }
        check = uart[0] + uart[1] + uart[2] + uart[3] + uart[4] + uart[5] + uart[6] + uart[7];
        if (uart[8] == (check & 0xff)){
          dist = uart[2] + uart[3] * 256;     //calculate distance value
//          h = String(dist);
//          testscrolltext(h);
        }
      }
    }
  }
  return dist;
```

圖 35 雷達感測器運算距離圖

```
void loop(){
  r1=radi1();
  r2=radi2();
  if(digitalRead(2)==1){
    if(r1<=100){
      notice();
    }
  }else{
    if(r2<=100){
      warning();
    }else if(r2<=300){
      closed();
    }else{
      none();
    }
  }
}
```

```
15:00:30.407 -> 240cm/4cm
15:00:30.641 -> 241cm/4cm
15:00:30.827 -> 245cm/4cm
15:00:31.059 -> 244cm/4cm
15:00:31.248 -> 240cm/4cm
15:00:31.574 -> 2438cm/6cm
15:00:31.762 -> 245cm/7cm
15:00:31.996 -> 245cm/18cm
15:00:32.181 -> 244cm/57cm
15:00:32.694 -> 2479cm/2549cm
15:00:33.019 -> 246cm/2488cm
15:00:33.393 -> 246cm/2548cm
15:00:33.720 -> 245cm/2502cm
15:00:34.093 -> 244cm/2522cm
15:00:34.278 -> 239cm/46cm
```

圖 36 雷達感測器偵測距離及螢幕顯示判斷

(六)警示

雷達感測器運算完距離後，若發現距離太近，或是有大車要靠近、逼近，便會投影文字如 Close、Notice、Warning 以警示騎士要多加注意行車距離。

1.後方有來車，但無立即危險時顯示 Notice。

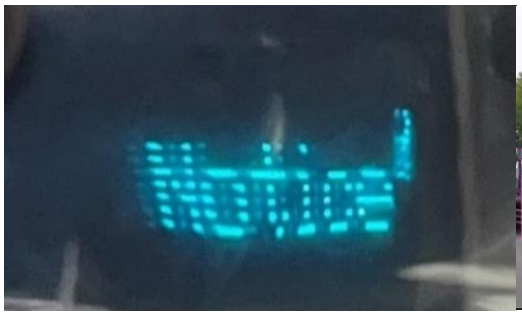


圖 37Notice 提醒

圖 38 安全帽上之相機畫面

2.車側車輛太靠近機慢車時顯示 Close，以避免跟汽車太過靠近發生擦撞。

圖 39Close 提醒

圖 40 安全帽上之相機畫面

3.當有大卡車接近時，進行內輪差計算，若暴露在危險區域中時顯示 Warning。

圖 41Warning 提醒

圖 42 安全帽上之相機畫面

肆、研究結果

目前的成品已經可以讓使用者戴在路上做使用，外殼是用一般安全帽為雛形，大多數的零件都會在外面，導致外在的飛沙塵土會影響使用、減少其使用壽命，而且若發生敲擊、碰撞等情形，外部零件有可能因此受損，嚴重需要更換，以免影響準確性及危害行車安全。以目前的功能來說，我們可以辨識車輛大小，再利用雷達感測器測量我們該車輛的距離，並且將警示語如 Close、 Notice、 Warning 等利用抬頭顯示器顯示在安全帽的鏡片上，如此一來，

就可以讓騎車的人士多一分注意少一分危險。未來也可將其鏡頭與螢幕連結，我便可以將相機所捕捉到的畫面放在儀表板或是連接手機投影在手機上，就可以讓騎士即時的看到左後方之影像，彌補後照鏡的死角，增加騎士的可視範圍，讓騎車更安全。

圖 43 安全帽成品

伍、討論

一、這樣的設計是否可以使用在人潮眾多的情下呢？

目前我們物件識別主要對象為車輛，當系統偵測到車輛時會利用方框進行匡列，因此不會將人匡列進來，但目前在沒有偵測到車輛 80%影像的情況下有時候會偵測到有時則不會，這是我們未來可以去優化判定，給予電腦自主學習的方向。

二、我們遇到的困難是什麼？

起初在架設辨識軟體時因有參考較舊的文獻，因應軟體不斷的更新，很多原本可以使用的程式語言有些變得要換另一種方式才能使用，我們之後參考了很多資料，拼湊出不同的組合，最後才把系統架設完成，在定義及偵測車輛大小時也到相同狀況，我們從車輛大小的判定公式一直到翻譯成電腦所能接受的語言花了不少功夫，在不斷的推導證明後我們一直懷疑是否有誤呢?在多次的求證下，我們找到了解答。

三、作品是否不侷限在機車上，而可以用到自行車及電動自行車上呢？

我們的構想就是因應現在有很多不同的交通方式，各式各樣的車都有，排除我們平

時最常見的四輪車之外，不只自行車、電動自行車，其實還有電動滑板車、普通的滑板等，只要不是在室內或車內，基本上都是可以戴上此安全帽來使用的。

四、安全帽的抬頭顯示器是否會因為外面的光線過強而導致顯示不清楚呢？

目前測試確實會因為陽光過強而影響顯示效果，但我們不斷的測試不同的材質所能過濾的光線亮度及效果是否能夠兼顧顯示畫面及行車視線，這是我們非常重視的部分也是未來亟待改善的地方。

陸、結論

安全帽目前已經到可以使用及調整的階段了，如上有提到關於安全帽顯示器的問題，因為不同材質所能穿透的光線亮度都是不一樣的，所以會因為光線導致至顯示不清楚，我們有找到幾個適合的材質，例如以現有的太陽眼鏡鏡片去製作鏡面。目前雷達感測器所偵測到的距離非常接近實際值，但有時也有可能因為外界的反射物質導致有所誤差，我們期待能夠將此作品更深入的去開發討論，不只軟體的進步，在硬體上我們也希望能夠縮小體積這樣才不會影響到安全帽原本的美觀，透過這個研究我們發現其實目前的科技都把所謂的安全技術著重在大型車輛上面，不只是倒車顯示器、超音波感測器，就連盲區紅外光顯示幾乎都是安裝在大型車輛上，就算這類型的車輛都安裝上了這些系統，但只要駕駛一個不注意就還是有可能釀下意外，相反如果這類型的系統建立在雙方的車輛或安全帽上，至少有一方可以注意到，而減少意外的發生。未來也許能夠用更快速的方式去做判斷，讓反應的時間更短，運算的速度更快，希望有一天大眾在使用此安全帽的產品時能夠提升安全的品質，減少受傷的機會。

柒、參考資料及其他

一、朱克剛(2021)。AIOT 與 OpenCV 實戰應用(第三版)：Python、樹莓派、物聯網與機器視覺。碁峯資訊股份有限公司。

二、數位新知(2021)。Python 程式設計：初心者超凡入門。深石數位科技股份有限公司。

三、曹永忠、許智誠、蔡英德(2016)。Arduino 程式教學(入門篇)。崧燁文化事業有限公司。

四、YOLO v4 模型訓練實作。I code so I am (2021)。
https://ithelp.ithome.com.tw/articles/10282549?sc=pt

五、LabelImg 影像標註工具使用教學，製作深度學習用的資料集。G. T. Wang(2017)。

https://blog.gtwang.org/useful-tools/labelimg-graphical-image-annotation-tool-tutorial/

六、交通安全宣導-內輪差。國立成功大學。

https://military-osa.ncku.edu.tw/p/404-1055-229624.php?Lang=zh-tw

七、「友善超車，請給單車 1.5 公尺」在台灣可行嗎？。潘柏翰編(2019)。

https://www.thenewslens.com/article/112033

八、智慧影像，「One」視平安，大型車轉彎安全偵測警示系統。顏妤晅、陳以華、曹家瑜(2021)。

https://twsf.ntsec.gov.tw/activity/race-1/61/pdf/NPHSF2021-032808.pdf?0.8211693671307927

九、TFmini LIDAR 12M。

http://en.benewake.com/?gclid=Cj0KCQjwqPGUBhDwARIsANNwjV4-Ydxra7xA9WptMVKXLCi68k5p-Olr1XkzRNwZAuCMx9PAq-iW_z4aAhvHEALw_wcB

【評語】052507

本作品整合攝影機、OpenCV、YOLO 影像物件辨識套件、雷達測距儀、OLED 螢幕和抬頭顯示器來設計與實作一個安全帽系統，以判斷目前機車是否會太靠近汽車或是落入汽車之危險內輪差範圍內。此作品已有實作出可運作之 prototype，且有內輪差的理論探討，作品的完整度和實用性佳。此外，此研究動機新穎實際，考慮台灣機車事故的需求進行系統的設計與建置。研究中對其使用情境 (內輪差等探討) 做過許多分析，也考慮了許多不同類型的車輛與輪子可能會需要在不同距離進行警示，探究內容深入詳盡，實作的成品可發展成為完整系統，具有實用的價值。

作品簡報

作品編號: 052507
作品組別: 高級中等學校組
科　　別: 電腦與資訊學科

它罩得住我

前言：研究動機

從以前到現在我們在電視新聞中常常會看到機車騎士因大車的內輪差及視線死角的關係遭到輾壓重傷甚至死亡，近年在車輛的安全上增加了許多裝備，我們發現幾乎都是裝設在大車上，甚至行車時只有大車知道那些地方需要特別注意，但機車只要一個閃神或沒有注意到，往往就被大車的輪子輾壓過去了。我們知道安全從來不是單方面需要注意的事情。我們在歷屆科展及新聞上發現，目前有做研究討論的安全裝備都是大型車輛上，但機慢車沒有類似的安全備配出現，所以我們希望能夠達成大車有安全裝備機慢車或腳踏車也要有，我們認為建置在安全帽上，不論是機車或者腳踏車都可以做使用，這樣不管是大車或小車總會有一方注意到安全問題，減少意外發生。

對機車族來說，安全帽是不可或缺的配備

圖片來源：自由時報

大型車因內輪差與視線死角，未注意到機車

研究構思與基礎構想

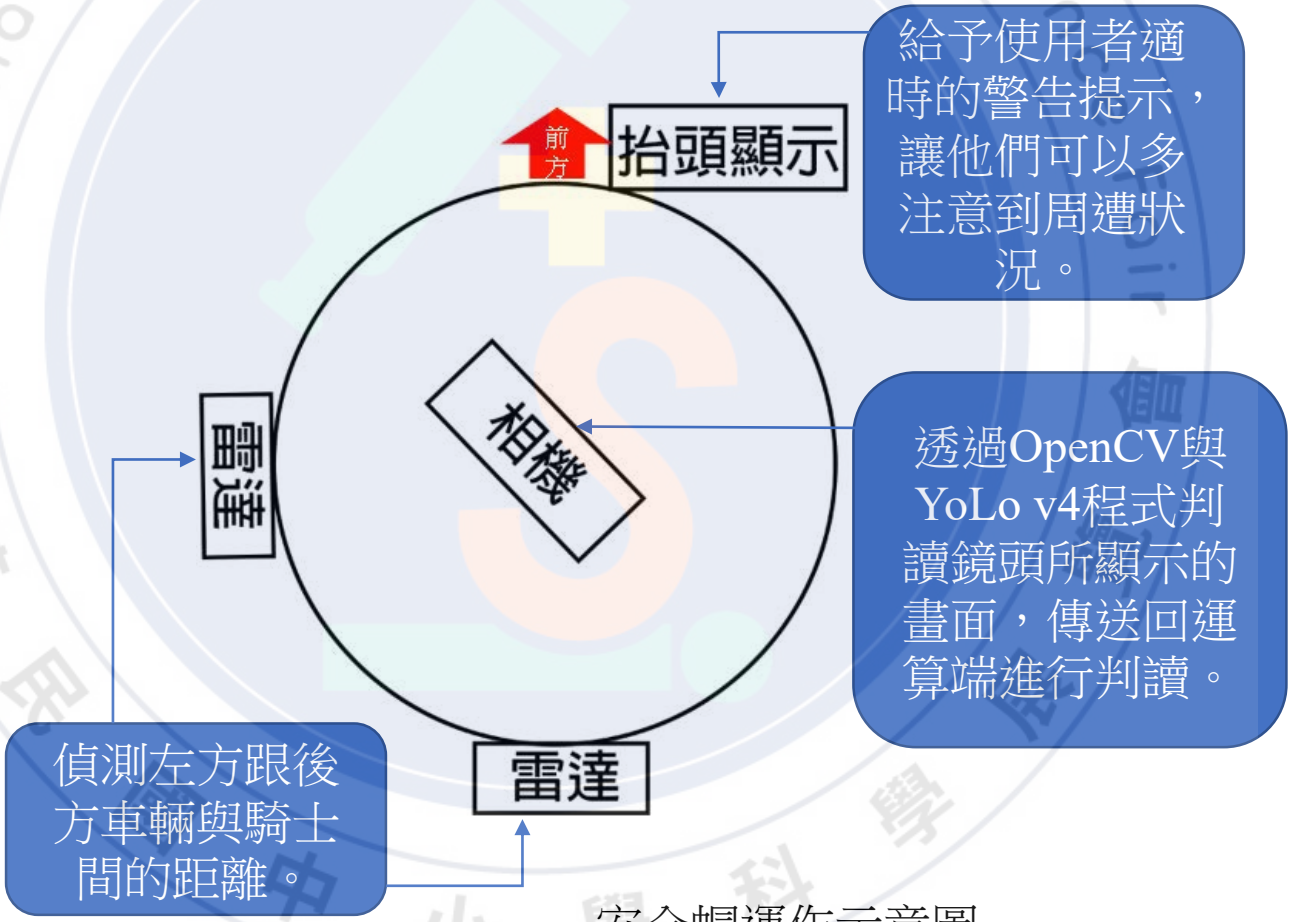
我們在經過討論之後，決定製作一個智慧型安全帽，透過物件辨識車輛種類與大小，再透過偵測車輛間距離的數據回傳到系統運算後告知使用者內輪差安全問題。

偵測距離

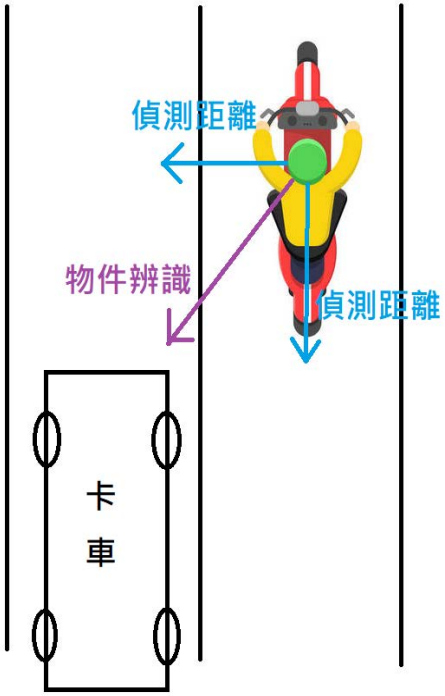
透過雷達偵測與車之間的距離給予使用者適當的提醒，以避免視線死角或未注意到的問題。

物件辨識

物件辨識可將卡車與一般的小型轎車分開標註，並且將標註的資料回傳給主控端進行運算。



安全帽運作示意圖



安全帽運作邏輯

研究方法：開發環境

1.裝置放置位置→安全帽

所有機車及腳踏車必備配戴器具為安全帽，故安裝在安全帽上以隨身攜帶、使用。

2.輸入影像設備→webcam

透過攝影機將後方來車影像讀入運算端。

3.影像演算→OpenCV+YOLO v4

利用OpenCV給予電腦視覺及判讀，並使用YOLO v4建立辨識資料庫進行比對。

4.溝通橋梁→python

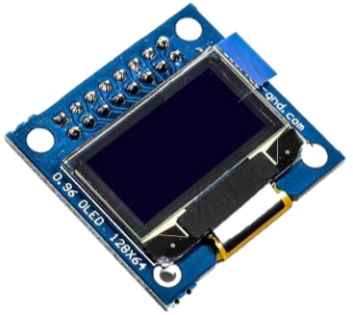
利用python程式將判讀結果的訊息給予Arduino。

5.距離測量→Tfmini雷達感測器

使用Tfmini雷達感測器測量數值，並傳回Arduino判斷是否落入內輪差。

6.警示訊息顯示→抬頭顯示器

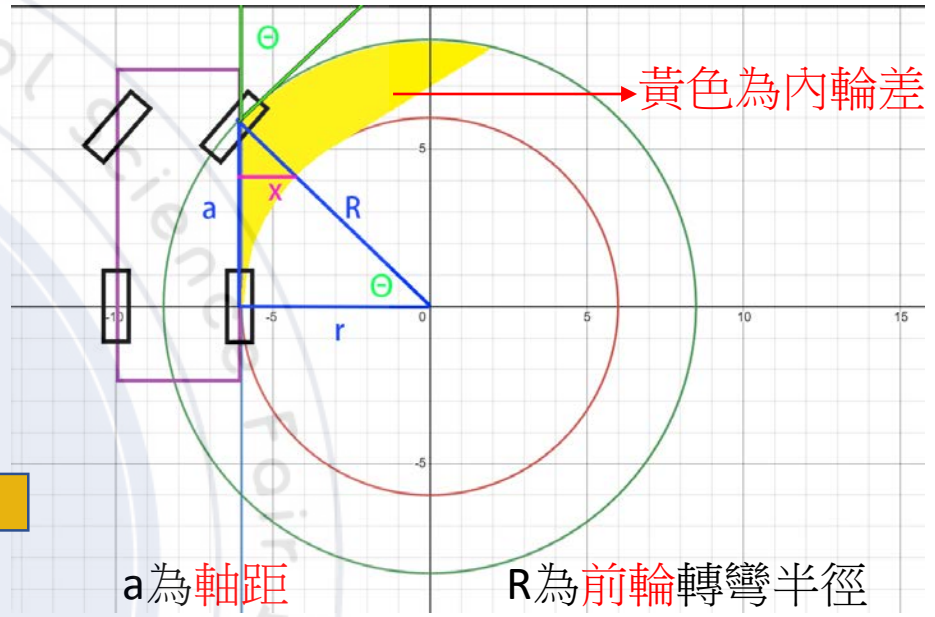
將警示標語利用0.66OLED投射在安全帽鏡片上，使騎士更容易注意。



研究方法：內輪差的研究

車輛在行進間，因前輪與後輪轉彎時的轉彎半徑有所不同，產生偏移所導致的**盲區**，對大型車來說當**轉彎角度越大所產生的內輪差範圍越大**。

我們也去查找並求證相關算式，最終得出兩種計算方法：



a為軸距　R為前輪轉彎半徑

Θ為轉彎角度　r為後輪轉彎半徑

利用三角函數計算轉彎半徑及內輪差 (單位：公尺)

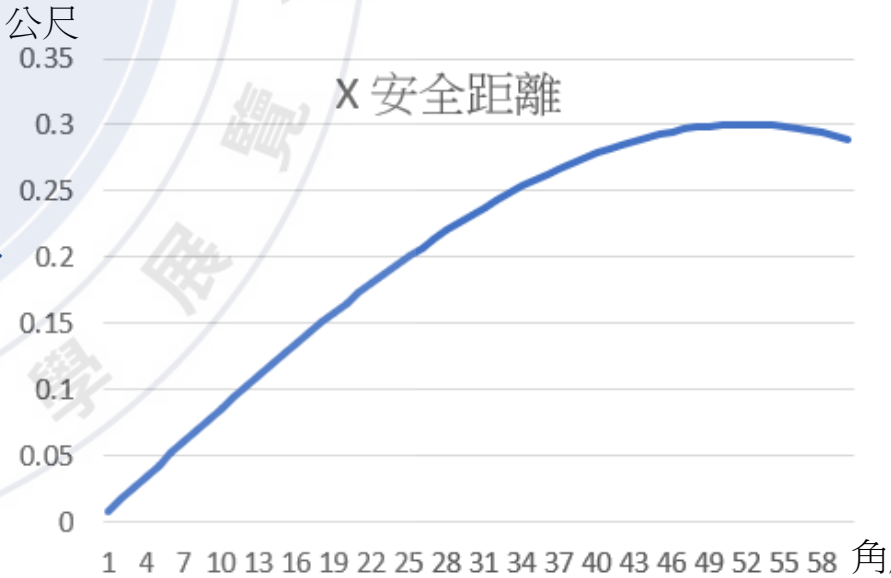
a 軸距	R 前輪轉彎半徑	r 後輪轉彎半徑	R-r內輪差	角度	X 安全距離
1	11.47371325	11.4300523	0.043660943	5	0.0434948
1	5.758770483	5.67128182	0.087488664	10	0.086159514
1	3.863703305	3.732050808	0.131652498	15	0.127166548
1	2.9238044	2.747477419	0.176326981	20	0.165693163
1	2.366201583	2.144506921	0.221694663	25	0.200923599
1	2	1.732050808	0.267949192	30	0.232050808
1	1.743446796	1.428148007	0.315298789	35	0.258277647
1	1.555723827	1.191753593	0.363970234	40	0.278817375
1	1.414213562	1	0.414213562	45	0.292893219
1	1.305407289	0.839099631	0.466307658	50	0.299736785
1	1.220774589	0.700207538	0.520567051	55	0.298584994
1	1.154700538	0.577350269	0.577350269	60	0.288675135

計算內輪差

1.$x=(R-r)*\cos\theta$

2.$x=(R-r)*r/R$

繪出折線圖



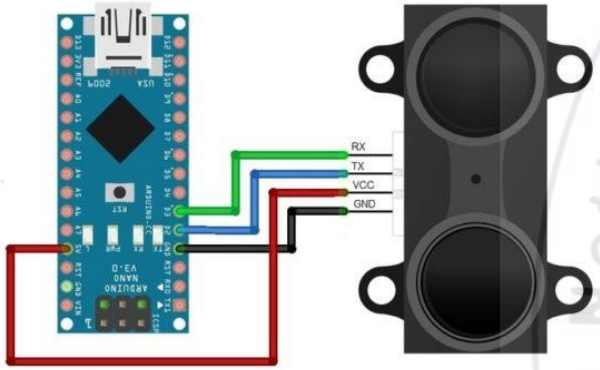
前輪轉彎半徑　$a/\sin\theta = R$　　後輪轉彎半徑　$a/\cos\theta = r$

透過計算得知應距離車輛至少達0.3倍的車長(軸距)

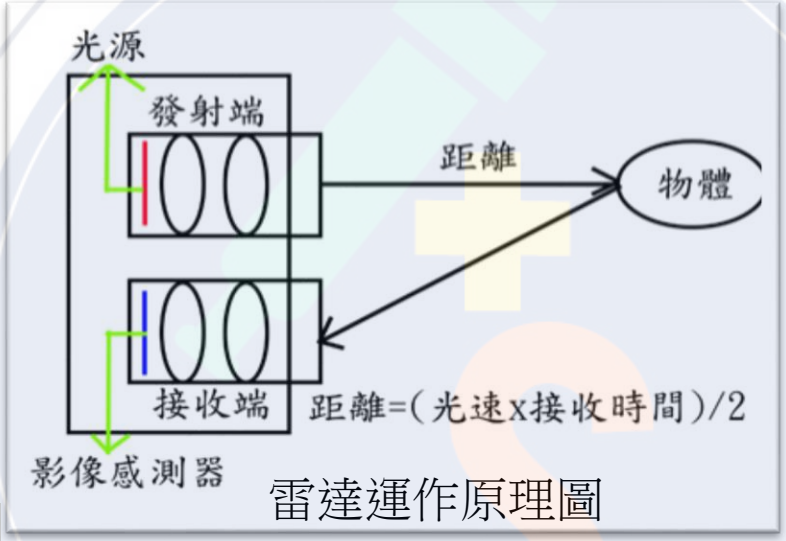
(一般來說以車長來計算較安全)

研究方法：雷達測距

透過雷達測距，根據光速為3 X 10^8 m/s的公式做計算，
我們可以得知與車輛間的距離為多少，再傳送給系統判斷。



雷達感測器接線圖

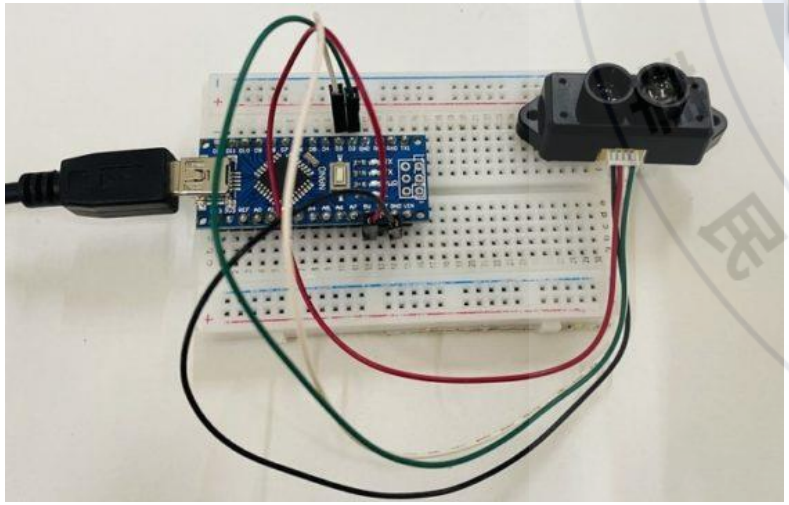


雷達運作原理圖

```
void loop(){
  r1=radi1();
  r2=radi2();
  if(digitalRead(2)==1){
    if(r1<=100){
      notice();
    }
  }else{
    if(r2<=100){
      warning();
    }else if(r2<=300){
      closed();
    }else{
      none();
    }
  }
}
```

```
15:00:30.407 -> 240cm/4cm
15:00:30.641 -> 241cm/4cm
15:00:30.827 -> 245cm/4cm
15:00:31.059 -> 244cm/4cm
15:00:31.248 -> 240cm/4cm
15:00:31.574 -> 2438cm/6cm
15:00:31.762 -> 245cm/7cm
15:00:31.996 -> 245cm/18cm
15:00:32.181 -> 244cm/57cm
15:00:32.694 -> 2479cm/2549cm
15:00:33.019 -> 246cm/2488cm
15:00:33.393 -> 246cm/2548cm
15:00:33.720 -> 245cm/2502cm
15:00:34.093 -> 244cm/2522cm
15:00:34.278 -> 239cm/46cm
```

雷達感測器偵測距離及回傳資料畫面



單顆雷達感測器實際接線圖

```
int radi1(){
  if (Serial.available()){
    if (Serial.read() == HEADER){
      uart[0] = HEADER;
      if (Serial.read() == HEADER){
        uart[1] = HEADER;uart2[1] = HEADER;
        for (i = 2; i < 9; i++){
          uart[i] = Serial.read();
        }
        check = uart[0] + uart[1] + uart[2] + uart[3] + uart[4] + uart[5] + uart[6] + uart[7];
        if (uart[8] == (check & 0xff)){
          dist = uart[2] + uart[3] * 256;     //calculate distance value
//          h = String(dist);
//          testscrolltext(h);
        }
      }
    }
  }
  return dist;
```

雷達感測器運算程式碼

研究方法：模型訓練、測試

資料集

COCO dataset
數量：6000+(大型車)
數量多、種類多樣

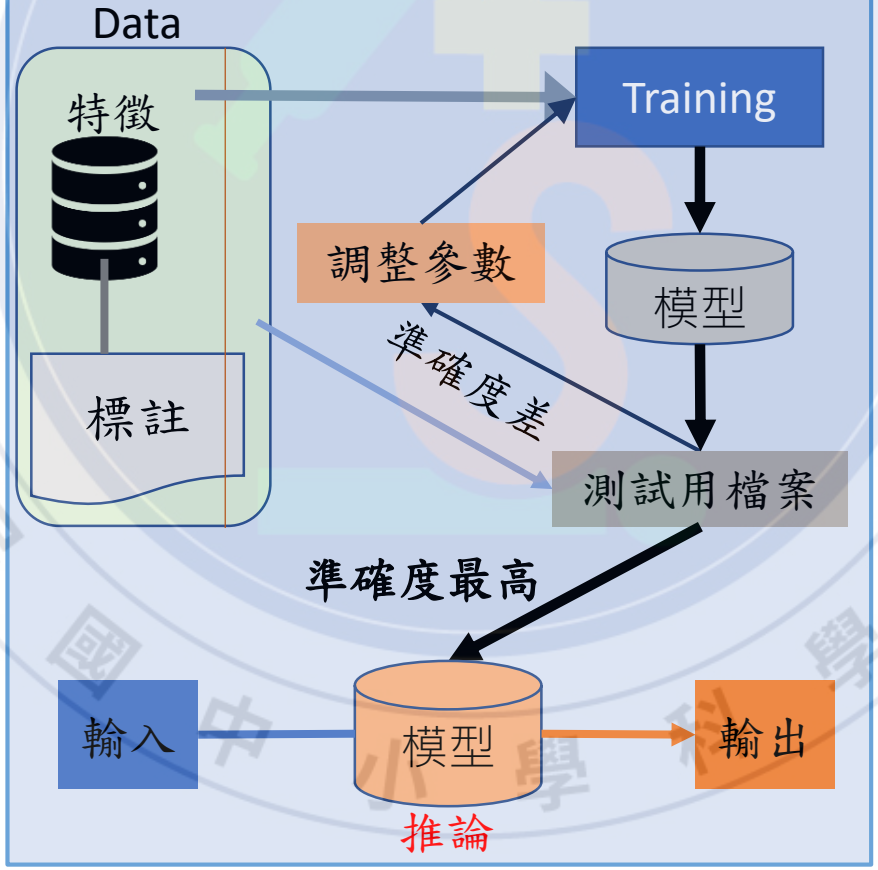
起初進行模型訓練時，採用約500張圖片作為資料集，發現到其數量不足準確度偏低，種類偏少導致無法辨識某些車型，後續改用網路上已有6377張已標註的資料集，能夠達到準確的辨識。

標註物件進行訓練

Flow Chart

監督式機器學習

將資料集送入，一部分進行訓練，另一部分為測試用檔案，透過多次調整找出最適合的模型，便可進行預測。



Test

輸出結果

研究方法： YOLOv4 即時物件偵測

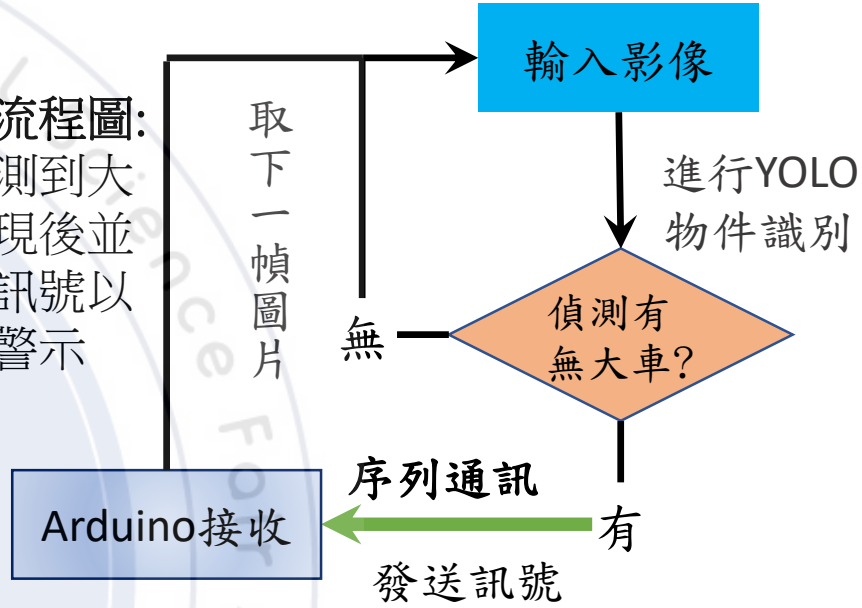
原圖(輸入影像)

經過YOLO模型偵測

偵測準確度

偵測時間(0.22s)

偵測流程圖:
當偵測到大車出現後並發送訊號以做出警示



為了能夠讓電腦去判斷路況是否危險，我們採用OpenCV和神經網路YOLOv4進行影像處理與機器學習，透過輸入**目標物的資料集**去訓練YOLOv4模型，它就可以幫我們在道路上達到**快速且精確的偵測**，此外還採用了序列通訊將偵測到的情況傳到Arduino。

```
configPath = "./cfg/yolov4-tiny.cfg"
weightPath = "./yolov4-tiny.weights"
metaPath = "./cfg/coco.data"
```

路徑輸入(配置檔案、權重檔案、已訓練資料)

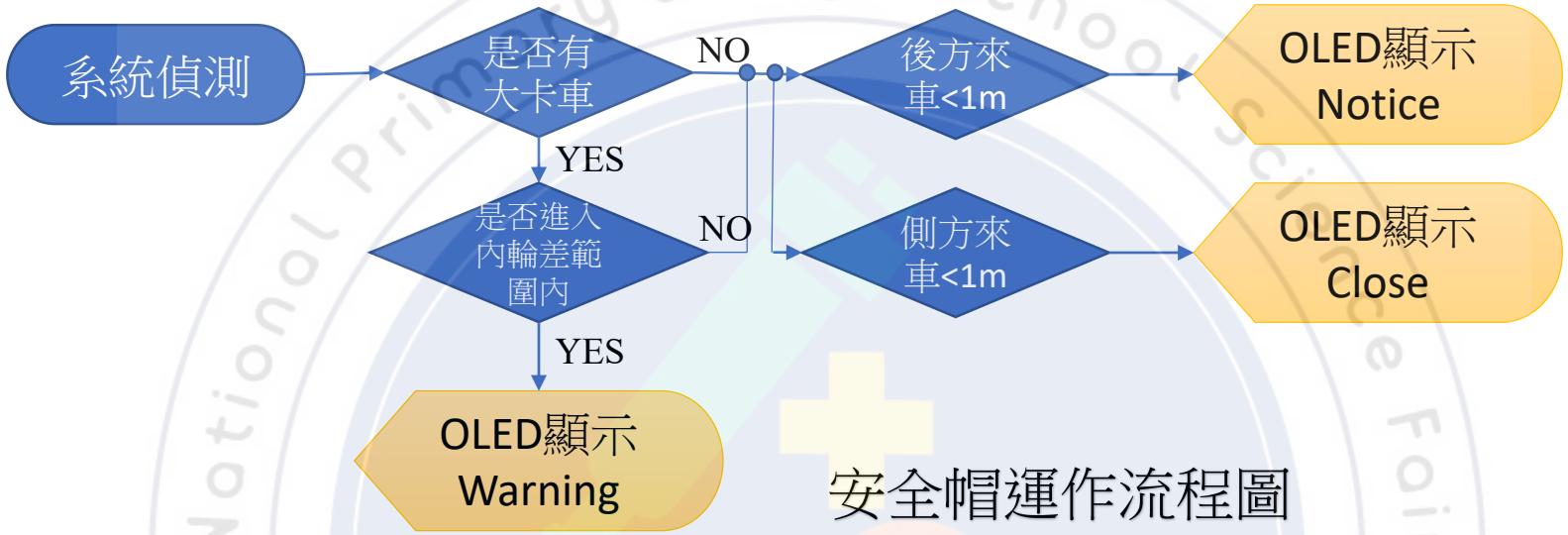
```
darknet.copy_image_from_bytes(darknet_image,frame_resized.tobytes())

detections = darknet.detect_image(network, class_names, darknet_image, thresh=0.25)
end=  time.time()
pretime=end -start
image = cvDrawBoxes(detections, frame_resized ,pretime)
```

以darknet神經框架控制YOLOv4神經網路進行物件識別

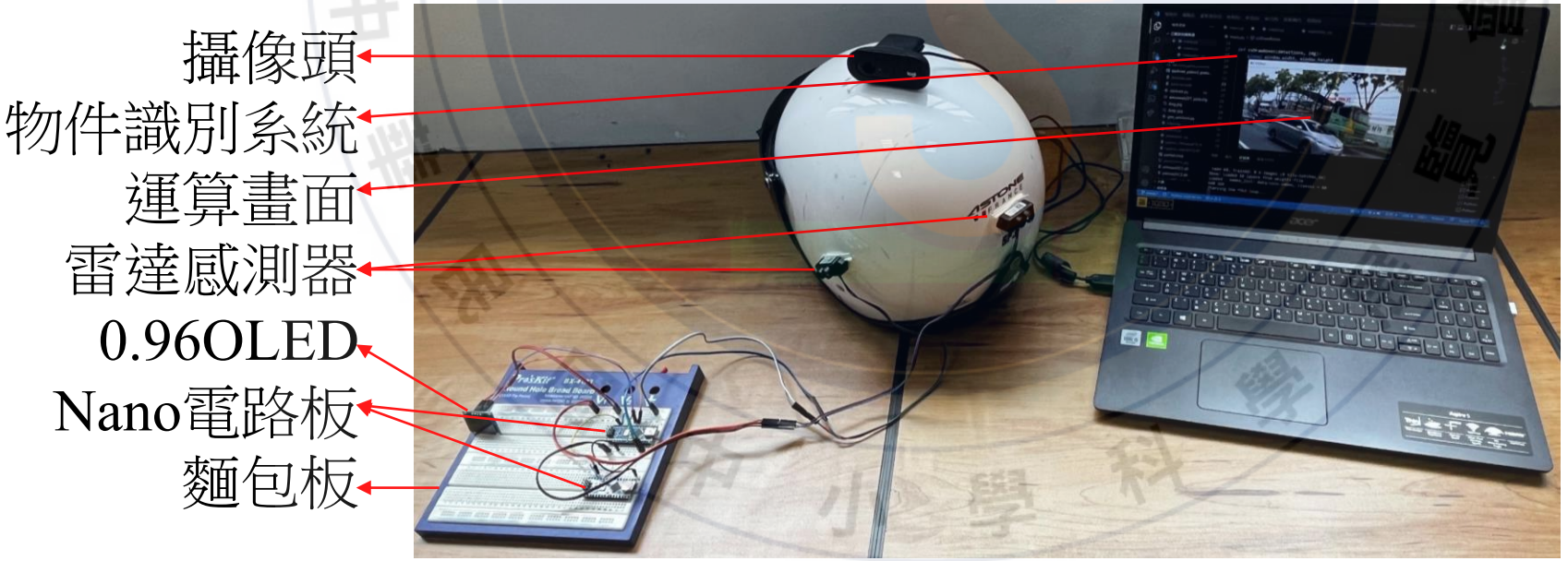
YOLO模型進行偵測，yolov4-tiny相較yolov4準確度稍微下降，但可提高偵測速度

研究方法：安全帽運作整合



安全帽運作流程圖

成品展現



研究方法：安全帽實測

透過系統運算，發送結果給抬頭顯示器來顯示警語，我可以透過偏光膜反射的警語來得知需要注意的行車狀況，增加駕駛人注意盲區的機會，減少意外的發生。

*左方為第一人稱視角右方為安裝位置

後方車輛(汽、機車)保持1公尺以上

周圍有車輛接近並小於1公尺以下

周圍有大型車輛並落入最大內輪差(0.3車長)內

結語

本研究以機車騎士為研究出發點，建置安全帽自動辨識系統提醒使用者路況。

以下為目前的研究成果：

- 可自動辨識車輛種類，如：小客車、中型車、公車、聯結車、砂石車等。(辨識時間為0.3秒以內)
- 可依照車種大小判斷須保持的最大安全距離。
- 以系統判斷可以即時給予使用者警語，讓使用者可以更加的注意周遭安全。
- 減少騎士在行進間不必要的分心。
- 解決機車騎士在行進間，無法注意盲區的問題。
- 透過模型訓練，可以提升判斷準確度，並且依照車輛的種類可以不斷的更新訓練。
- 本作品使用的系統都是開源的，若日後要做任何使用不會有版權問題。
- 透過Python編輯，日後要做功能新增或維修可以更加方便。

未來展望

- 針對系統做模組化處理，縮小硬體能執行我們所要的程式。
- 將外觀的雷達及抬頭顯示器縮小或隱藏到安全帽中，提升整體美觀。
- 設計續電裝置供應系統的運作，增加便利性。
- 縮短系統判斷縮需的時間及準確性。

參考資料

一、YOLO v4 模型訓練實作。I code so I am (2021)。
https://ithelp.ithome.com.tw/articles/10282549?sc=pt

二、如何以YOLO v3訓練自己的資料集－以小蕃茄為例。許哲豪 Jack(2019)。
https://makerpro.cc/2019/12/train-your-dataset-with-yolov3/

三、LabelImg 影像標註工具使用教學，製作深度學習用的資料集。G. T. Wang(2017)。
https://blog.gtwang.org/useful-tools/labelimg-graphical-image-annotation-tool-tutorial/

四、智慧影像，「One」視平安，大型車轉彎安全偵測警示系統。顏妤晅、陳以華、曹家瑜(2021)。
https://twsf.ntsec.gov.tw/activity/race-1/61/pdf/NPHSF2021-032808.pdf?0.8211693671307927

五、How to use TFMini-S LiDAR Distance Sensor with Arduino。Admin(2021)
https://how2electronics.com/how-to-use-tfmini-s-lidar-distance-sensor-with-arduino/