IA: una nova eina per ajudar els invidents

Treball de Recerca

Aya Talbi El Kouaihi

Jordi Ramon Domènech

2 Batxillerat A (2022-23)

Tutora: Anna Armengol

IES Puig de la Creu

12 de desembre de 2022

Castellar del Vallès

"La tecnologia és important, però allò que realment importa és què fem amb ella"

Muhammad Yunus, Premi Nobel de la Pau 2006.

Agraïm l'ajuda de tota la comunitat de GitHub i StackOverflow.

Abstract

Els invidents han de conviure dia a dia amb una gran quantitat d'obstacles en el seu camí, alguns difícilment detectables amb les eines d'ajuda tradicionals. La motivació d'aquest projecte ha sigut la falta d'innovació tecnològica en aquest sector malgrat les moltes possibilitats de progrés que hi existeixen. Un dels camps tecnològics que està tenint un creixement exponencial en els últims anys és el de la IA. Volem demostrar que amb el seu potencial podem fer coses tan curioses com la portada o projectes com aquest.

L'objectiu d'aquest treball de recerca és dissenyar un aparell basat en algoritmes de IA que faciliti la vida de les persones amb problemes de visió, sigui amb visió nul·la o aquelles amb baixa visió. Tant per millorar la seva mobilitat en espais exteriors (i en menor mesura, en tancats), com per permetre la realització de certes habilitats com la lectura.

Hem dividit la investigació en dues parts. La primera dedicada a informar i explicar el context en què es troba aquesta minoria i tots els conceptes teòrics pertinents, com ara les eines que tenim a la nostra disposició per tal de desenvolupar el projecte. L'altra consisteix en tot el procés de planificació i construcció del dispositiu, tant el hardware com el software.

Blind people lives every day with a great amount of obstacles in their way that they are not able to detect with the traditional help tools such as the white cane. The motivation for this project has been the lack of technological innovation in this sector despite the many possibilities for progress that exist. One of the technological fields that is having an exponential growth in recent years is AI. We want to show that with its potential it is possible to do things as curious as the cover or projects like this one.

The aim of this research work is to design a device based on AI algorithms that facilitates the lives of people with vision problems, either with zero vision or those with low vision. Both to improve their mobility in outdoor spaces (and in a minor part, in indoors), and to enable the performance of certain skills like reading.

We have divided the research into two parts. The first one dedicated to inform and explain all the relevant theoretical concepts, such as those we had at our disposal to develop the project. The other consists of the whole process of planning and building the device, both hardware and software.

Paraules clau: Visió per Computador, Llibreries, Python, Object Detection, Depth camera, Text-to-Speech.

Índex

1. Anatomia i Història de la ceguera: 1
1.1. La visió de l'ull humà 1
1.1.1. La còrnia 1
1.1.2. L'iris i la pupil·la 2
1.1.3. El cristal·lí 3
1.1.4. Humor vitri 4
1.1.5. Retina 4
3. Història i evolució de la ceguera 9
3.1. Com es tractava la ceguera antigament? 9
3.2. Evolució de les eines d'ajuda per a cecs: 13
4. Avaluació social de l'entorn de l'invident: 22
5. Està l'entorn adaptat als cecs? 28
5.1. Què entenem com a espai adaptat? 28
5.2. Història de l'adaptabilitat amb el pas del temps 28
5.3. Igualtat? 29
5.4. Exigències actuals 30
5.5. Ciutats adaptades 30
6. Precedents 32
6.1. Projectes anteriors relacionats 32
6.1.1. Eyesynth 33
6.1.2. Envision 35
6.1.3. Liberty Delta 37
6.2. Característiques similars 38
6.3. Anàlisi i captació d'idees 39
7. Fonaments teòrics 40
7.1. Com interpretar el món real a través de la informàtica? 40
7.1.1. Sensors infrarrojos 41
7. 1. 2 Càmeres 43
7.2. Gestió de dades 48
7.2.1. Processadors de 8 bits 50
7.2.2. Processadors x86/64 55
7.2.3. Processadors ARM 58
7.2.4 Elecció d'arquitectura i placa: 59

7.3. Firmware 62
7. 3.1. Llenguatges de programació 62
7.3.2. Tipus de llenguatges 62
7.3.3. Llibreries 64
8. Disseny del programa informàtic 65
8.1. Raspberry Pi 4 67
8.2. Detecció d'objectes 69
8.2.1. Elecció dels models 70
8.2.2. Instal·lació dependències YOLOv5 75
8.2.3. Personalització del model 76
8.2.4. Entrenament 84
8.2.5. Avaluació del model 86
8.3. IA Stairs Detection 96
8. 3. 1 Definició del model 96
8.3.2. Funcionament 96
8.3.3. Instal·lació prerequisits 99
8.3.4. Configuració de l'espai de treball. 101
8.3.5. Instal·lació del Stairs detection 102
8.3.6. Prova del model 104
8.3.7. Extensió per àudio 106
8.4. Text-to-Speech 107
8.4.1. Network architecture 108
8.4.2. Instal·lació de dependències 109
8. 4. 3 Funcionament del programa 110
8. 4. 4 Prova del model 113
9. Conclusions 115
Repositoris de GitHub 118
Webgrafia 119
Annexos 129
Annex 1. Yolov52.0 129
Annex 2. Programació Object Detection 132
Annex 3. Entrevista Anna Morancho 141
Annex 4. Model Stairs detection and Stairs modeling 148
Annex 5. Preparació Kinect 149
Annex 6. Programació Text to speech 150
Annex 7. Entorns de programació 152

Índex de figures

Figura 1. Imatge d'un Nautilus. 7
Figura 2. Malla del braille 14
Figura 3. Teclat Mantis Q40 amb lectura braille 20
Figura 4. Esquerra: esquema de les parts de guiatge Mini Cheetah 21
Figura 5. Dreta: Robot Mini Cheetah 21
Figura 6. Fotografia d'Anna Morancho, presidenta de l'associació B1+B2+B3 23
Figura 7. Ulleres Eyesynth. 33
Figura 8. Esbós de les ulleres Eysynth amb el sistema d'àudio coclear indicat 34
Figura 9. Ulleres Envision 35
Figura 10. Fotografia d'un dels creadors de les ulleres Liberty Delta utilitzant-les 38
Figura 11. Característiques del model d'Eyesynth. 39
Figura 12. Característiques del model d'Envisión 40
Figura 13. Característiques del model de Liberty Data 40
Figura 14. Esquema de l'espectre visible de la llum per l'ull humà 41
Figura 15. Esquema funcionament dels sensors infrarojos reflexius 42
Figura 16. Esquema funcionament d'un sensor infraroig de proximitat 42
Figura 17. Esquema funcionament de les parts d'una càmera 43
Figura 18. Passos realitzats per un sistema de visió artificial comú 44
Figura 19. Imatges resultants de la segmentació i processat d'una imatge a través de tècniques de visió artificial 44
Figura 20. Càmera PixyCam connectada a la seva placa corresponent. 45
Figura 21. Patró de punts que projecta la càmera Kinect. 46
Figura 22. Camp de visió horitzontal de la càmera Kinect. 47
Figura 23. Camp de visió vertical modificable de la Kinect. 47
Figura 24. Espai que ocupa un registre. Dins d'aquest registre es poden emmagatzemar fins a 64 bits 50
Figura 25. Dos exemplars d'Intel 8008. 51
Figura 26. Totes les plaques oficials Arduino que existeixen actualment 52
Figura 27. Esquerra: Processador AMD, Dreta: Processador Intel 57
Figura 28. Processadors RISC dissenyats per ARM i fabricats per diferents marques 59
Figura 29. Per ordre d'aparició: Classificació, Object Detection i Segmentació d'Imatge 69
Figura 30. Comparació entre els models més destacats d'Object Detection. 72
Figura 31. Comparació entre les tres últimes versions de YOLO (v5, v6, v7) 73

Figura 32. Procés d'anotació de les imatges 79
Figura 33. Modificacions que apliquem a les imatges del dataset. 81
Figura 34. Estructura que hauria de tenir el nostre directori amb YOLOv5 82
Figura 35. Els 5 models de YOLOv5 que hi ha actualment. 83
Figura 36. Comparació entre els models de YOLOv5 i EfficientDet 83
Figura 37. Missatges que ens mostra el programa mentre està entrenant. 86
Figura 38. Exemple d'imatge que dona com a sortida el nostre model 86
Figura 39. Exemple d'imatge que dona com a sortida el model. 87
Figura 40. Confusion matrix del model YOLOv5s entrenat 89
Figura 41. Formulà gràfica de com es calcula IoU 93
Figura 41. Gràfica Precision-Recall del model YOLOv5s1.0 93
Figura 42. Esquema de les mesures que pren el model de les escales 98
Figura 43. Imatge del transformador un cop obert. 103
Figura 44. Imatge del nou transformador. 103
Figura 45. Imatge del donant del component. 103
Figura 46. Procés de comprovació de la polaritat del transformador i la càmera. 104
Figura 47. Captura del programa detectant les escales de l'institut. 105
Figura 48. Imatge de la detecció d'escales que no compleixen estrictament la norma 105
Figura 49. Esquema del funcionament d'un sistema TtS. 108
Figura 50. Comparació amb la mateixa fotografia en model RGB i BGR. A l'esquerra el BGR, a la dreta RGB. 111
Figura 51. Text escrit a mà en una petita pissarra on es pot llegir "Hola, ¿Tienes huevos?". 113
Figura 52. Fotocòpia de text acadèmic de caràcter filosòfic, reflexió sobre la censura i l'expulsió dels poetes a La República de Plató. 113
Figura 53. Pàgina del llibre El Lazarillo de Tormes, ho detecta i ho llegeix. Confirmem la idea que només detecta text imprès. 114
Figura 53. Exemples d'imatges de "background", on no apareix cap categoria de les que hem de detectar. 130
Figura 54. Confusion Matrix del model Yolov5s2.0 130
Figura 55. Gràfica Precision-Recall del model Yolov5s2.0 131

1. Anatomia i Història de la ceguera:

Per tal de poder entendre totalment els inconvenients que suposa tenir afectacions en els ulls, primer hem de saber quin és el procés i les parts que formen l'ull, les quals ens permeten poder captar la llum que es projecta des de l'exterior del nostre cos i transformar-la en impulsos nerviosos que poden ser interpretats pel nostre cervell.

1.1. La visió de l'ull humà

La visió de l'ull humà és el resultat d'un complex procés de transformació i adaptació de la llum procedent de l'exterior del cos. Així, les diverses parts de l'ull han de dur a terme els processos adequats per complir l'objectiu final: fer que la llum pugui arribar a la retina, la qual transformarà la llum (en última instància, radiació electromagnètica) en impulsos nerviosos que després seran interpretats en el cervell per crear la imatge final que nosaltres veurem.

Primer de tot, donarem un cop d'ull a les diferents parts que componen l'ull humà. Tot i que moltes d'elles estan en coneixement de gran part de la població, altres són conegudes per menys gent i no per això deixen de tenir funcions importants.

1.1.1. La còrnia

La còrnia és la part més exterior de l'ull. De fet, és la part de l'ull encarregada de protegir d'agents exteriors els altres components de l'ull. A part de la seva funció de protecció, també és l'encarregada de deixar passar la llum de l'exterior a l'interior de l'ull. Per aquest motiu, la còrnia ha de tenir un color totalment transparent, ja que si no la llum no podria entrar a l'interior de l'ull. Això es traduiria en una pèrdua de la qualitat de visió.

Aquesta part de l'ull presenta una forma corbada, perquè, conjuntament amb el cristal·lí, efectua part de la correcció dels feixos de llum per tal que aquests quedin ben enfocats a la retina. Concretament, es calcula que la còrnia és la responsable de tres quarts de la potència focal total de l'ull. Si ens fixem en la seva extensió, podrem observar com aquesta no es continua al voltant de l'ull, sinó que es limita a la part frontal d'aquest, unint-se així amb l'escleròtica en una zona anomenada limbe.

És important saber que no disposa de vasos sanguinis, per tant, es nodreix exclusivament dels nutrients aportats per les llàgrimes i l'humor vitri que se situa just després d'ella. Tot i ser un teixit de només 550 micres, està formada per més d'una capa, les quals duen a terme diverses funcions.

1.1.2. L'iris i la pupil·la

Just després de la còrnia, trobem el teixit de l'iris. Aquest és el que conforma el cercle colorit que podem observar des de l'exterior de l'ull. La seva funció principal és la d'obrir-se o tancar-se per la seva zona central per tal de permetre l'accés a l'interior de l'ull a més o menys quantitat de llum. Aquesta part mòbil de l'iris, la qual varia la seva mida segons les condicions lumíniques de l'exterior, rep el nom de "pupil·la" i és la circumferència negra que veiem just al centre de l'Iris.

L'augment o disminució de la mida de la pupil·la es duu a terme de forma involuntària, mitjançant dos músculs diferents accionats per dos sistemes nerviosos diferents. D'aquesta forma, en dilatar la pupil·la s'acciona el múscul dilatador de la pupil·la, accionat pel Sistema Nerviós Simpàtic (S.N.S.)[1]. En contraure la pupil·la s'activa al múscul esfínter, el qual es contrau i redueix el diàmetre de la pupil·la, però al contrari que en la midriasi, aquest múscul és controlat pel Sistema Nerviós Parasimpàtic (S.N.P.)[2].

[1] Part del sistema nerviós encarregat de dur a terme involuntàries per posar en estat d'alerta al cos
[2] Part del sistema nerviós que duu a terme les accions involuntàries rutinàries.

1.1.3. El cristal·lí

Un cop la llum entra dins de l'ull, el primer que es trobarà serà el cristal·lí. Aquest és un teixit transparent que també té capacitat focal. Tal com hem vist anteriorment, la còrnia oferia gran part de la capacitat focal de l'ull, no obstant això, el cristal·lí és l'encarregat d'acabar formant de forma correcta les imatges sobre la retina. Això ho pot fer, entre altres coses, gràcies a la seva forma de lent biconvexa i a la seva capacitat elàstica.

La situació del cristal·lí dins de l'ull és diferent de la d'altres parts de l'ull. A diferència dels altres, el cristal·lí està suspès just darrere de l'iris. La seva fixació als teixits interiors de l'ull es duu a terme mitjançant les zònules de Zinn. Aquestes fibres són semitransparents i permeten unir el cristal·lí amb el cos ciliar.

El cos ciliar:

És una estructura circular que forma part d'una estructura més extensa anomenada úvea. D'entre les seves funcions trobem que és l'encarregat de formar l'humor aquós, líquid que conté la càmera anterior de l'ull (compresa entra la còrnia i el cristal·lí), aquest líquid conté gran quantitat de nutrients necessaris per alimentar la còrnia i el cristal·lí. Per altra banda, el cos ciliar també és l'encarregat de deformar i formar el cristal·lí mitjançant el múscul ciliar.

Com hem vist anteriorment una de les funcions del cristal·lí és la d'acabar d'enfocar de forma correcta les imatges a la retina. Si bé la gran majoria de la potència òptica la té la retina, el cristal·lí també té potència òptica, tot i que en menor quantitat. Per tant, la seva funció no és tant la d'enfocar amb potència tota la llum, sinó que serveix com a lent polivalent. La retina no varia la seva potència òptica en cap moment, ja que és una lent fixa. No obstant això, les imatges que provenen d'objectes llunyans i d'objectes propers no s'enfoquen d'igual manera a la retina. Per això és necessària una lent amb la capacitat de poder variar la seva potència òptica depenent de la

situació. Aquest procés rep el nom d'acomodació visual i és una de les funcions principals del cristal·lí.

1.1.4. Humor vitri

Un cop la llum ha passat a través del cristal·lí, abandona la càmera anterior de l'ull per entrar a la càmera posterior. Aquesta part de l'ull està composta gairebé en la seva majoria per l'humor vitri, un gel transparent el qual està compost en la seva majoria d'aigua. L'humor vitri manté el seu estat gelatinós gràcies a l'humor aquós produït pel cos ciliar.

1.1.5. Retina

Finalment, la llum arriba a la retina on, com hem explicat anteriorment, ha d'enfocar-se just a sobre de la retina per tal que es pugui enfocar la imatge correctament. És just en aquesta superfície on la llum és traduïda per la retina en senyals interpretables pel nostre cervell (impulsos nerviosos) les quals són conduïdes cap al cervell a través del nervi òptic. El procés de traducció de la llum és un procés complex que es duu a terme mitjançant les cèl·lules nervioses (neurones) fotosensibles, que s'ubiquen a la capa més exterior de la retina. El procés de transformació és complex, ja que inclou tant fenòmens químics com elèctrics. Tot i aquesta complexitat, podem identificar dos tipus de cèl·lules que fan la major part del procés . Aquestes cèl·lules es divideixen en dos grans tipus, segons la funció que exerceixen:

<u>Cons:</u>

Són les cèl·lules que s'encarreguen de transmetre la informació dels colors que conformen la imatge formada sobre la retina. A més a més, també ens permeten diferenciar colors entre ells. Per tal de poder identificar els diferents colors, calen bastons diferents capaços de reaccionar a diferents longituds d'ona de la llum. Així, trobem bàsicament tres tipus de bastons, repartits uniformement: Bastons sensibles a la llum vermella, bastons sensibles a la llum

blava i bastons sensibles a la llum verda. Les principals diferències entre ells són les proteïnes que contenen.

Bastons:

Si la funció dels cons era la de transmetre la informació sobre el color de les imatges en situació d'alta lluminositat, la funció dels bastons és la d'enviar informació en situacions de baixa lluminositat (visió escotòpica). La seva situació dins de la retina és molt més extensa que la dels cons, s'estenen per tota la retina. La proteïna que contenen per tal de poder captar aquesta baixa llum és la rodopsina, proteïna que reacciona amb la llum de longitud d'ona de 500 nm, corresponent a la llum verda-blava. A diferència dels cons no són capaços de proporcionar molta definició.

1.2. Evolució cap a l'ull humà

Un cop observat el funcionament de l'ull, cal veure com aquest va evolucionar cap a l'estructura que actualment té i com va anar canviant la seva estructura amb el pas del temps.

La vida comença al brou primitiu, en una Terra primitiva molt diferent de la que actualment coneixem. En aquest brou comença a desenvolupar-se la vida i amb ella els ingredients necessaris per poder fabricar el primer òrgan fotosensible.

Primer de tot es va forma la vitamina A, a conseqüència dels primers cloroplasts[3]. Per altra banda, també va començar a desenvolupar-se una altra proteïna involucrada en el primer ull primitiu: l'opsina. Aquesta va ser creada en l'interior d'alguns bacteris procariotes primitius i, més endavant, va unir-se amb el retinal per formar la rodopsina, un compost fotoreceptiu. Aquest compost va conviure amb altres compostos de la mateixa funció els quals es teoritza que serien els primers capaços de transformar la llum en senyals interpretables per un ésser viu. No obstant això, aquests altres

[3] Els primers éssers a poder transformar energia solar en química.

compostos podrien arribar a ser tòxics pels bacteris primitius en certes circumstàncies i, a causa d'això, molts éssers van optar per la rodopsina, que no resultava tòxica. El principal component de la visió ja estava creat. Només era qüestió de temps per tal que l'adoptessin els primers animals.

Els primers éssers en utilitzar un mètode semblant a la detecció de la llum van ser les esponges. Animals presents a la terra des de fa 700 MA[4]. Aquests animals, tot i no poder percebre la posició de la llum, van adoptar la rodopsina com a mètode de detecció de llum. Concretament, amb aquests compostos tan primitius només eren capaces de determinar si hi havia llum o no. La llum els hi era essencial per a poder reproduir-se i fer funcions metabòliques. Aviat la majoria d'éssers primitius van adoptar la rodopsina com a receptor lumínics i es va produir un dels primers passos evolutius més importants en l'ull: Els petits sensors individuals van evolucionar per formar petites taques de receptors units entre si anomenades taques oculars o ocels. Això sense cap millora remarcable per a ells, ja que les taques oculars seguien tenint les mateixes capacitats lumíniques que els petits sensors individuals

Aquest sistema, tot i haver quedat obsolet per les seves evolucions, encara es pot observar en certs éssers vius de l'actualitat. Un dels que avui en dia encara utilitza el sistema de les taques oculars és l'euglena[5], un ser flagel·lat de color verd el qual incorpora una taca ocular just al costat del flagel.

Tot i afirmar que els Ocels van ser els primers ulls, cal tenir en compte que aquests tenien una forma molt diferent de la que avui en dia podríem anomenar ull. Es calcula que el pas d'aquestes taques oculars a un ull amb forma similar a l'actual van ser necessàries 364 000 generacions d'éssers vius variats. I per poder arribar a una estructura i forma semblant a l'actual van ser necessaris 500.000 anys.

[4] Tot i que la data és objecte de discussió entre els entesos.

[5] Editors de Viquipèdia (consultat 25 agost 2022). Viquipèdia l'enciclopèdia lliure. https://ca.wikipedia.org/wiki/Euglena

El següent pas que va desenvolupar l'ull va ser el de formar una espècie de copa al voltant de les taques oculars i l'enfonsament de l'ocel. Això millorava substancialment la detecció del que hi havia al davant, ja que l'enfonsament dels sensors òptics permetia posicionar millor l'origen de la font de llum i detectar molt millor els canvis de llum que provocava el moviment d'alguna cosa davant seu. A mesura que la copa es feia més i més profunda, les capacitats visuals de l'animal augmentaven.

Una de les millores més destacables que va assolir l'ull primitiu va ser passar de formar una copa a ser una esfera, amb un petit forat al final com a conseqüència del tancament de l'anterior copa. Aquest canvi va permetre poder començar a distingir formes i començar a resoldre les primeres imatges de l'exterior. El procediment que seguia la llum en aquesta etapa de l'ull era exactament la mateixa que com ho feia amb les càmeres estenopeiques[6]. Cal destacar que, tot i que l'ull va anar evolucionant al llarg del temps respecte d'aquesta forma, encara podem trobar animals que fan ús d'aquest sistema. Un gran exemple seria el famós Nautilus, mol·lusc que encara actualment habita el fons marí

Figura 1. Imatge d'un Nautilus.
Font: https://www.oceanoculus.com/news-from-the-sea/chambered-nautilus-nautilus-pompilius

El principal problema que plantejava l'ull aleshores era la impossibilitat de poder-se usar a l'exterior del medi aquàtic i la poca protecció que aquest mateix tenia sobretot de cara a paràsits que poguessin fer d'aquella cavitat el seu lloc de residència. Així, la següent evolució notable que va patir l'ull va ser la creació d'una capa transparent

[6] Càmera fotogràfica sense lent ni sistema òptic.

que el cobria per l'exterior tot creant així una primera versió de la còrnia. Així, s'aconseguia un doble efecte: En primer lloc, aconseguien una protecció extra contra invasions parasitàries i, en segon lloc, plantejava una opció inèdita de poder portar la visió a l'exterior del medi aquàtic. Gràcies al tancament de l'ull es va poder desenvolupar més estructures a l'interior d'aquest creant així un humor transparent que ocupava tota la cavitat interior de l'ull. Gràcies a aquest humor, els ulls podien tenir funcionalitats extres, com per exemple el bloqueig de raigs ultraviolats (nocius) i van poder aconseguir un major índex de difracció de llum, amb la qual cosa podien aconseguir millors enfocaments a la retina primitiva.

Finalment, l'últim gran pas evolutiu que va patir l'ull va ser la incorporació de lents. Es creu que la formació de les lents va sorgir a causa de la creació de diverses capes dins de la còrnia. Entres aquestes capes, circularia líquid amb la funció d'aportar els nutrients necessaris a aquestes dues capes i també serviria com a mètode immunològic per a l'ull, tot això sense que les capes hagin de perdre la seva transparència. De forma involuntària, aquest líquid exerciria pressió en contra de les dues capes entre les quals estava comprès i faria que es formessin plecs amb una certa curvatura. Aquests plecs acabarien modificant la trajectòria de la llum i, per tant, com incidiria aquesta a la retina. Amb el pas del temps aquestes lents improvisades abandonarien la còrnia i esdevindrien una estructura pròpia dins del mateix ull. El principal avantatge que es va obtenir amb aquesta millora era la capacitat de poder enfocar de forma correcta les imatges i, en alguns casos, poder tenir molt més angle de visió. D'aquesta forma, també es desenvoluparia un sistema per regular la quantitat de llum que entrava dins de l'ull, permetent així poder detectar preses o perills en molts escenaris amb diferents nivells d'il·luminació.

Com podem apreciar, l'ull ha patit una llarga evolució fins a la forma amb la qual actualment el coneixem, tot i que els anys necessaris per a poder arribar a aquesta forma han sigut bastants, s'ha de tenir en compte que en el context de l'evolució en són relativament pocs. A més, cal destacar que la majoria de canvis es van dur a terme en el període de l'explosió Cambriana, on la supervivència va passar a tractar-

se de veure qui aconseguia adaptar-se millor al medi i qui tenia més capacitat per sobreviure. En aquest context, tenir una bona visió del que passava al teu voltant era un fet imprescindible per a poder sobreviure, d'aquí la ràpida evolució de l'ull en aquest transcurs del temps.

3. Història i evolució de la ceguera

3.1. Com es tractava la ceguera antigament?

La concepció que tenim avui en dia de la ceguera o la baixa visió és, com és molt diferent de la que diverses societats han tingut i han aplicat, arribant a considerar als cecs com persones "inútils" i incapaces de poder sobreviure per si mateixes. Per tal de poder conèixer i valorar el punt en el qual es troben els cecs avui en dia, primer cal saber d'on provenen i quins avenços s'han produït des de l'inici de la humanitat fins als nostres temps.

La ceguera comença de la mateixa manera que comença el mateix ésser humà, amb la prehistòria. Durant aquest període, la vida era difícil. L'ésser humà havia de subsistir en condicions de vida molt baixes i també havia d'aprendre a valer-se a un mateix. Les migracions a altres indrets on viure millor eren freqüents i, un cop ja ens vam fer sedentaris, tothom tenia una tasca a fer, la qual era indispensable per la subsistència del grup. A més, les condicions de perill constants en la que la gent vivia propiciaven que, tot i que hi hagués gent que no fos cega de naixement, perdessin la vista a conseqüència d'una batalla pel menjar o per una simple infecció. En aquestes condicions tan dures, la tribu només contemplava el cec com a una molèstia, una persona que no podia exercir les seves tasques i que tampoc podia aportar res a la comunitat en la qual vivia. És per això que els cecs eren eliminats de la tribu, mitjançant el seu abandonament (en el cas de la gent que perdia la vista) o bé mitjançant l'assassinat ja des de ben petits, si es concloïa que l'infant no tenia

capacitat plena per a poder observar i poder exercir la seva funció. En molts casos, l'assassinat es barrejava amb rituals religiosos.

Durant l'edat antiga, les seves condicions de vida no van millorar gaire. Tot i la millora general de la vida dels ciutadans de societats com la grega o la Romana, els cecs encara no eren considerats persones. En la majoria dels casos, també es practicava l'infanticidi, és a dir que el nadó (un cop acabat de néixer) era presentat davant d'un tribunal, on se'l jutjava sobre les seves capacitats sensorials. En cas que el jurat considerés que no posseïa totes les capacitats sensorials, se sentenciava el nadó a mort. Les metodologies utilitzades per treure'ls-hi la vida eren molt variades: Des d'abandonar-los a la seva sort dins de gerres de fang fins a tirar-los daltabaix d'un penya-segat. Tot i que matar a infants no era legal (es castigava amb la mort), es feien excepcions i una d'elles era el cas de néixer amb una discapacitat.

La influència de la religió tampoc va millorar gaire la qualitat de vida dels cecs. En un principi, la religió concebia als cecs com a persones que no havien obeït els senyals de Déu i que aquest els castigava per això. En molts casos també s'atribuïa la ceguera com a un càstig hereditari, és a dir, que si tenies ceguera tenies algun parent o familiar (encara que fos llunyà) que no havia obeït a Déu.

Deixant de banda els motius religiosos, ens traslladem als segles XVII i XVIII, enmig de la Il·lustració. En temps de grans pensadors, la ceguera passa a ser objecte de debat entre aquests. Principalment, discuteixen sobre com pot el cec interpretar el món i si aquest interpretaria el món igual si tingués vista. Entre aquests pensadors, trobem el filòsof John Locke el qual planteja en el seu llibre *Assaig sobre l'enteniment humà* (1689) que si una persona nasqués cega i de sobte recuperés la visió, aquesta no seria capaç de poder utilitzar els ulls com a font d'informació. Així, deixava clar la seva postura: Els cecs no poden aconseguir tants coneixements com les persones que no pateixen ceguera. Per altra banda, també teníem el pensador George Berkeley, el qual planteja que la visió realment no ens ajuda a distingir els objectes, sinó que només ens permet descobrir el cos d'aquests, buit de significat. Els objectes

no es veuen sinó que es perceben. Per tant, obria la possibilitat que els cecs tinguessin la mateixa capacitat envers el coneixement de l'entorn que les persones videntsⁱ[7].

De la mateixa manera que els cecs entraven dins dels debats dels filòsofs, també es començava a pensar en la seva educació. Fins ara l'educació dels cecs no s'havia plantejat seriosament. Tots podien aprendre coses, però sense arribar al nivell el qual la gent no cega podia arribar a assolir. El debat sobre si la visió era necessària per poder aprendre estava servit. Una de les primeres persones que va posar en pràctica aquest debat va ser el matemàtic Nicholas Sauderson, el qual va arribar a entrar a les altes esferes de la ciència essent cec des de naixement.

Finalment, el debat arriba a una conclusió amb Diderot. El filòsof francès va plantejar que la solució al problema podria venir per altres sentits de l'ésser humà: El tacte. El francès va fixar-se en l'exemple de Sauderson i va argumentar la importància de l'experiència sensorial en l'ésser humà. També va plantejar que l'única capacitat que no podia ser substituïda per una altra en termes d'educació era la capacitat de raonar i d'entendre les coses.

Capa final del segle XVIII, s'inaugura a França la primera escola enfocada a ensenyar a cecs, basant-se en metodologies molt diferents de les escoles convencionals. El religiós Valentin Haüy, es basa per obrir la seva escola en un alfabet tàctil desenvolupat per la pianista Austríaca Maria Theresia, pianista cega. Així, crea un nou sistema de llenguatge per a cecs que s'implantarà en la societat durant els pròxims cinquanta anys, inspirant així a moltes altres persones a obrir també la seva pròpia escola per a cecs.

El model d'educació cap als cecs variarà cap a dues grans direccions: La primera, enfocada només a ensenyar a aquests a ser autosuficients econòmicament i, per tant, a ensenyar treballs que els cecs poguessin fer. Aquest és el model que es plantejava

[7]Berkeley, George (1709). *An Essay Towards a New Theory of Vision.*

en moltes de les successives escoles per a cecs que van anar obrint a l'illa anglesa. Per altra banda, el segon model es basava en una educació teòrica d'aquestes persones. Aquest model creia en les capacitats dels cecs en poder assolir grans nivells de coneixement de la mateixa forma que ho podria fer una persona normal. Finalment, destacaria la figura de Laura Bridgman, la qual va demostrar als E.U.A. que els cecs podien arribar a saber i aprendre al mateix nivell que els altres. Bridgman va entrar en Howe's school, on va demostrar a tothom que podia aprendre igual que els altres i que un cec era capaç també de poder escriure i llegir.

La següent gran revolució en el camp de l'educació i la ceguera seria protagonitzada per Louis Braille, el qual va inventar durant la dècada de 1820 un sistema de comunicació per a cecs basat en petits cercles impresos en relleu que permetien, mitjançant el tacte, escriure caràcters mitjançant un ordre i uns patrons que el mateix braille va desenvolupar. Aquest sistema va ser molt ben rebut per la comunitat cega, ja que a més de poder aprendre a través d'aquest llenguatge, també podien ensenyar a iguals. La magnitud d'aquesta nova creació va ser tan gran que ha arribat fins als nostres dies, on el braille és un dels principals sistemes de lectura per a cecs.

Les primeres associacions de gent cega no arribarien fins al segle XX, concretament als Estats Units, on a principis d'aquest segle es van agrupar les primeres associacions professionals de cecs. La primera que es va formar va ser l'American Association of Workers for the Blind (AAWB per les seves sigles en anglès) fundada l'any 1905. Els principals objectius d'aquestes associacions no era altra que el de pressionar la política i la societat per tal que tinguessin una ment més oberta enfront dels cecs. Més endavant (cap al 1930) van aparèixer els primers activistes a favor de la del respecte a les persones cegues i del reconeixement dels problemes dels cecs per l'administració pública. El seu únic propòsit: donar a conèixer les necessitats que la ceguera els imposava per tal que les administracions poguessin actuar al respecte. Finalment, aquestes organitzacions regionals van anar organitzant-se i ajuntant-se a escala nacional fins a crear la National Federation of the Blind (NFB), organització encara present avui en dia.

3.2. Evolució de les eines d'ajuda per a cecs:

Un cop hem entès el context en el qual els cecs van haver de viure, creiem convenient repassar la història dels objectes que han anat utilitzant al llarg dels anys per tal de poder ajudar-se a desenvolupar en un món en el qual no poden veure. Aquests, a més, són els principals precedents del sistema de detecció que pretenem plantejar al final d'aquest treball. Serà llavors interessant veure com han anat evolucionant les ajudes al llarg de la història, sobretot amb l'entrada de la tecnologia, principal aliat també en aquesta recerca.

3.2.1. Braille

Una de les primeres eines que van inventar-se per poder suplir la falta de visió va ser el braille. Inventat pel ja citat Louis Braille (Coupvray, França 1809- París, França 1852), cec total des dels 5 anys, va idear una nova forma d'interpretar i llegir una llengua mitjançant altres sentits que no fossin la vista. El seu sistema de lectoescriptura es basa en el tacte com a mètode per poder captar la informació.
Per desenvolupar-lo va basar-se en un mètode desenvolupat per l'exèrcit francès que utilitzaven els soldats per parlar entre ells sense ser escoltats per l'enemic. No obstant això, aquest sistema (de punts i ratlles en relleu) era fonètic i no seguia les normes estàndard d'ortografia. Així, Braille va crear un nou sistema basat només en punts en relleu i que, a part de poder-se llegir, també era fàcil d'escriure per a un cec. Va tardar uns cinc anys a dur-ho a terme i va considerar que ja estava prou desenvolupat quan ell tenia només quinze anys.

Tot i que en un principi el sistema no va ser gaire ben rebut per la comunitat educativa, poca a poc va anar convertint-se en el mètode més usat en l'àmbit mundial. Avui en dia, el braille és el sistema de lectoescriptura més popular del món. Cada llengua adapta les seves lletres i expressions a aquest sistema i, per tant, hi ha tants idiomes de braille com idiomes al món.

El seu mètode de funcionament és senzill, es basa en una xarxa de sis punts distribuïts en dues columnes i tres files en les quals cada punt té un ordre i estan numerats:

1 4

2 5

3 6

Figura 2. Malla del braille

F*ont: www.once.es/servicios-sociales/braille*

La combinació del relleu individual d'aquests punts ofereix fins a 64 combinacions diferents, on cadascuna representa un signe diferent. En cada llegua els signes canvien i també s'utilitzen algunes combinacions per caràcters especials de cada una. Per exemple, en el català les vocals accentuades tenen una combinació diferent de la de la vocal sola.

3.2.2. Bastó guia

El segon invent que va suposar un gran augment de la qualitat de vida dels cecs va ser el bastó guiador. Si bé és un objecte senzill, aquest aporta una gran autonomia a les persones que el porten, ja que els hi permet reconèixer obstacles al seu voltant. Així, no han de dependre d'altres persones que les guiïn i poden anar amb més seguretat pel carrer.

La seva història és bastant confusa, perquè el bastó com a eina per a cecs s'ha fet servir durant molts anys. El bastó pintat de color blanc va sorgir a Anglaterra a principis del segle XX, enmig d'un canvi en els mètodes de transports, on els carros van deixar pas als cotxes. D'aquesta forma, els cecs anaven insegurs en intentar travessar els carrers, ja que la velocitat dels cotxes podria fer que el conductor no veiés el bastó i es pensés que la persona que estava creuant sabia de la direcció del cotxe i, per tant, s'apartés. Així, James Biggs una de les persones cegues a Anglaterra que patia per la seva seguretat, va decidir pintar el seu bastó de color blanc. Aquest color destacaria

d'entre els colors de l'entorn i farien ressaltar el bastó enmig del bullici del trànsit. D'aquesta forma s'asseguraven que els conductors d'automòbils estiguessin assabentats de la ceguera de la persona que creuava.

Amb un argument més o menys igual, el bastó blanc va travessar el canal de la Mànega i va instaurar-se a París. Allà, Madame Guilly d'Herbemont es preocupava pel mateix fet que l'anglès Biggs, però en aquest cas ho feia pels nens que veia travessant el carrer en una escola per a gent cega. En el cas francès, el bastó ressaltava per la seva similitud amb els bastons, també blancs, utilitzats per la policia.

Mentrestant, a E.U.A. el bastó blanc també s'introduïa al país, però aquest cop en mans del president del club Lleons[8] de Peoria, Georges A. Bonham. Aquest va recaptar fons el 1930 per a poder comprar bastons blancs per a tots els cecs de la seva comunitat.

Una evolució en el disseny del bastó blanc es faria just després de la Segona Guerra Mundial, on als Estats Units es desenvolupa una iniciativa per poder ajudar a tots aquells veterans de guerra que havien perdut la vista durant el conflicte. Allà van estudiar com els podria ajudar la implantació d'un bastó i en van desenvolupar un model amb mides i dissenys adaptats per les necessitats reals dels cecs (fixant-se més en la funcionalitat que en la visibilitat del bastó). Aquest model és el que avui en dia encara s'utilitza.

Hi ha diverses tècniques per a poder fer servir el bastó, depenent de la situació en la qual un es trobi. Depenent del tipus de terreny, situació (espais interiors o exteriors) i de si hi ha escales o no, la persona ha d'utilitzar un dels mètodes establerts per poder caminar de forma segura. No és el mateix caminar per un carrer en un espai obert que enmig d'un centre comercial on hi haurà molts més obstacles i de menor dimensió. De la mateixa manera que tampoc és el mateix caminar per un terreny

[8] Els clubs lleons són una organització mundial dedicada a satisfer les necessitats de les comunitats, tant en l'àmbit local com internacional. Per a més informació: www.lionsclubs.org/es

asfaltat o encimentat que per un terreny muntanyós i ple de sotracs. Distingim els següents mètodes:

- Mètode de Hoover:

  Utilitzat per espais oberts i urbanitzats (ciutats, pobles). Moviments amples per a poder detectar tots els obstacles i, a la vegada, protegir-se un mateix d'aquests.

- Mètode de lliscament:

  S'utilitza en espais tancats. Moviments menys oberts. En contacte la major part del temps amb la paret. Pot combinar-se amb el mètode Hoover.

- Mètode de tocs:

  Es fa servir en terrenys irregulars. Consisteix a anar donant petits tocs a terra per comprovar la seva irregularitat

- Mètode per pujar i baixar escales:

  El bastó es converteix en una eina de mesura, la qual mesurarà amb la punta l'altura i l'amplada de l'esglaó a superar. Així la persona pot fer una idea de com és aquest.

Cal distingir també entre els diferents colors dels bastons. Si bé tots fan una mateixa funció, ens adverteixen de la problemàtica que la persona té:

- Bastó blanc:

  Com ja hem indicat, vol dir que la persona que el porta és cega
- Bastó blanc-i-vermell (ratlles blanques i vermelles):

  És el bastó escollit per la Federació Mundial de Persones Sordcegues. Ens indiquen que la persona que el porta no només és cega sinó que també és sorda.

- Bastó verd:

  Ens indica que la persona té un percentatge visió entre el 10-30%. Aquestes persones no són cegues, sinó que tenen baixa visió.

3.2.3. El gos guia

Si hi ha una eina que realment ajuda a les persones cegues per tal que puguin ser autònomes és el gos guia. Aquest animal no només té una funció pràctica (conduir a l'invident en situacions no favorables) sinó que acaba desenvolupant un vincle d'afecció entre amo i gos. A diferència d'altres eines, com ho pot ser el bastó blanc, el gos guia no només és capaç de guiar a la persona, sinó que també és capaç d'assistir-lo i escollir per a ell el recorregut més apte. En casos de situacions amb molts obstacles, el gos sap trobar la forma més fàcil per poder-ne sortir. Fins i tot poden guiar els seus amos a seients buits en transports públics.

La relació entre l'home i el gos es remunta a milers d'anys enrere i apareixen casos concrets on es pot deduir que un gos ajudava a una persona cega. A part d'alguns casos excepcionals, els gossos guia es considera que apareixen just després de la Primera Guerra Mundial. Després d'un dels majors conflictes a escala global, alguns dels soldats que van aconseguir sobreviure a ella van quedar sense visió com a conseqüència dels atacs amb gasos tòxics que es van realitzar durant la guerra. El doctor Gerhard Stalling, metge alemany en un dels hospitals on es tractaven ferits de guerra, va tenir la idea d'entrenar gossos per tal que fessin de guies a aquestes persones. Va anar investigant i al final va acabar obrint la seva escola de gossos guia el 1916 a la ciutat d'Oldenburg. Des d'aleshores, les escoles es van anar expandint fins a estar per tota Europa.

A Amèrica arribarien per part de Morris Frank, el qual va llegir un article que va escriure una estudiant americana en un dels centres d'entrenament de gossos de més rendiment d'Alemanya. Aquest va contactar amb ella i van viatjar fins a Suïssa, on

van aprendre a entrenar i educar a un gos. Al seu retorn als Estats Units, Frank va portar el primer gos guia del país. Més endavant l'estudiant (Dorothy Harrison Eustis) va obrir la primera escola de gossos al continent americà.

Com podem imaginar no qualsevol gos pot exercir com a gos guia. Cal tenir en compte que aquests gossos han de ser animals molt sociables i també han de tenir un molt bon comportament. La majoria de gossos tenen, en major o menor mesura, certs instints d'agressivitat que els gossos guies han de suprimir al màxim (mitjançant l'entrenament). A més, un gos guia també ha de saber aguantar situacions adverses per a molts gossos, sorolls forts, grans estímuls per la distracció, etc. En definitiva, un gos guia ha de ser un gos amb una capacitat de concentració excel·lent, tenir molta voluntat de treball i tenir un caràcter molt dòcil i noble.

No totes les races de gossos són aptes per aquesta labor. Com sabem, hi ha races que tendeixen a ser més agressives, d'altres que tendeixen a ser molt instintives i altres que no són sociables. D'aquesta forma, els requisits que han de complir ens deixen amb una selecció bastant acurada de les millors races per fer aquesta feina.

Tradicionalment, els gossos guia són Labradors Retrievers. Aquesta raça té molt bones qualitats, com per exemple la seva gran sociabilitat i la versatilitat. Si bé aquesta és la principal raça, també es poden formar gossos guia a partir d'altres races com per exemple Golden Retriever, Pastor alemany i el Flat-coated retriever.

3.2.4. Tiflotecnologia

A finals del segle XX s'obre un nou món per la humanitat. Apareix internet i es comença a interactuar amb unes capses quadrades les quals anomenem ordinadors. Amb el gran creixement que va tenir la tecnologia i la gran acceptació que ha continuat tenint durant el segle XXI és normal que tothom volgués estar present en aquesta revolució. No obstant això, les persones cegues tenien un gran problema en relació amb la tecnologia digital. Si bé fins ara el coneixement i la informació es transmetia

només a través de paper (el qual es podia transcriure al Braille), amb l'arribada de l'era digital les coses canvien per complet. La informació ja no es transmet a través del paper sinó amb una pantalla i mitjançant l'ús d'un teclat i ratolí. D'aquesta manera tota persona que no pogués interactuar amb la pantalla quedava automàticament exclosa d'aquest món. Amb aquest context neix la tiflotecnologia, una disciplina de la tecnologia que té com a objectiu aportar els medis necessaris perquè la gent amb problemes visuals pugui ser autònoma en l'ús de la tecnologia. A mesura que avança la tiflotecnologia també trobem una especialització sobre com pot la tecnologia ajudar i millorar la qualitat de vida d'una persona.

L'objectiu principal de la tiflotecnologia és fer accessible la tecnologia als cecs. Per tant, el seu primer objectiu va ser el d'adaptar tots els avenços en el camp de la ceguera a la informàtica. Eines com un teclat amb Braille o simples programes d'amplificació dels continguts mostrats per pantalla van ser uns dels primers passos que la disciplina va fer.

Avui en dia, podem trobar evolucions d'aquests teclats Braille, molt més moderns i amb moltes més capacitats en comparació amb les primeres unitats. Aquests aparells es basen en el mateix mecanisme que el braille escrit, es conformen d'unes 40 cel·les amb tres files i dues columnes de petits pilars que pugen o baixen segons els que s'està mostrant en pantalla. Així, la persona que l'utilitza pot anar llegint en Braille tots els textos que trobi a l'ordinador. Pel que fa a l'escritura, no és realment necessari que se serigrafiïn les lletres en Braille per poder saber quina lletra estan prement en aquell moment. Tots els teclats incorporen dues ratlles en relleu sobre les lletres "f" i "j". Aquests relleus serveixen per posicionar els dits en l'art de la mecanografia[9]. Així, es poden aprofitar aquestes tècniques d'escriptura per memoritzar la disposició de les lletres i poder escriure.

[9] Segons diccionari.cat, "Art d'escriure a màquina" 08/10/2022

Figura 3. Teclat Mantis Q40 amb lectura braille
font: www.pocklington-trust.org.uk/technology/tech-news-and-views/mantis-q40-braille-review/

Altres eines que s'han desenvolupat per poder fer més accessible la tecnologia són la síntesi de veu que llegeix el text en pantalla i impressores en Braille, entre d'altres.

Per altra banda, mirarem també la segona branca de la tiflotecnologia, utilitzar la tecnologia per millorar les condicions de vida de la gent cega:

Els primers exemples que molta gent coneix són els audiollibres. Aquests, tot i no ser un desenvolupament únic per les persones cegues, els permet llegir títols en molt menys temps que no pas si fossin llibres escrits en braille (molt més grans que els normals i més lent de llegir). Actualment, podem trobar multitud de plataformes on poder escoltar aquests llibres, tot i que per les persones cegues poden accedir alguns d'aquests llibres de forma gratuïta, gràcies a la biblioteca d'audiollibres que l'associació ONCE té. Altres biblioteques també s'utilitzen, però sota una subscripció mensual que pot variar segons la plataforma.

La tecnologia també pot ajudar a les persones invidents a poder desplaçar-se de forma autònoma, mitjançant la invenció d'objectes que complementin l'ús del bastó blanc. Un dels exemples que val la pena comentar és el bastó intel·ligent que han desenvolupat a la Universitat de Stanford (Califòrnia). Aquest bastó incorpora una sèrie de sensors capaços de generar un mapa en tres dimensions de l'entorn en què es troba la persona. Mitjançant algoritmes d'intel·ligència artificial, el bastó és capaç de determinar el millor camí a seguir. Es comunica amb el portador mitjançant la roda que toca a terra. Depenent del camí a seguir, exerceix més o menys força cap a un dels costats d'aquesta.

La tiflotecnologia no només crea dispositius tecnològics per a les persones amb dèficits de visió, sinó que també adapta tecnologia existent per poder ajudar a aquestes persones. Un exemple d'això és el robot en forma de gos anomenat mini cheetah, Aquest robot desenvolupat al Massachusetts Institute of Technology (MIT). La funció principal és la de poder caminar sobre qualsevol tipus de terreny sense desestabilitzar-se i oferir, a més una capacitat d'equilibri sense precedents. En cas que perdés l'equilibri, també és capaç d'aixecar-se tot sol. Gràcies a totes aquestes característiques, és un candidat excel·lent per servir com a guia a persones cegues. Mitjançant càmeres, sensors i algoritmes, també és capaç de predir el millor camí a seguir i guiar, mitjançant una lleu tirada de la corda amb la qual s'uneix al portador, a la persona sense desestabilitzar-la o fer-la caure.

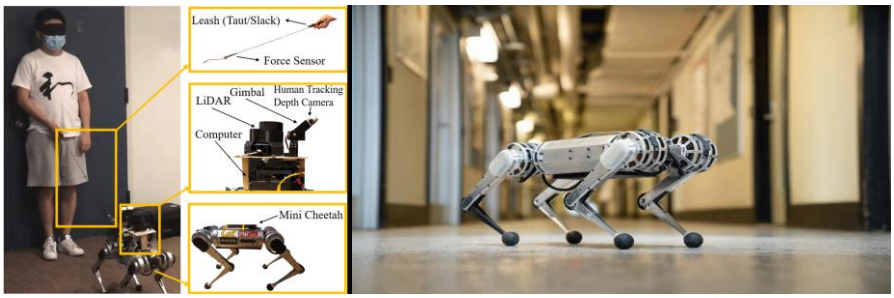


Figura 4. Esquerra: esquema de les parts de guiatge Mini Cheetah

font: https://www.researchgate.net/figure/The-Mini-Cheetah-is-guiding-a-blindfolded-person-to-avoid-obstacles-with-leash-guided_fig1_355428864

Figura 5. Dreta: Robot Mini Cheetah

font: https://news.mit.edu/2019/mit-mini-cheetah-first-four-legged-robot-to-backflip-0304

Per acabar amb la tiflotecnologia, explicar també que existeixen multitud de projectes destinats a millorar la vida dels cecs. Per exemple, s'està comercialitzant unes ulleres amb sensors la qual és capaç de distingir i dir les coses que es troben davant de la persona. També és capaç de llegir textos que no estiguin escrits en braille i reproduir-los auditivament. Aquestes ulleres s'anomenen OrCam i les veurem més endavant en l'escrit.

4. Avaluació social de l'entorn de l'invident:

Quan una persona pateix qualsevol mena de discapacitat, no només ha de patir les conseqüències de viure d'una forma diferent de la normal, sinó que també ha d'adaptar-se a una nova vida social. En un món on la majoria de la gent només té al cap una sola manera de viure i interactuar amb la gent, és difícil poder adaptar-se completament tenint una forma de viure molt diferent de la que la majoria tenen. Tot això porta al que socialment anomenem "desigualtats" i "exclusió social". Com és evident, ser una persona invident no n'és una excepció. D'aquesta forma, en aquest apartat posarem el focus sobre quins problemes socials i personals provoca la discapacitat a aquelles persones que, per diversos motius, no poden veure o bé tenen baixa visió.

Per poder fer més fàcil la vida d'aquestes persones actualment tenim arreu del territori associacions i fundacions destinades a intentar millorar la vida de totes aquestes persones. Moltes d'elles són conegudes nacionalment, com pot ser el cas de l'ONCE (Organización Nacional de Ciegos Españoles), però també existeixen d'altres associacions que si bé no tenen tant abast com podrien ser les nacionals, tenen una gran influència tant en l'àmbit local com autonòmic. D'aquesta forma, hem volgut posar-nos en contacte amb una d'aquestes associacions per tal que ens expliquin de primera mà quins són les principals dificultats socials amb les quals es troben les persones invidents.

L'associació que ens ha permès fer-ho ha estat l'Associació Discapacitat Visual Catalunya B1+B2+B3, situada al Carrer Cardenal Reig, 32 del districte de les Corts de la ciutat de Barcelona. En aquest cas, ens han concedit una entrevista amb la seva presidenta, Anna Morancho Lluelles. Per tant, parlem una mica més de l'associació i de com ajuden a totes les persones que en busquen[10]:

[10] Podeu llegir la transcripció de l'entrevista en l'Annex 3

Figura 6. Fotografia d'Anna Morancho, presidenta de l'associació B1+B2+B3
font: https://es.linkedin.com/in/anna-morancho-lluelles-1b06b162?original_referer=

Segons ens ha explicat la presidenta, l'associació porta ja vint-i-vuit anys d'història al seu darrere ajudant a les persones cegues a portar una vida més amena dins de la seva discapacitat. L'associació neix, per tant, l'any 1994 com la resposta dels seus membre fundadors al fet que la gent cega total en els seus temps d'oci no disposaven de llocs accessibles o anar a passar-s'ho bé.

"Les persones que van fundar l'entitat van veure que la gent cega total es quedaven a casa en el seu temps d'oci (...) a partir d'aquests fets, es va començar a crear aquesta entitat per muntar activitats accessibles per a aquestes persones."

L'associació va començar en el seu inici com una activitat de voluntariat i a mesura que el temps passava va anar professionalitzant-se. Durant tot aquest transcurs de temps, l'entitat també va anar incorporant activitats i serveis a oferir fins a arribar als que avui en dia tenen. Avui en dia, l'entitat pretén tenir un abast a escala de Catalunya, però admeten que intenten fer el que poden amb el finançament que tenen:

"*Intentem atendre totes les peticions que ens fan, si la petició es fora de l'àrea de Barcelona i el personal s'ha de moure es mou, però tot això té un cost per a nosaltres molt elevat."*

El finançament per poder dur a terme totes aquestes activitats socials el treuen per una part dels fons dels òrgans de govern que es destinen anualment a aquest tipus d'administracions. A més també reben finançament per part de col·laboradors,

persones que aporten diners de forma continuada, de les quotes de soci i també intenten recaptar diners d'empreses o altres entitats que vulguin col·laborar amb ells tot oferint-los tallers i xerrades.

Al marge de l'organització de l'associació, la presidenta ens explica els principals problemes que les persones amb discapacitat visual tenen en el seu dia a dia. S'afanya a aclarir que les necessitats i els problemes són individuals de cadascú, és a dir que depenent del tipus de malaltia que pateixi una persona i del grau de ceguera que tingui, li resultarà més o més o menys senzill fer certes activitats:

"Hi ha un ventall molt gran de problemes de visió (...) nosaltres hem fet la separació del B1, B2, B3 del nom de l'associació en l'àmbit social per poder distingir quines necessitats tenen cada nivell (...) les persones amb visió B3, tenen entre el 100% i el 50% de l'agudesa visual[11] i no els afecta els problemes de visió al dia a dia. És a partir del B2, quan considerem que una persona ja comença a tenir problemes de visió. Aquest tenen entre el 50 i el 10 % d'agudesa visual i els considerem persones amb baixa visió, sobretot del 30 al 10%. Finalment, tenim els B1 que són aquelles persones que són legalment declarades com a cegues, tot i que algunes poden veure una mica. Cada grup i persona té les seves necessitats"

Tot seguit, passem a parlar sobre els principals problemes que més o menys totes les persones amb dificultats de visió tenen. Els principals impediments que es troben són impediments arquitectònics, és a dir que formen part o bé de l'arquitectura de la ciutat o bé del mobiliari del carrer, que algunes persones posen amb tota la seva bona voluntat, però que no s'adonen que realment estan molestant a la gent amb problemes de visió. Posa com a exemple la gent que sol posar testos amb plantes a la vorera:

"Podem arribar a detectar el test, però el que és la part alta de l'arbust o planta no i si és molt gran doncs ja te l'has menjat amb la cara"

[11] L'agudesa visual és la qualitat de visió que una persona té.

Hi ha altres elements que també dificulten la vida a aquestes persones, com són les bastides de les obres i els carrers que comencen a fer ara que s'anomenen de "plataforma única". En la vida quotidiana també depèn del grau de visió que una persona tingui. En general la presidenta afirma que les persones amb dificultat de visió se solen valdre per si soles i solen ser bastant autònomes. No obstant això, també reconeix que hi ha certes tasques com pot ser anar a fer la compra que persones amb ceguera total no poden fer. És en aquests casos en què l'associació posa a la seva disposició a voluntaris per poder ajudar-lo a fer aquest tipus de coses. En acabar conclou:

"El món no està fet per tenir un problema de visió, la veritat…"

Seguidament, passem a parlar sobre el món laboral en les persones cegues. En aquest camp afirma que hi ha molt de respecte envers els cecs i que poca gent s'atreveix a contractar-los:

"Hi ha molts dubtes de les capacitats de les persones amb problemes de visió"

També creu que la gent no n'és conscient que si una persona ha sigut capaç de treure's el títol en una especialitat, serà capaç de desenvolupar la seva funció sigui cega o no. En general les persones amb problemes de visió no solen anar a buscar feines que saben que no poden fer. La presidenta posa com a exemple que un cec mai anirà a una entrevista per ser conductor de tren o d'autobús, sinó que aplicaran a aquelles feines que realment són capaces de fer. Tampoc entén la negativa de molts empresaris en no contractar a persones cegues mentre posen l'excusa que no hi ha cap lloc en l'empresa on pugui fer bé la seva feina:

"*Jo mateixa he estudiat empresarials i sé que dins de l'empresa hi ha moltes posicions. Jo m'he dedicat més al tema de recursos humans, però ara sabem que*

tot està molt digitalitzat i en principi les empreses poden fer més accessibles aquests documents amb un ordinador adaptat (...). Encara queda molta feina per fer en els dos mons, tant en el tema de l'accessibilitat com en el de l'empresa. Les capacitats les tenim, només necessitem que ens proporcionin les eines apropiades."

No es descuida de mencionar que l'atur en la gent discapacitada, sobretot discapacitades visualment, tenen un atur molt superior al que les persones sense discapacitats tenen. Posa com a referència que pot haver-hi tranquil·lament un atur del 75% de les persones cegues.

Seguidament, comencem a parlar sobre la tecnologia i de com pot ajudar aquesta a les persones cegues. Destaca que la tecnologia ha estat crucial per millorar la qualitat de vida d'aquestes persones. També comenta que s'han desenvolupat moltes eines específiques per solucionar una mica més la vida de les persones amb problemes de visió, però que hi ha altres aplicacions que no tenien aquest objectiu i que han acabat esdevenint aplicacions indispensables per totes aquestes persones:

"*Hi ha moltes aplicacions que ens ajuden en el nostre dia a dia i inclús n'hi ha que no estan fetes per a nosaltres i que ens ajuden molt a ser més autònoms*"

Tot i això, també ens comenta que la gran majoria de gent que pateix discapacitat visual té una certa edat i que moltes vegades aquesta gent no està tan predisposada a aprendre a fer servir i utilitzar aquestes aplicacions.

Per acabar, ens parla sobre els diferents serveis que ofereix l'associació. En primer lloc, parla del primer servei que es va crear, el de servei cultural. Com bé explica al principi la finalitat d'aquesta associació era en un principi treure de casa les persones invidents que no tenien res a fer en el seu temps lliure. A més a més, afegeix que avui en dia aquest servei ha passat a ser més que res una excusa per entrar en contacte entre persones que tenen problemes de visió:

“En un principi es va crear [el servei de cultura] perquè no hi havia absolutament res accessible per a persones cegues. Avui en dia, ja hi ha més espais accessibles, però se segueixen fent, ja que dins de l’entitat s’estan creant grups (...). Hi ha gent que sap que amb nosaltres pot fer altres activitats sense haver de dependre de la família. Això els fa més autònoms”

El servei que més gent utilitza segons la presidenta és el servei de l’assistenta social. És el primer servei que reben les persones quan entren a l’associació. És la primera persona que entra en contacte amb el discapacitat i la que valora quines coses els hi poden oferir per millorar la seva qualitat de vida.

L'assistenta social deriva a la persona capa a altres departaments, entre ells destaca el departament de psicologia. Aquí, quan una persona necessita atenció psicològica, normalment degut a un empitjorament de la malaltia o bé perquè ha perdut la vista, rep l’atenció necessària.

Un dels altres departaments que també és molt important és el servei de rehabilitació bàsica. Aquí, els socis aprenen totes les tècniques necessàries per poder desenvolupar-se de forma autònoma amb una malaltia visual. Aquí seria on aprendrien a llegir el braille o bé a utilitzar el bastó blanc.

Entre altres serveis que ofereixen trobem el servei d’integració laboral, on ajuden als socis a trobar llocs de feina i el servei de tiflotecnologia, on ensenyen eines per fer servir els ordinadors amb eines fetes especialment per a persones amb dificultats de visió. Tot i no haver més servies, la presidenta apunta que estan oberts a incorporar-ne més i que si en algun moment algun soci proposa crear un altre servei, ells ho faran sense dubtar-ho.

Dins de les tasques de l’associació també trobem la de promocionar i finançar estudis en l'àmbit de la visió i els problemes relacionats. De la mateixa forma que també duen a terme campanyes al tercer món on ajuden a persones amb dificultats de visió:

"El primer motiu de ceguera el món és les cataractes, que aquí es poden operar en cinc minuts i al tercer món… tots sabem que coses que aquí tenim com a bàsiques allà no tenen per què ser-ho. Intentem donar un cop de mà com podem."

5. Està l'entorn adaptat als cecs?

5.1. Què entenem com a espai adaptat?

El món està dissenyat per a la gent que no pateix cap mena de discapacitat física. Això fa que, petites coses que per a nosaltres no son cap inconvenient siguin un gran obstacle per aquells amb una minusvalidesa.

Per poder combatre aquesta desigualtat a l'hora de desplaçar-se i interactuar amb allò que els envolta s'han creat els espais adaptats. Entenem com a espai adaptat aquella àrea, instal·lació o servei que reuneix totes les condicions i seguretat per poder ser utilitzat per persones amb discapacitats. Però aquests intents d'inclusió a favor dels discapacitats són una novetat.

5.2. Història de l'adaptabilitat amb el pas del temps

La inclinació cap a l'esperit de la marginació és part de la natura humana. Des que la humanitat va començar a viure en comunitat es va cercar qualsevol excusa, siguin les condicions socials i/o econòmiques, la ideologia, l'ètnia o les preferències dels individus per tal de crear una suposada classificació entre aquells "normals" (és a dir, que encaixen dintre d'allò que una majoria considera com a comú) i aquells "anormals", els que es desvien d'aquestes concepcions.

La pauta comú de la discriminació cap als discapacitats va ser una constant durant pràcticament tota la història. Les iniciatives públiques per tal d'impulsar la inclusió d'aquest grup no van començar a aparèixer fins el segle XX.

No va ser fins a l'any 1982 que les Nacions Unides van aprovar el primer Programa d'Acció Mundial per les Persones amb Discapacitat[12]. En l'article 21 d'aquest es declara que les mesures adoptades fins llavors no són suficients si es vol assolir la igualtat entre persones. Si no que s'ha de poder facilitar al màxim l'accés al treball, la vida pública i familiar, l'educació, els espais d'oci, la seguretat social, l'habitatge, les relacions i les instal·lacions públiques. És a partir d'aquí que es va impulsar la idea dels espais adaptats.

5.3. Igualtat?

En aquest punt volem parar un moment per fer una petita correcció general. Si bé s'ha estat repetint la idea d'igualtat a través de tot aquest treball i és aquest el concepte que s'acostuma a utilitzar quan es vol parlar sobre la integració d'una minoria en el context social. Però considerem aquest un error. Si ho analitzem, la definició[13] d'igualtat (en aquest cas, social) la declara com "la característica d'aquells estats en els quals tots els seus individus o ciutadans sense exclusió, aconsegueixen a la pràctica tots els drets humans, fonamentalment els drets civils, polítics, econòmics, socials i culturals necessaris per arribar a la veritable justícia social". Així, s'entén que es pretén arribar a aquella anomenada justícia aconseguint que tots els individus arribin a tots els requisits mencionats. Però aquí es troba la problemàtica. El terme "igualtat" apel·la a què tots els individus obtinguin el mateix a través dels mateixos mitjans. Un altre cop, es cau en una concepció: si aquestes ajudes funcionen per a un, funcionen per a tots. Òbviament, això només funcionaria si tothom partís des de les mateixes condicions, però no és així.

És per aquest motiu que des del nostre punt de vista, no s'ha de parlar d'aplicar igualtat, sinó equitat. Adaptant-nos a les necessitats de cada individu sí que seria possible arribar a l'anhelada justícia social.

[12] ONU, A. G. (1983). Programa de acción mundial para las personas con discapacidad. *Obtenido de Aprobado por Resolución*, *37*, 52.
[13] Definició d'acord amb la RAE.

5.4. Exigències actuals

Fins a l'actualitat, les associacions de minusvàlids i l'OMS[14] han centrat les seves exigències i peticions en la millora i augment de tots aquests mecanismes en a favor de la integració i facilitació a l'accés al món en general. Com a resposta a això, la tecnologia ha posat el punt de mira en les ciutats, intentant adaptar aquestes amb tots els recursos dels quals es disposen avui dia. Són les anomenades *smart cities* o ciutats adaptades.

5.5. Ciutats adaptades

Enfocar la majoria d'esforços en l'entorn global no té cap incògnita, ja que set de cada deu persones amb discapacitat viuen en entorns urbans, segons Accedes, empresa dedicada a crear aquest tipus d'entorns adaptats. La mateixa ens dóna una definició bastant encertada de *smart city*; aquella ciutat que permet que tota la ciutadania, tingui o no discapacitat, pugui viure, desplaçar-se i participar en la vida urbana. A més, no limiten el terme "accessibilitat" a l'eliminació d'obstacles en la via, sinó que també inclou l'accessibilitat cognitiva, comunicativa, cultural, etc.

Aquestes ciutats inclouen tres aspectes que intenten implantar el millor possible:

1. **Aspectes físics**: asseguren la mobilitat de l'individu per tota la ciutat. Això per tal de garantir que pugui dur a terme totes les activitats quotidianes. Exemples d'aquests elements són: voreres amples i llises, rampes, absència d'obstacles com fanals, semàfors amb veu, cartells amb braille, etc.

2. **Aspectes econòmics**: es tenen en compte a les persones amb discapacitat a l'hora del disseny d'oportunitats laborals i productes. Alguns exemples serien: treballs adaptats, productes enfocats en les necessitats dels minusvàlids, etc.

[14] Organització mundial de la salud. Organisme especialitzat de les Nacions Unides fundat el 1948, l'objectiu del qual és assolir per a tots els pobles el màxim grau de salut.

3. **Aspectes socials**: faciliten la interacció i participació de tots els individus en la comunitat. Per exemple, museus amb guies especialitzats, escoles amb integració,etc.

El nombre de *smart cities* al món va en augment, però encara està molt lluny del necessari per aconseguir que tots els discapacitats puguin accedir-hi. Actualment, el *rànquing*[15] de ciutats millor adaptades i més accessibles és el següent:

1. Berlín (Alemanya)
2. Denver(EUA)
3. Gdynia(Polònia)
4. Milà(Itàlia)
5. Seattle(EUA)
6. Tel-Aviv(Israel)
7. Nantes(França)
8. Estocolm(Suècia)
9. Pamplona(Espanya)
10. Tallaght(Irlanda)

Si fem una mica més de zoom i busquem només a Espanya, el rànquing és el següent:

1. Àvila (Castella i lleó)
2. Lugo (Galícia)
3. Logronyo (La Rioja)
4. Màlaga (Andalusia)
5. Terrassa (Catalunya)
6. Santander (Cantàbria)
7. Burgos (Castella i Lleó)
8. Pamplona (Navarra)
9. Bilbao (País Basc)[16]

[15] La classificació d'aquest no està ordenada de cap manera.
[16] Aquesta llista tampoc no està ordenada amb cap tipus de criteri.

Totes aquestes ciutats comparteixen la característica d'haver guanyat o haver tingut mencions al Premi Ciutat Accessible (*Access City Award*), premi que la Comissió Europea, en col·laboració amb el Fòrum Europeu de la Discapacitat entrega cada any a les quatre ciutats que més avenços han fet i aplicat en aquest sector.

Amb totes aquestes innovacions i noves aplicacions que s'estan desenvolupant, podem dir que l'entorn està per fi adaptat per a les persones discapacitades, específicament pels invidents? Arran de l'entrevista realitzada a la presidenta de l'organització B1+B2+B3, surt a la llum (i nosaltres no hi podem estar més d'acord) que la resposta és "quasi, però no". Evidentment, s'ha aconseguit facilitar molt més la vida d'aquest sector de la població a través de tots aquests mitjans, però no són suficients. La gran majoria es concentra en les grans metròpolis, deixant sense tota aquesta innovació a petites ciutats i pobles on també es necessiten. A més, considerem que amb el nivell de desenvolupament tecnològic actual es podrien fer moltes més innovacions, però que no hi ha prou interès per fer-ho.

6. Precedents

6.1. Projectes anteriors relacionats

La idea d'utilitzar la tecnologia per facilitar l'autonomia a les persones invidents no és del tot nova, però sí bastant recent. Actualment, ens trobem a la segona generació de les anomenades *smart glasses*. La primera va néixer aproximadament el 2011, any en el qual algunes empreses es van començar a interessar en aquesta idea de negoci. Així, comencem des de petits models basats únicament en sensors que puguin detectar objectes a l'alçada del cap fins a arribar a avançats models que fan ús de la intel·ligència artificial i la neurologia.

Malgrat això, si ho comparem amb altres tecnologies hi ha hagut molt poc esperit d'innovació. Potser a causa del fet que el producte no es podria vendre bé. De fet,

l'únic *Big Tech*[17] que sí que s'ha arribat a interessar en aquesta idea ha sigut Google, que ha arribat a desenvolupar i posar en el mercat alguns models d'aquest tipus d'ulleres.

Després d'analitzar el gran nombre de dispositius d'aquest tipus en els que ens podríem inspirar, hem fet una selecció d'aquells que més ens han interessat. Ja sigui per les seves funcions, disseny, preu, etc. Són els següents:

6.1.1. Eyesynth

L'empresa Eyesynth ha dissenyat una de les més innovadores ulleres intel·ligents d'aquesta nova generació. Es tracta d'unes ulleres que, a través d'unes càmeres, registren l'espai del voltant. Llavors un microordinador processa la informació que arriba de les imatges i ho converteix en àudio. «Nosaltres ho anomenem "experiència de sentit augmentat"», declara l'empresa.

Figura 7. Ulleres Eyesynth.
Font: https://eyesynth.com/que-es-eyesynth/

Hi ha tres característiques que destaquen en aquest model:

Característiques:

- **Detecció en 3D.** Es pot identificar la forma, l'espai, la profunditat i la localització dels objectes.

[17] Gegant tecnològic. Concepte que fa referència a grans empreses del sector tecnològic que tenen una gran xarxa d'operacions i capital.

- **So abstracte**. No s'utilitza un idioma real per traduir l'entorn, sinó un nou tipus de llenguatge. Així, el coneixement d'idiomes no és un impediment per poder fer servir-les.
- **Àudio coclear.** El so es transmet a través dels ossos del cap, així s'evita fatigar l'oïda de l'usuari. Per aconseguir aquest àudio s'utilitzen uns petits coixins de goma que estan en contacte amb els laterals del cap.

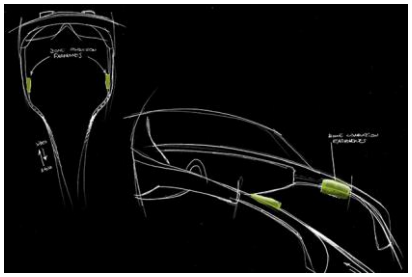


Figura 8. Esbós de les ulleres Eysynth amb el sistema d'àudio coclear indicat
Font: https://eyesynth.com/que-es-eyesynth/

Un altre gran punt a favor d'aquest dispositiu és que compta amb dos modes: rastreig i panoràmic. El primer només captura la zona central frontal, el segon ens dona informació de tota l'escena que ens envolta.

Especificacions tècniques:

Hardware:

- Processador Dual Core ARM
- Xip de processat FPGA d'alt rendiment
- Bateria: 5000 mAh
- Funcionament continu: 8 hores

Software:

- Modes operatius: Rastreig / Panorama complert
- Distàncies de visionament seleccionables: 6m, 2m, 0.7m.
- Nivell de bateria amb senyals acústics

Ulleres:

- Muntura: ABS, Zamak
- Lents: f1.8, FOV 76º
- 2 Sensors d'alta velocitat i lluminositat (60 FPS)
- Sistema piezoelèctric integrat d'àudio coclear.

6.1.2. Envision

Com abans hem mencionat, Google és de les poques grans empreses que es van interessar en les *smart glasses*. El model Envision és el resultat del seu treball en conjunt amb una startup[18] holandesa. És una combinació de Google Glass 2 (un model anterior) i la IA d'Envision.

Google no ha volgut explicar pautadament el seu funcionament, només sabem que processen les imatges captades amb la càmera i que l'anomenada IA s'encarrega d'entendre i comunicar la informació, que després és narrada en veu alta pel dispositiu.

Figura 9. Ulleres Envision
Font: https://www.tecnoaccesible.net/catalogo/envision-glasses

Algunes de les seves característiques són:

· **Lectura intel·ligent.** Pot llegir textos de llibres i imatges en veu alta per a l'usuari en més de 60 idiomes.

[18] Terme utilitzat per referir-se a empreses tecnològiques emergents.

· **Reconeixement facial.** Pot reconèixer rostres d'amics i avisar quan ens trobem enfront d'un.

· ***Wi-fi i Bluetooth*.** El dispositiu es controla a través d'un panell tàctil que es troba a les mateixes ulleres i es pot connectar amb el telèfon.

· **Detectar colors.** Pot detectar el color d'allò que estiguem veient.

· **Videotrucada.** És possible mantenir una videotrucada a través de les ulleres amb un conegut que s'instal·li l'app complementària.

Es poden augmentar les possibilitats de funcions descarregant l'app Envision a la Play Store.

Especificacions tècniques:

Sistema:

- Sistema operatiu: Android 8.0 (Oreo)
- Plataforma: Android
- Procesador: Qualcomm Quad Core
- Freqüència processador: 1700 MHz
- Freqüència procesador D8: 1700 MHz
- Memòria RAM: 3GB LPDDR4

Emmagatzament:

- Espai en disc: 32GB Emmc Flash

Connexions:

- Entrada/Sortida: USB C
- Descripció de connectors: Charging & Data: USB Type-C, USB 2.0 480Mbps.
- Xarxa sense fils: Bluetooth v5.2, Wi-Fi 802.11ac

Pantalla i imatge:

- Mòdul de pantalla òptica
- Resolució de pantalla: 640x360
- Càmera: 8Mp, 80 DFOV

Àudio:

- Altaveus: Altaveu mono, audio USB, audio BT
- Micròfon: Intern

Alimentació:

- 820 mAh amb càrrega ràpida

6.1.3. Liberty Delta

Aquest és un projecte desenvolupat per tres estudiants de l'Escola Superior d'Enginyeria de la Universitat d'Almeria (UAL); Antonio Daniel Guerrero, Aidas Dackus i Alejandro Pino. *Són tres dispositius que van junts, començant per unes ulleres amb sensors de proximitat que adverteixen d'objectes pròxims (. . .) i es trasllada aquesta informació a través d'unes bandes col·locades als peus perquè l'usuari conegui la seva posició en funció de si vibra el peu dret o esquerra*, declaren en una entrevista al diari InnovaSpain[19]. La informació dels sensors és processada i traduïda a través del mòbil, que al mateix temps està connectat a les ulleres.

Aquest model utilitza cables, encara que els implicats volen implementar-hi *Bluetooth* i un sistema GPS.

[19] Podeu trobar l'entrevista al link següent: https://www.innovaspain.com/universidad-almeria-liberty-delta-gafas-para-invidentes/.

Com en altres models que hem trobat, els creadors indiquen que es van basar en el sistema d'ecolocalització dels ratpenats per dissenyar el seu model fonamentat en sensors d'ultrasons.

Figura 10. Fotografia d'un dels creadors de les ulleres Liberty Delta utilitzant-les.

Font: https://www.20minutos.es/noticia/3716973/0/estudiantes-crean-gafas-dar-vision-invidentes/

No s'han trobat especificacions tècniques d'aquest model.

6.2. Característiques similars

Ara que ja tenim els tres models en els quals ens fixarem a l'hora de desenvolupar el nostre, és hora de veure quines són les característiques ens interessen extreure de cadascun per poder aplicar-les al nostre prototip.

Però abans d'això, hem de comparar i posar en comú els tres aparells per veure quines són aquelles funcions i/o particularitats en el disseny que es repeteixen als tres, ja que el fet que tots comparteixin una característica ens indica que aquesta resulta molt positiva, tenint en compte que comporta que tots tres equips van arribar a la mateixa conclusió.

El primer punt comú és la localització de l'aparell encarregat de la detecció de l'entorn. El fet de mantenir-lo en les mateixes ulleres i minimitzar la seva mida eviten que l'aparell sigui incòmode i limiti la mobilitat de l'usuari.

El segon és el forma i composició de les ulleres. Totes utilitzen materials lleugers, però a la vegada amb suficient resistència per evitar ratllades i petits danys. A més de mantenir colors neutrals com ara el negre, el blanc o el gris.

En tercer lloc, tenim que tots tres dissenys tenen pràcticament el mateix procés de captació d'informació i traducció en estímuls. El podem resumir amb l'esquema següent:

Per altra banda, pel que fa a funcions més específiques, les més rellevants i que més ens interessen per al nostre model són les següents:

· Detecció d'obstacles en el camí.

· Reconeixement i lectura en veu alta de textos (*Text to Speech)*

· Capacitat de detecció de nivells de profunditat (enfocat a obstacles com escales)

Aquests són els interessos inicials, les condicionants que es trobin més endavant poden modificar les idees preliminars i provocar canvis i modificacions en el model.

6.3. Anàlisi i captació d'idees

A continuació, veurem quina característica específica de cadascuna de les ulleres anteriors ens interessa implantar.

Començant pel model d'Eyesynth:

Càmeres	Sí
Sortida d'informació en forma d'àudio	Sí
Detecció en 3D	Ns
Sortida d'informació en forma de so abstracte	No
Àudio coclear	No

Figura 11. Característiques del model d'Eyesynth.

Model d'Envision:

Lectura intel·ligent	Sí
Reconeixement facial	No
Wifi i Bluetooth	No
Detecció de colors	No
Videotrucada	No

Figura 12. Característiques del model d'Envisión.

Model Liberty Delta:

Sensors de proximitat	No
Sortida d'informació en forma de vibració	Sí
Connexió sense fil	No

Figura 13. Característiques del model de Liberty Data

7. Fonaments teòrics

7.1. Com interpretar el món real a través de la informàtica?

El món en el seu conjunt és una gran barreja d'estímuls exteriors que cada individu percep d'una manera o altra. No obstant això, existeixen molts inconvenients (com en el cas tractat en aquest treball, les discapacitats) que poden dificultar i/o impossibilitar la recepció i interpretació d'aquests estímuls en qüestió. En aquest apartat abastarem part del gran ventall de dispositius que, gràcies a la informàtica, són capaços de detectar i traduir aquests estímuls per poder retornar la capacitat d'entendre el món completament a l'usuari.

Els aparells mencionats són els que poden tenir una relació directa amb el nostre projecte.

7.1.1. Sensors infrarrojos

Anomenem radiació infraroja, radiació IR o radiació tèrmica a la radiació electromagnètica amb major longitud d'ona que la llum visible. Tinguent així menor freqüència que aquesta.

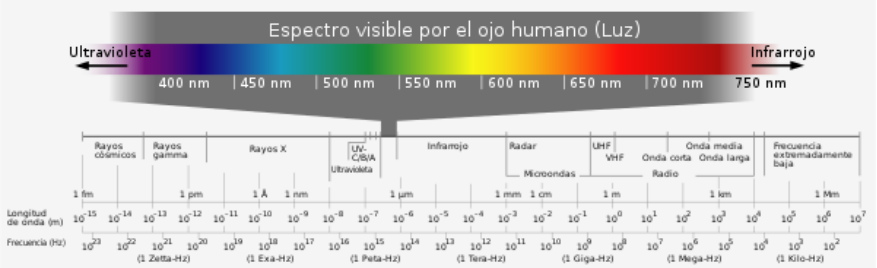


Figura 14. Esquema de l'espectre visible de la llum per l'ull humà

Font: https://es.wikipedia.org/wiki/Espectro_visible

Els sensors infrarojos són dispositius fotoelèctrics, receptors i emissors, l'espectre de treball els quals es troba entre els 700 i 900 nm. És a dir, en la longitud d'ona infraroja.

Tots els objectes amb temperatura superior als 0 Kèlvins (273 °C) emeten certa quantitat de radiació invisible per a l'ull humà. Els infrarojos s'encarreguen de mesurar la IR dels objectes en el seu camp de visió. Així, aquests dispositius estan dissenyats per detectar, localitzar i classificar objectes a través d'aquesta característica.

Dintre dels diferents tipus d'infrarojos, ens fixarem en els de circuit integrat. Aquests són els que tenen tant els receptors com els emissors en el mateix circuit tancat. Segons el seu funcionament en trobem dos tipus: reflexius i de proximitat.

> **Reflexius:** tenen un rang de treball de 3 a 5 mm. El receptor i l'emissor acostumen a estar en paral·lel. Funcionen de la següent manera: l'emissor emet un fes de llum que al xocar amb una superfície, depenent del color d'aquesta, produirà o no l'efecte de reflexió, essent així captat pel receptor.

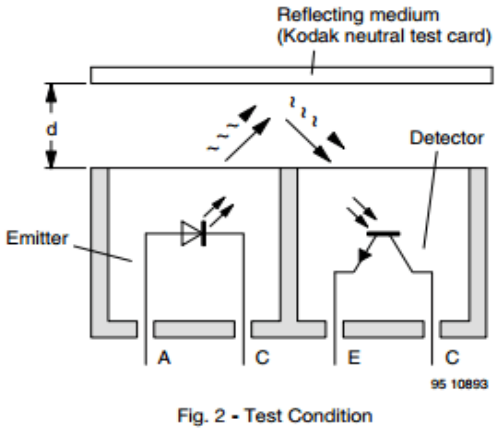


Figura 15. Esquema funcionament dels sensors infrarojos reflexius

Font: https://www.jsumo.com/cny70-transistor-output-reflected-light-detector

De proximitat: el LED emissor emet llum infraroja dispersa que, a través d'una lent, queda concentrada en un raig. Aquest surt projectat cap endavant i rebota al trobar un obstacle reflectant, formant un cert angle d'inclinació. La llum retornada es concentra en una altra lent, incidint tots els rajos en un únic punt; el receptor.

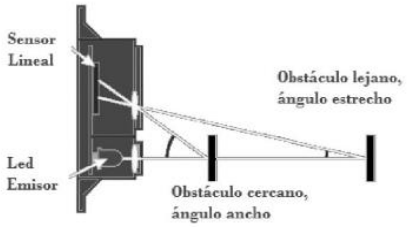


Figura 16. Esquema funcionament d'un sensor infraroig de proximitat.

Font: https://www.jsumo.com/cny70-transistor-output-reflected-light-detector

De cara a la detecció d'obstacles en el camí, els infrarojos de proximitat serien una bona opció a tenir en compte. Es poden utilitzar per imitar la capacitat d'ecolocalització dels ratpenats, substituint l'ona sonora per la llum i un estímul que tradueixi la distància a la qual es troba l'objecte.

El principal inconvenient d'aquesta opció serien les limitacions del rang de cada sensor, ja que per poder cobrir els gairebé 180° de visió serien necessaris més d'un; això implicaria afegir-hi més pes a l'aparell, traient-li comoditat al conjunt.

7. 1. 2 Càmeres

El següent aparell que pot encarregar-se de substituir la vista és la càmera. I no és d'estranyar, ja que el funcionament d'una és extremadament semblant al de l'ull humà.

De fet, sol dir-se que les càmeres són "traductors òptics": descomponen la llum en tres parts fonamentals (els colors vermell, blau i verd) a través d'un prisma de miralls dicroics[20]. A l'altra cara del prisma es troben els captors que reconstrueixen la imatge i envien la informació resultant al dispositiu.

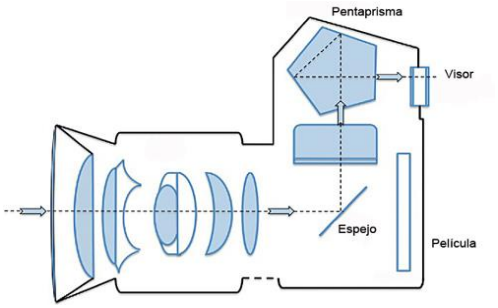


Figura 17. Esquema funcionament de les parts d'una càmera
Font: https://www.curso-fotografia-digital.com/blogs-de-fotografia/tips-y-consejos/128-partes-de-tu-camara.html

És gràcies a l'existència de les càmeres que s'ha pogut desenvolupar una de les últimes novetats informàtiques en el sector de l'òptica: la visió artificial. Però, de què es tracta?

Visió artificial

"La visió artificial o visió per computador és la ciència i la tecnologia que permet a les màquines veure-hi, extreure informació de les imatges digitals, resoldre alguna tasca o entendre l'escena que estan visionant."[21] Es tracta d'una nova branca del *Machine*

[20] Filtre dissenyat per reflectir només una porció determinada de l'espectre visible de la llum.
[21] I García, E. M. (2012). Visión artificial. *Fundación para la universidad Oberta de Catalunya*.

Learning[22] que entrena a algoritmes per tal que siguin cada cop més eficaços a l'hora de detectar, reconèixer, classificar, distingir i modelar imatges, entre moltes altres funcions.

Les tècniques de visió artificial acostumen a seguir el següent procés de treball:

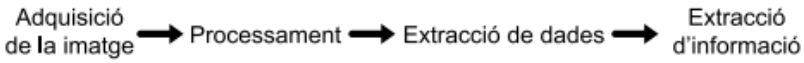


Figura 18. Passos realitzats per un sistema de visió artificial comú

Font: i García, E. M. (2012). Visión artificial. Fundación para la universidad Oberta de Catalunya.

La visió artificial ha sigut implantada ja en molts camps del món industrial, ja que facilita i accelera molts dels processos repetitius que les empreses intenten minimitzar.

Tot i que aquesta és una ciència immensa que ens dóna massa de què parlar, ens limitarem a posar un exemple d'un procés efectuat a partir de visió artificial per poder donar certa idea de la capacitat d'actuació que té.

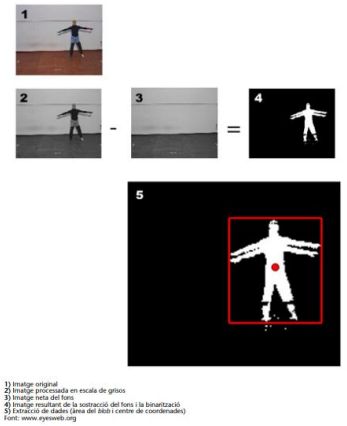


Figura 19. Imatges resultants de la segmentació i processat d'una imatge a través de tècniques de visió artificial.

Font: i García, E. M. (2012). Visión artificial. Fundación para la universidad Oberta de Catalunya.

[22] Aprenentatge automàtic o aprenentatge automatitzat. Subcamp de les ciències de la computació i una branca de la intel·ligència artificial que té com objectiu desenvolupar tècniques que permetin que els ordinadors aprenguin.

Aquest procés que passa per la classificació per escala de grisos, l'eliminació del fons i la binarització de la matriu podria ser part d'un algoritme de detecció de cossos per a càmeres de seguretat, per exemple.

Ara que ja hem introduït la visió artificial, hem de presentar les càmeres candidates a encarregar-se de l'obtenció de la imatge. I és que malgrat que hi ha moltes càmeres que es poden connectar a l'equip que es desitgi, ja que tenen drivers lliures, el més preferible són les càmeres intel·ligents, que ja estan especialment preparades per a aquesta funció. Hem trobat dues opcions que encaixen amb el que busquem:

PixyCam

Figura 20. Càmera PixyCam connectada a la seva placa corresponent.
Font: https://www.hwlibre.com/pixy-una-camara-inteligente-para-nuestros-proyectos/

Aquesta càmera, a més de poder-se connectar a altres dispositius sense problemes, no només emmagatzema dades; la seva capacitat arriba fins a poder (dintre de les imatges que grava) reconèixer, memoritzar i rastrejar colors i objectes.

Pixy compta amb el seu propi software: PixyMon. A través d'aquest es poden guardar i controlar les dades aconseguides per la càmera perquè no es perdi el progrés memoritzat.

A més, no només pot ser connectada a ordinadors, sinó també a plaques com Raspberry Pi o Arduino. Tot això ve acompanyat del fet que el seu preu és molt assequible si el comparem amb altres càmeres similars al mercat.

Els dos majors desavantatges d'aquesta càmera són que, en primer lloc, no és una càmera totalment intel·ligent, ja que una vegada iniciat el sistema hem de supervisar-la a través de PixyMon manualment. I, en segon lloc, trobem l'inconvenient que, si

utilitzéssim Arduino per al seu funcionament ens veuríem pràcticament obligats a canviar de llenguatge de programació, perquè Python no és directament compatible amb aquesta placa.

Microsoft Kinect

Aquesta càmera va ser desenvolupada per Microsoft per a la seva consola Xbox 360. Va ser dissenyada per poder percebre i reconèixer els cossos, moviments i veus dels jugadors. Per això incorpora dues càmeres frontals (una convencional de RGB i un sensor de distància), a més de diversos micròfons.

El sistema de percepció de profunditat té de tres parts: un projector làser d'infrarojos, un sensor CMOS[23] i un microxip que processa la informació. Va ser desenvolupat per PrimeSense, una companyia Israeliana. El seu funcionament es basa en la projecció d'un patró de punts aleatori i la seva lectura i triangulació mitjançant el sensor CMOS.

Aquest sistema és, precisament, el que més ens interessa d'aquesta càmera. La Kinect utilitza un procediment de llum estructurada per extreure la profunditat. L'emissor emet un patró de punts, i observant a través del sensor les variacions al patró es troba la forma i situació dels objectes mitjançant un procés de triangulació. Per realitzar la triangulació hi ha d'haver un patró previ de referència. Aquest patró, que és constant i es projecta a l'escena, el crea la divisió d'un raig únic emès pel làser.

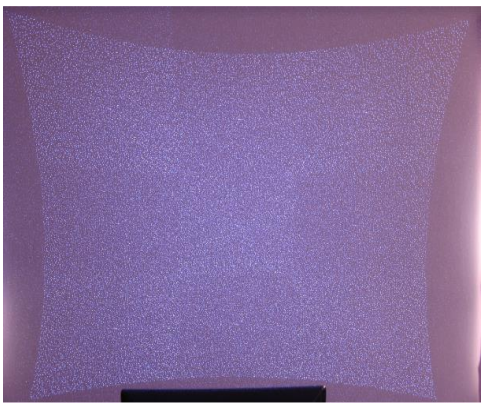


Figura 21. Patró de punts que projecta la càmera Kinect.
Font: TAZ-PFC-2013-649_ANE.pdf

[23] Un sensor de píxels actius capaç de detectar la llum.

A més de la detecció de profunditat, un altre aspecte positiu de la Kinect és el seu rang de treball. El camp de visió en horitzontal de la càmera és de 57° i el rang màxim de distància des de la càmera per a la captació de profunditat és de 0,8 metres a 4 metres, considerant que el rang òptim es troba entre els 0,4 metres i 3,5 metres, en la versió adaptada per Windows. Aquest és justament el rang en el qual treballem.

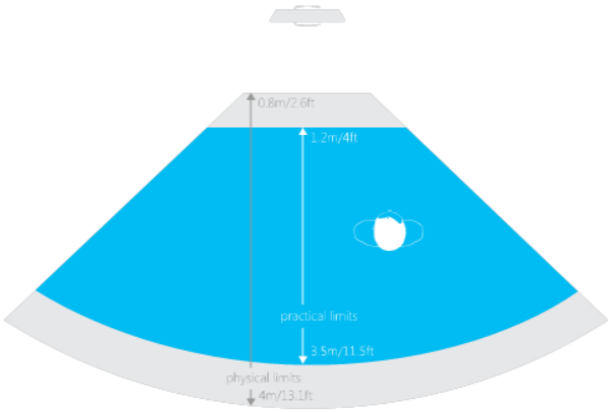


Figura 22. Camp de visió horitzontal de la càmera Kinect.
Font: https://zaguan.unizar.es/record/12845/files/TAZ-PFC-2013-649_ANE.pdf

La càmera consta d'un motor amb el qual es pot inclinar l'angle d'aquesta. Aquesta funció també ens va bé, ja que d'aquesta forma podem modificar l'angle d'acord amb les necessitats de dades que tinguem (en funció de si volem capturar en vídeo més part del terra o volem un enfocament més frontal).

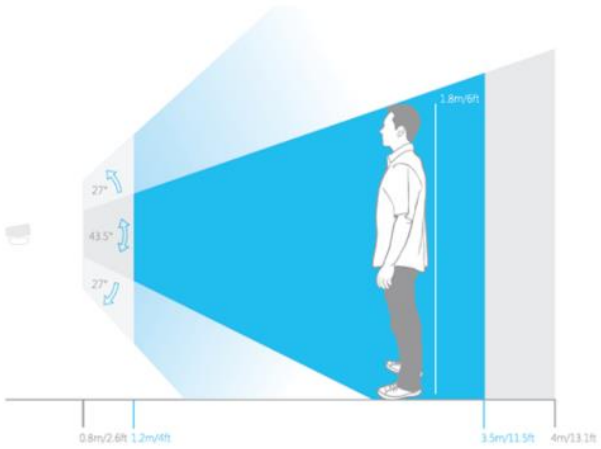


Figura 23. Camp de visió vertical modificable de la Kinect.
Font: https://zaguan.unizar.es/record/12845/files/TAZ-PFC-2013-649_ANE.pdf

7.2. Gestió de dades

Tota la informació que recopilem amb els sensors ha de ser processat en una unitat central. Aquesta unitat s'ha d'encarregar de rebre la informació, processar-la i executar certes línies de codi segons els resultats que hagi obtingut de les dades entrants. Sense una unitat central, les dades seran inútils, ja que no ens permet actuar en conseqüència a elles.

Els ordinadors constitueixen una de les unitats centrals més comunes que solem trobar. No obstant això, existeixen molts tipus d'ordinadors, amb característiques i utilitats diferents. En el cas d'aquest projecte depenem d'aquest ordinador per poder donar vida i sentit al projecte, de la mateixa manera que també necessitem que aquest tingui certes característiques per tal que es pugui adaptar a les nostres necessitats:

1. Mida:
   El resultat d'aquest projecte ha de ser un prototip que ha de poder ser fàcilment transportable. No ens és útil un aparell amb àmplies mides, per molta potència que ofereixi. Si no és de mides reduïdes no ens permetrà fer-lo portable i, per tant, perdrà tota utilitat.

2. Eficiència energètica:
   Consisteix a obtenir els millors resultats possibles, consumint en mínim d'energia possible. Necessitem que l'ordinador sigui molt eficient energèticament, ja que no disposarem d'una font d'alimentació contínua la qual pugui estar en moviment. Com més eficiència energètica tingui, més autonomia podrà tenir l'aparell. Així, no ens serviran ordinadors d'alts consums, perquè aquests serien gairebé impossibles d'alimentar de forma còmoda i eficaç.

3. Versatilitat:

És la capacitat de poder adaptar-se ràpidament a noves situacions o funcions. Necessitem una placa capaç de poder fer múltiples funcions, és a dir que no estigui únicament dedicada a un cert tipus d'instruccions o processos. Exemples d'ordinadors poc versàtils serien la gran majoria de microxips que incorporen molts aparells electrònics actualment. Són molt bons en la seva funció específica, però no en altres.

4. Econòmic:
   Un dels requisits que també demanem a aquesta unitat central és que tingui un preu contingut. L'objectiu és poder muntar un sistema de guiatge el qual sigui en gran manera econòmic. Sense haver de gastar tanta quantitat de diners com en altres aparells d'ús similar.

5. Potència de càlcul:
   Per poder dur a terme el projecte, necessitem una unitat de gestió de dades que sigui capaç de poder efectuar totes les operacions que necessitem en el menor temps possible. Ordinadors molt potents solen tenir poca eficiència energètica i ordinadors poc potents solen ser molt eficients energèticament.

Un cop hem analitzat les principals característiques que hem de tenir en compte, ara cal investigar quin tipus d'ordinador en serà més útil. Per això cal saber com es distingeixen els ordinadors entre si i quines són les diferències entre cada tipus. No obstant això, primer necessitem aclarir certs conceptes:

Anteriorment, ens hem referit a la unitat de processament central com a ordinador. Tanmateix, cal aclarir que les diferències entre aquests rau, bàsicament en el seu nucli central, anomenat Central Processing Unit (CPU), popularment conegut en català com a Processador. Si bé també hi ha altres components que varien i que influeixen en certa manera al producte final (com potser placa base, unitat de disc dur, etc.) ho fan en molt poca mesura i de forma gairebé negligible si tenim en compte com

s'executarà el codi que programem. En altres paraules, afectarà l'experiència d'usuari, però en cap cas ho farà a la velocitat i el rendiment de l'execució del programa.

En segon lloc, parlarem principalment d'arquitectures que es basen en un cert nombre de bits (8 bits, 32 i 64 bits). Aquesta terminologia es basa en com està construït el processador interiorment. Aquests bits es refereixen a la màxima quantitat de dades (és a dir vuit espais on poden anar valors positius (1) o negatius (0)) que pot processar com a màxim el processador en qualsevol moment per cada registre (un registre és una part de la memòria la qual s'encarrega d'emmagatzemar informació que indiquen quina dada s'ha de processar) Aquest terme també és conegut com a "Amplada de dades".

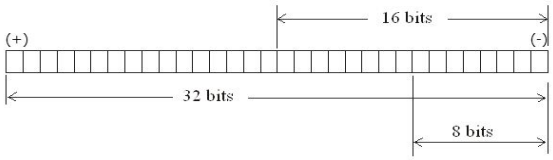


Figura 24. Espai que ocupa un registre. Dins d'aquest registre es poden emmagatzemar fins a 64 bits
Font: https://slideplayer.es/slide/26606/

Cal Observar que com més bits tingui un registre, més gran serà el nombre que podrà emmagatzemar i, per tant, podrà executar instruccions més complexes.

Aclarits els conceptes, podem començar a investigar quins són els principals tipus d'ordinadors. Com hem dit que l'element que més influeix en els ordinadors (per les tasques que volem fer) és el processador, donarem un cop d'ull als diferents tipus de processadors i en què es diferencien, tan arquitectònicament com des del punt de vista de la seva utilitat.

7.2.1. Processadors de 8 bits

En primer lloc, començarem per descriure els tipus de processadors més bàsics que s'usen avui en dia. Parlem dels processadors de 8 bits.

Aquests processadors van ser els primers a aparèixer en el mercat consumista a finals de la dècada dels 70. El processador més conegut amb aquesta arquitectura (i també un dels primers a aparèixer) va ser l'Intel 8008.

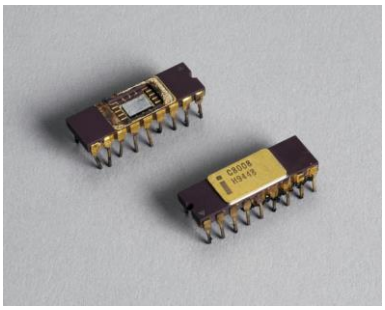


Figura 25. Dos exemplars d'Intel 8008.
Font: https://collection.sciencemuseumgroup.org.uk/

Aquests processadors són coneguts per la seva senzillesa (eren els primers processadors destinats al sector domèstic). La seva arquitectura és simple perquè, com hem dit anteriorment, només és capaç d'emmagatzemar registres de vuit espais. La qual cosa vol dir que només podrem executar programes senzills.

Aquests processadors van ser dels primers a aparèixer i porten entre nosaltres des de gairebé el principi de la informàtica, són simples i la gran majoria d'ells tenen molts anys, per la qual cosa seran poc eficients i difícils d'utilitzar amb llenguatges de programació actuals. Per tant, com pot ser que tinguem en compte als processadors de 8 bits si, aparentment, no tenen res a fer en contra d'altres processadors més nous?

La resposta no la trobarem pas en els vells processadors de 8 bits. En canvi, sí que la trobarem en els nous processadors que s'han construït i dissenyat partint de l'arquitectura de 8 bits, però amb un enfocament completament diferent del que els processadors originals tenien. Parlem del projecte Arduino:

Arduino neix el 2003, a l'Institut italià IVRAE. Allà Massimo Banzi, estudiant de l'institut, va veure la necessitat de crear una placa amb microcontrolador que fos accessible econòmicament per a tots els estudiants de computació i electrònica del seu institut. Aleshores, les plaques dirigides a estudiants de programació, les BASIC Stamps, valien molts diners i poca gent se les podia permetre. Així, decideix crear un

primer prototip (el qual encara no podem anomenar Arduino). A mesura que anava desenvolupant el prototip, va anar atreien d'altres persones que van voler treballar amb ell al projecte, oferint les seves capacitats per millorar la primera idea i poder treure al mercat la primera placa.

D'aquesta forma podem definir que Arduino és una plataforma de creació d'electrònica de codi obert. Que sigui de "codi obert" voldrà dir que qualsevol persona o empresa pot accedir als esquemes i els diagrames electrònics d'Arduino i pot crear el seu propi Arduino a base d'aquests diagrames, sense haver de pagar res a l'empresa. De la mateixa forma que també publiquen software per poder controlar la placa que també és de codi públic, tothom pot accedir a ell sense necessitat de pagar, a condició que no en facis negoci amb ell. Actualment, aquest software per controlar l'Arduino s'anomena Arduino IDE.

A diferència dels processadors de 8 bits antics, Arduino no està pensada per programar en ella. La programació es fa des d'un altre ordinador, mitjançant l'Arduino IDE on es compila el programa escrit en C++ i s'envia ja codificat per tal que el processador ho pugui entendre mitjançant el port USB que incorpora la placa. Gràcies a la simplicitat del processador, resulta molt fàcil de programar en comparació d'altres programes i conjuntament amb els múltiples ports a través dels quals el processador és capaç de controlar components electrònics fa que aquesta placa sigui una de les principals opcions si es vol fer projectes d'electrònica i programació.

Figura 26. Totes les plaques oficials Arduino que existeixen actualment
font: https://www.xataka.com/makers/

Avantatges d'Arduino pel nostre projecte:

- Facilitat de Programació:

  Tot i tenir un processador relativament senzill i bàsic, Arduino ens ofereix un entorn senzill de fer servir i ràpid d'aprendre, el qual pot arribar a desenvolupar grans projectes de programació. Altres processadors són més difícils de programar i necessites aprendre llenguatges de programació molt específics (Arduino necessita C++, un llenguatge molt corrent en informàtica).

- Gran comunitat programadora:

  Gràcies al fet que és Open Source (Codi lliure), hi ha multitud de persones organitzades en fòrum i a través d'internet en general que programen i aporten les seves idees per desplegar-les en aquestes plaques. Així, hi ha molts complements i llibreries desenvolupades per Arduino, cosa que facilita i molt el desenvolupament de qualsevol projecte.

- Poc consum energètic:

  Que Arduino estigui equipat amb un processador poc potent no té per què ser dolent. Gràcies a aquest, l'Arduino consumeix molt poc i pot ser alimentat amb qualsevol bateria per fer-lo portàtil.

- Mida reduïda:

  La majoria de plaques d'Arduino tenen una mida relativament reduïda (com a màxim la mida d'un telèfon mòbil). Si bé hi ha plaques amb mides grans, depèn del nombre d'entrades i sortides que vulguem. En aquest cas, no ens són necessàries moltes sortides, pel qual podem obtenir un Arduino en dimensions acceptables pel projecte.

- Preu:

En ser una placa la qual té els seus diagrames penjats a internet de forma pública, podem obtenir una gran varietat de plaques diferents d'un preu molt assequible (la majoria es troben al voltant dels 30 euros).

Mancances que hi trobem:

- Poca potència:

  Com hem dit anteriorment, l'Arduino té un consum molt reduït gràcies a la seva poca potència. Si bé això està molt bé en termes energètics, ens posa moltes restriccions sobre les magnituds que pot tenir el programa que executi l'Arduino. És difícil poder trobar un programa capaç de desenvolupar totes les tasques que vulguem en un processador tan petit com pot ser qualsevol dels que porta l'Arduino.

- Impossibilitat de poder executar un S.O.:

  La segona mancança ve també com a conseqüència directa de la primera. Com que no té un processador potent, tampoc és capaç de poder executar algun sistema operatiu. La seva arquitectura de 8 bits tampoc ajuda, ja que els sistemes operatius disponibles per aquestes arquitectures es redueixen els primers sistemes operatius, completament obsolets i, a més no cabrien dins de les memòries de l'Arduino (anteriorment, els processadors de 8 bits tenien unitats externes per poder emmagatzemar més dades, per l'ús que se li dona a l'Arduino, no existeixen). Molts recursos de programació necessiten un S.O. per poder funcionar.

- Poca compatibilitat amb accessoris que no són per fins de prototipatge:

  Arduino representa un entorn perfecte per poder desenvolupar els teus petits projectes i connectar-hi els teus sensors electrònics, els teus components electrònics i tots els accessoris que puguis incorporar-hi. No obstant això, no disposa de controladors per ports USB. Si bé incorpora un USB, aquest el necessitem per poder comunicar l'Arduino amb l'ordinador, per tal de

programar-lo. Aquests ports resulten fonamentals per poder connectar altres eines que no tenen per què tenir connexions estàndard d'Arduino (cables que has d'introduir a cada port). Així, quedarien descartats molts dispositius que ens poden ser útils en aquest projecte, com pot ser el cas de la Microsoft Kinect.

7.2.2. Processadors x86/64

Fora dels processadors de 8 bits només han sobreviscut avui en dia els processadors de 32 i 64 bits. Aquests es divideixen en dos grans grups, segons el conjunt d'instruccions que fan servir:

Les instruccions no són altra cosa que les unitats més petites de codi que fan que el processador faci una tasca o altra[24]. El conjunt d'instruccions que pot entendre un processador s'anomena ISA (Instruction Set Architecture).

El fet que existeixin vàries ISA no és casual, cada una està dissenyada per poder construir processadors diferents, amb propòsits diferents. En dissenyar un processador, els enginyers han de tenir en compte l'ISA en què funcionarà el seu processador, ja que la disposició dels transistors dins del processador variarà depenent d'aquesta. Així, no es pot executar un processador amb instruccions diferents de les que està dissenyat per funcionar. Si ho féssim, el processador no encendria. Bàsicament, el disseny intern del processador influeix directament en el conjunt d'instruccions que com a programador pots utilitzar. No es tracta d'utilitzar un conjunt o altre depenent dels interessos que necessitis en aquell moment. Un processador només pot funcionar amb la ISA pel qual ha estat fabricat.

Arribats a aquest punt, ja estem en disposició de presentar el primer conjunt d'instruccions que podem trobar avui en dia i el primer que es va crear: CISC.

[24] Cal recordar que el que nosaltres anomenem "tasca" o "programa" no deixa de ser el resultat de moltes operacions matemàtiques calculades pel processador.

Ens situem en plena dècada dels 50, la informàtica ja ha començat a donar els seus primers passos, els primers ordinadors ja s'han inventat i ara evolucionen cap a ordinadors més compactes (veníem dels primers ordinadors, gegants, com l'ENIAC). És just en aquest procés de miniaturitzar cada cop més els ordinadors on una empresa, IBM, s'adona que tots els ordinadors es desenvolupen aïllats els uns als altres, resultant en ordinadors que no poden executar els mateixos programes. En altres paraules: cada processador tenia les seves instruccions i, per tant, era impossible poder programar un software capaç d'executar-se en tots els ordinadors. Així, IBM va reunir a un grup d'experts per mirar de crear un conjunt d'instruccions que fossin prou potents i universals per a poder desenvolupar processadors en els quals es pogués executar un mateix programa i sense haver de limitar-los en termes de potència. Aquesta set, acabaria imposant-se a tots els altres fins a arribar als nostres dies.

CISC té la filosofia de desplegar instruccions complexes, però que ocupen poc espai. Aquest fet fa que no s'hagi de consultar tantes vegades a la memòria RAM[25], el qual consumia una gran quantitat de temps. Actualment, encara hi ha un petit temps d'espera entre Processador i memòria, però és gairebé inapreciable. Tot i això, en ser aquestes instruccions complexes, solen tardar més d'una volta de rellotge a completar-se.

Aquestes arquitectures les podem trobar en la majoria d'ordinadors actuals. Es troben en els processadors fabricats per Intel i AMD, empreses que acaparen la majoria de les vendes de xips al món. Aquestes dues empreses creen i venen processadors tant com per a portàtils, Ordinadors d'escriptoris, o com per servidors.

[25] Random Acces Memory.

Figura 27. Esquerra: Processador AMD, Dreta: Processador Intel
font: https://www.techspot.com/review/2351-intel-core-i9-12900k/

Avantatges:

- La seva popularitat ve deguda al molt bon rendiment que donen els seus processadors. Ofereixen molta més potència per nucli i també en múltiples nuclis que les altres arquitectures.

- Aquests processadors també gaudeixen de la compatibilitat de la majoria de programes i sistemes operatius presents avui en dia, cosa que els proporciona una universalitat molt elevada (no t'has de preocupar per veure si un programa és compatible o no).

- Gran diversitat de potències i consums (podem trobar processadors amb tots els múltiples de 2 fins arribar a 16). Consumiran més els processadors amb més nuclis que els que menys.

Desavantatges:

- Necessitem una gran quantitat de diners per poder obtenir un ordinador complet.

- La gran quantitat de transistors i la potència que aporten fan que consumeixin molta més energia (les últimes generacions s'acosten als 170 W[26]) encara que només siguin uns pocs nuclis.

- Els grans consums d'energia que necessiten fan que produeixin molta calor, la qual s'ha de dissipar. Aquesta escalfor s'ha de dissipar amb mètodes que tenen un cert volum, per la qual cosa el conjunt de tot el sistema amb un processador CISC pot arribar a ser molt voluminós.

[26] TDP del AMD Ryzen 9 7950X, el processador més potent segons la mateixa AMD i experts que es basen en el rendiment d'aquest comparat amb els altres.

7.2.3. Processadors ARM

Entrar en el món dels 32 i 64 bits és gairebé sinònim de fixar-se en els processadors més utilitzats en el món dels ordinadors d'escriptori i portàtils, aquells fabricats per Intel i AMD (x86/64). No obstant això, des de fa ja uns quants anys, hi ha un altre tipus de processadors que comencen a emergir i a posicionar-se just al costat dels processadors d'arquitectures típiques. Aquests processadors són els processadors ARM.

La principal diferència entre ARM i x86 és el conjunt d'instruccions (ISA) que pot entendre un processador. De fet, aquestes dues instruccions són les dues més usades al món, deixant un espai gairebé inexistent a altres. Parlem dels conjunts d'instruccions CISC i RISC. Com és evident, si els processadors x86/64 feien servir l'arquitectura CISC, els processadors ARM empraran l'arquitectura RISC (Reduced Instruction Set Computer).

Els processadors i l'arquitectura RISC que avui coneixem es van desenvolupar gràcies a l'interés d'una empresa (Acron Computers) en aquesta tecnologia. El 1980, Acron Computers crea ARM (Advanced RISC Machines) amb l'ajuda d'Apple computers i VLSI Tecnology. Aquesta empresa es dedicarà exclusivament a desenvolupar l'arquitectura RISC i a dissenyar processadors molt més potents.

Part de l'èxit de RISC i ARM ve donat gairebé completament a l'aparició de dispositius mòbils com els telèfons intel·ligents, tauletes, rellotges intel·ligents, Routers, NAS, calculadores, etc. Aquests dispositius necessiten processadors amb potència, però sobretot amb una gran eficiència energètica. Avui en dia, fins i tot podem trobar processadors ARM en portàtils i ordinadors d'escriptoris, com és el cas d'Apple i els seus MacBook, iMac i Mac Studio.

Figura 28. Processadors RISC dissenyats per ARM i fabricats per diferents marques
font: https://www.xataka.com/ordenadores

Avantatges ARM:

- Mida reduïda, ens permet fabricar ordinadors amb mides molt contingudes

- Baix consum: En tenir menys transistors, el consum disminueix dràsticament, permetent-nos poder alimentar-lo amb bateria.
- Genera poca calor: Al tenir baix consum no desprèn tanta calor, tot i necessitar dissipadors, no en necessita de tan grans com els CISC.

- Poden incorporar coprocessadors que ens agilitzin certes tasques (alliberen més al processador principal)

- Preu més baix que els x86/64

Desavantatges ARM:

- Menys potència que el x86/64

- Els softwares i sistemes operatius no són compatibles a no ser que s'hagin programat per treballar amb ells (Linux)

7.2.4 Elecció d'arquitectura i placa:

Si comparem les diverses arquitectures que tenim disponibles avui en dia, podem descartar una d'elles gairebé immediatament. L'arquitectura x86/64 no s'adiu a les necessitats establertes al principi d'aquest apartat. En primer lloc, no disposa d'una bona situació en el mercat de la informàtica. Construir un ordinador bàsic, capaç de poder funcionar, sobrepassaria la barrera dels 100 €. A més, la seva mida seria considerablement massa gran pel projecte que nosaltres volem dur a terme. Si bé

existeixen ordinadors relativament petits amb aquesta arquitectura, no ho serien suficient per poder ser tan portàtil com nosaltres volem. Tampoc pot ser escollit per la seva eficiència energètica, ja que tot i tenir un molt bon rendiment, el seu consum energètic no està gairebé sostingut (en el millor dels casos 6,5W només el processador). Per contra, sí que complirien amb el requisit de versatilitat, ja que ens permeten poder dur a terme moltes tasques diferents. En resum, aquesta arquitectura no compliria els següents requisits:

- Preu reduït
- Mida reduïda
- Eficiència energètica

Per contra, sí que complirien amb els següents:

- Versatilitat
- Potència

Amb dues arquitectures restant, caldrà poder veure cada una quines de les característiques que hem plantejat anteriorment no compleixen i quines sí per tal de poder descartar-ne una.

L'arquitectura de 8 bits (generalitzarem al projecte Arduino) ens ofereix una molt bona eficiència energètica (la millor de les tres), a canvi d'una potència bastant reduïda. La seva mida és molt continguda i pot ser adaptada segons el tipus de placa Arduino que escollim. Podem optar per plaques molt petites, com l'Arduino nano, amb mides de només 45mmx18mm o bé per d'altres amb més grandària però estant encara dins de mides acceptables. Pel que fa al seu preu, també està molt contingut, per només 30 € pots obtenir una d'aquestes plaques. No obstant això, hi ha un dels apartats els quals no compleix en absolut. L'Arduino no és gens versàtil o, com a mínim, no ho és tant com ho necessitem. Si bé ens permet programar-lo amb codis personalitzats no té capacitat de poder executar i adaptar-se al nivell de programari que nosaltres tenim

previst desenvolupar. Gran part del problema ve degut a la poca potència que aquesta placa ens proporciona. Si fem un resum, quedaria així:
Punts forts de l'arquitectura de 8 bits:

- Eficiència energètica
- Mida reduïda
- Preu contingut

Els seus punts febles:

- Poca potència
- Poca versatilitat

Com podem apreciar, els Arduino tampoc s'acaben d'adaptar a les nostres necessitats. Per tant, només ens queda l'arquitectura ARM. Veiem, doncs, quins avantatges i inconvenients ens ofereix:

En primer lloc, l'arquitectura ARM ens permet executar molts més tipus de codi que l'arquitectura 8 bits. De fet, ens permet executar els mateixos codis que l'arquitectura x86/64, la qual cosa vol dir que és molt versàtil. Una altra característica que també compleix és la d'eficiència energètica. Tot i no ser tan eficient com Arduino, el seu consum és bastant reduït gràcies a tecnologies que incorpora. Pel que fa a la potència, els seus nuclis estan dissenyats per poder aprofitar al màxim l'energia i poden oferir unes potències raonables si se'ls acompanya d'un software optimitzat per a ells. Els ordinadors que contenen processadors ARM tenen unes mides suficientment reduïdes. Les seves mides no solen superar les mides dels telèfons mòbils més grans (uns dels dispositius ARM més grans que existeixen). La mida d'un telèfon mòbil és molt adient al treball, ja que el fet que molta gent ja utilitzi el telèfon mòbil farà més fàcil poder transportar-lo (es poden portar a les butxaques, motxilles, bosses, etc…). Per últim, hem de parlar de preu. Si bé els telèfons mòbils no són precisament barats, podem trobar plaques amb ARM especialitzades per aquest tipus

de projectes amb un preu molt més econòmic (aproximadament 40 euros). Per tant, tenim el següent:
Punts a favor:

- Bona potència de càlcul
- Mida continguda
- Preu assequible
- Eficiència energètica
- Versatilitat

Aquesta arquitectura no disposa de cap punt en contra, per la qual cosa podem afirmar que serà la que hàgim d'usar. Concretament, usarem una d'aquestes plaques ARM dissenyades per a prototipatge. Aquesta s'anomena Raspberry Pi. En la seva última versió (Raspberry Pi 4) ens ofereix les següents característiques:

- Processador Broadcom BCM2711 (4 nuclis ARM cortex A-72)
- Gràfica VideoCore VI
- 4 GB memòria RAM
- Connectivitat wifi i Ethernet
- Ranura per targeta SD (serà el seu disc dur)

7.3. Firmware

7. 3.1. Llenguatges de programació

Entenem com a llenguatge de programació aquell llenguatge amb normes gramaticals estrictament definides que permeten al programador donar instruccions amb el fi de controlar el comportament del sistema per tal d'executar diferents treballs. El conjunt d'ordres escrites a través d'aquest llenguatge s'anomena programa informàtic.

7.3.2. Tipus de llenguatges

Podem classificar els diferents llenguatges de programació segons les seves característiques de la següent forma:

· Llenguatge de baix nivell
· Llenguatge de programació d'alt nivell

El primer pas en l'evolució dels llenguatges de programació, el llenguatge de baix nivell, va ser el desenvolupament del què es coneix com a llenguatge assemblador. En un llenguatge assemblador, s'utilitzen mnemotècniques per representar operacions a realitzar per l'ordinador i cadenes de caràcters que representen adreces d'ubicacions a la memòria de l'aparell on s'emmagatzemen els operands. Com que el llenguatge s'adapta a l'estructura del processador d'un ordinador particular, és una màquina dependent. Un traductor anomenat assemblador tradueix un programa escrit en llenguatge assemblador a un conjunt d'instruccions de màquina, que pot ser executat per un ordinador.

Python

Python és un llenguatge de programació d'alt nivell amb el qual es poden desenvolupar aplicacions de gairebé qualsevol classe. Es considera un llenguatge de paradigmes múltiples, que admet programació funcional, estructurada i orientada a objectes.

- Avantatges

  Un dels punts més favorables d'aquest llenguatge és que es tracta d'un llenguatge interpretat, és a dir, que no fa falta compilar-ho per executar les apps que s'hi escriuen, sinó que s'executen directament a través d'un "interpretador". Així, no és necessari passar el codi a llenguatge de màquina. Altres dels seus avantatges és que és un llenguatge senzill de llegir i escriure, ja que manté molta semblança amb el llenguatge humà. A més, és un

llenguatge multiplataforma de codi obert, cosa que permet que tingui una àmplia biblioteca oberta al públic.

- Desavantatges:
  Les característiques intrínseques de Python el tornen també un llenguatge bastant lent i que consumeix força memòria. No obstant això, es pot intentar optimitzar l'aplicació desenvolupada per disminuir el nivell d'aquests inconvenients.

7.3.3. Llibreries

Les llibreries (a programació) són conjunts d'arxius importables, compostos per dades i codi, que s'utilitzen en el desenvolupament del software. El seu fi és facilitar la programació, ja que proporcionen diverses funcionalitats comunes que ja han sigut resoltes prèviament per algú altre, estalviant-te així molt
temps de desenvolupament.

Les llibreries que més ens interessen per a aquest treball són les següents:

TensorFlow: És una llibreria de codi obert per a la computació numèrica i Machine Learning a gran escala. TensorFlow és capaç d'entrenar i executar xarxes neuronals profundes per complir tasques com ara la classificació de dígits escrits, el reconeixement d'imatges, processament de llenguatge natural, etc. La seva API és Python, encara que també s'hi utilitza C++.

OpenCV (Open Computer Vision): És una llibreria lliure i multiplataforma de visió artificial. Algunes de les seves moltes aplicacions són: reconeixement facial i de gestos, segmentació, realitat augmentada, etc. La seva API és C++.

Libfreenect: És una llibreria que permet accedir a la càmera Microsoft Kinect i les seves funcions a través d'una connexió USB. Facilita l'accés a les càmeres de profunditat i RGB, al motor, els acceleròmetres i el d'activitat. Aquesta llibreria es troba dintre del projecte OpenKinect.

8. Disseny del programa informàtic

La idea principal d'aquesta part pràctica és poder desplegar una solució tecnològica capaç de poder ajudar a qualsevol persona amb problemes de visió (sigui baixa o nul·la visió). En tractar-se de ceguera, creiem que hauríem de desenvolupar un prototip capaç de poder veure tot allò que la persona portadora no podrà.

En fer el prototipatge ens fixarem especialment en les situacions en ambients exteriors, on creiem que les persones cegues poden tenir més dificultats per poder conviure amb normalitat. Si bé és probable que el resultat final pugui acabar funcionant en alguns casos en interiors, aquest no serà el nostre objectiu.

Per començar la part pràctica d'aquest treball, ens hem de fixar en quins són els principals obstacles que la gent amb problemes de visió pateix més. Tal com hem anat observant durant el desenvolupament de la part teòrica podríem recollir uns quants candidats a ser evitats.

En primer lloc, cal destacar els objectes que la presidenta de l'associació B1+B2+B3 va dir, sota la seva experiència, que eren els més molestos per la gent amb problemes de visió. Podríem dir que aquests són objectes comuns en vies públiques com podrien ser els següents:

1. Jardineres
2. Bancs
3. Senyalització de baixa altura (pilones, esferes, petites tanques…)
4. Cartells situats al terra
5. Objectes verticals i prims (fanals, pals elèctrics, senyalització vària…)

6. Desnivells pronunciats (com escales, imperfeccions en la vorera, bordó…)

Com que la presidenta Morancho va deixar la llista d'objectes oberta, ja que a cada persona li molesta certs objectes en particular, hem decidit acabar de completar-la amb objectes que podrien molestar o ser perillosos per aquestes persones segons el nostre criteri. Així creiem que els següents objectes també haurien de ser inclosos:

1. Persones(sobretot si van distretes i no veuen la presència de la persona invident)
2. Automòbils (cotxes, motocicletes…)
3. Vegetació (arbres, arbustos plantats a terra…)
4. Animals (gossos, gats, altres si s'escau)

Ara que ja sabem, a gran escala, els objectes que hem de ser capaços de detectar, hem de pensar com els detectarem. Segons el nostre criteri, podem dividir tots aquests objectes en dos grans grups.

El primer d'ells només el compon un dels objectes, en aquest cas les escales i desnivells. A diferència dels altres objectes, les escales es caracteritzen i es diferencien de la resta per ser un desnivell, una diferència d'altures. Si som capaços de digitalitzar les mides de l'entorn real, podríem ser capaços de detectar aquests desnivells quan trobem una diferència molt pronunciada entre dos punts. És en aquest punt on entra en joc la càmera Kinect amb el seu sensor de profunditat i el seu mapa de profunditat. Aquest mapa ens permetrà poder seguir les altures i diferenciar els esglaons i bordons.

En el segon grup, tenim la resta d'objectes a detectar. A diferència de les escales i bordons, aquests objectes no poden ser diferenciats per diferències d'altures ni per mapes de profunditat. El principal problema és la gran diversitat d'objectes que poden entrar dins d'una mateixa categoria. Per posar un exemple, dins de la categoria de "pilones" podem trobar multitud de formes, altures i colors. Tot i ser diferents, s'han

de tractar com una sola cosa. Utilitzarem les imatges que ens pot proveir la càmera RGB de la Kinect per poder detectar-los

8.1. Raspberry Pi 4

Com hem vist en apartats de la part teòrica, si comparem totes les opcions de computació que hi ha disponibles actualment, destaca la Raspberry Pi 4. Aquesta ens permetrà anar calculant i processant tota la informació que anem obtenint de les dues parts del projecte. A més, al ser de petites mesures i en poder-se alimentar amb una bateria externa de mòbil ("powerbank"), ens permetrà fer-la portable i poder usar-la al carrer.

El primer pas que necessitem fer és fer-nos amb una Raspberry Pi 4, preferentment amb una de 4 GB de memòria ram o bé la de 8 GB. Cal destacar que en un principi la versió de 4 GB ens hauria de sortir per, aproximadament un preu de 55$ (52,2 euros al canvi actual), però, per culpa de l'actual situació política, sanitària i logística[27] és impossible fer-se amb una d'aquestes plaques noves al seu preu original. Tot i no poder adquirir una nova placa, sempre podem recórrer al mercat de segona mà. Tot i haver-hi venedors que ofereixen aquestes plaques, el seu preu tampoc s'aproxima a l'inicial. A causa de la llei de l'oferta i la demanda, els venedors poden apujar el preu de les plaques, ja que són els únics proveïdors d'aquestes i saben que els compradors pagaran el preu que sigui per obtenir una. En el nostre cas, vam ser capaços de fer-nos amb una d'aquestes plaques a un preu raonable comparat amb altres ofertes que havíem buscat. El nostre venedor ens oferia un lot que incloïa la Raspberry Pi 4 (la versió de 4 GB), l'adaptador de corrent original de la casa Raspberry, dissipadors pels microxips de la placa, una targeta SD de 32 GB de capacitat d'alt rendiment (imprescindible per la Raspberry Pi) i una carcassa per la placa. Aquest lot l'oferia a un preu de 90 euros, un preu raonable tenint en compte els accessoris que en proveïa i comparant-la amb altres ofertes de segona mà.

[27] Amb la pandèmia de la Covid-19, les produccions de microxips van sofrir una important davallada en la seva producció. Amb les fàbriques de xips tancades, es donava prioritat als xips més potents, deixant per més endavant xips econòmics com el de la Raspberry. Com a conseqüència, es va aturar la producció d'aquestes plaques.

Un cop hem adquirit la Raspberry Pi 4, ara ens toca instal·lar-hi un sistema operatiu. Per això, hem de tenir en compte els requisits dels programes que volem instal·lar i també el fet que estem limitats a versions basades en ARM. És evident que haurem d'utilitzar un SO[28] basat en Linux, ja que Windows no suporta aquest tipus de plaques degut al seu baix rendiment.

La mateixa fundació Raspberry Pi ens posa a la nostra disposició un sistema operatiu Linux basat en Debian[29], el qual està espacialment dissenyat i optimitzat per utilitzar-se amb la seva placa. Aquest sistema operatiu s'anomena Raspberry OS i es pot descarregar de forma gratuïta des de la seva pàgina web[30].

Una altra opció és instal·lar la distribució Ubuntu[31], la qual també ofereix suport oficial per la Raspberry Pi 4. Tot i tenir suport oficial, aquesta distribució no està especialment optimitzada per la placa en qüestió. Per altra banda, gaudeix de molta més compatibilitat amb programes i llibreries de programació.

Vistes les dues principals opcions, decidim instal·lar Ubuntu, ja que la seva gran compatibilitat ens pot facilitar molt més la creació del projecte. Per fer-ho hem de descarregar-nos un programa específic[32] de Raspberry per poder instal·lar el SO dins de la targeta SD (la Raspberry no té disc dur com a tal)

Un cop instal·lat el SO, iniciem les configuracions bàsiques (usuari, contrasenya, idioma…) i ja podem utilitzar la Raspberry Pi 4 com a centre computacional. Encara falta, però programar i configurar els paquets que ens permetran poder ajudar als cecs.

[28] Abreviatura de Sistema Operatiu (SO).
[29] Sistema operatiu basat en Linux que utilitza un entorn gràfic (escriptori) basat en LXDE.
[30] https://www.raspberrypi.com/software/operating-systems
[31] Sistema operatiu basat en Linux que fa servir un entorn gràfic basat en GNOME.
[32] Raspberry Pi imager: https://www.raspberrypi.com/software/

8.2. Detecció d'objectes

L'objectiu d'aquest apartat és poder detectar tots els objectes de naturalesa general, sense cap coincidència més que el fet de situar-se al carrer. Per això farem servir un programa (amb finalitats didàctiques utilitzarem el terme IA, tot i que aquest està reservat a màquines que puguin superar el test de Turing, cosa que el programa que entrenarem no ho pot fer) el qual entrenarem mitjançant imatges per tal que pugui detectar tots els objectes que vulguem.

Primer de tot, ens cal saber que és realment això de l'Object Detection. Aquesta és una disciplina que podem emmarcar dins d'una disciplina més gran com és la Visió per Computador. Dins d'aquesta podem trobar altres tipus de programa que ens permeten d'igual manera que amb Object Detection treballar amb imatges i trobar patrons que es repeteixen en cada una. D'aquesta forma, s'entrenen per localitzar objectes (com bé el nom diu). Dins de la Visió per Computador també trobem la Classificació d'imatges i la segmentació d'imatges. La primera està programada per classificar les imatges que se li presenten segons criteris seleccionats, per exemple segons si apareix en la imatge un gat o si apareix un gos. Per altra banda, la Segmentació d'Imatges està programada per localitzar objectes amb els quals ha estat entrenat i, a més els delimita dins d'aquesta imatge. Situa exactament el que és un objecte, marcant el seu contorn. L'Object Detection es troba just enmig, dins d'una imatge detecta els objectes, però no en senyala els contorns, simplement l'emmarca dins d'una caixa amb el nom de l'objecte. Veiem una imatge per il·lustrar millor les diferències.

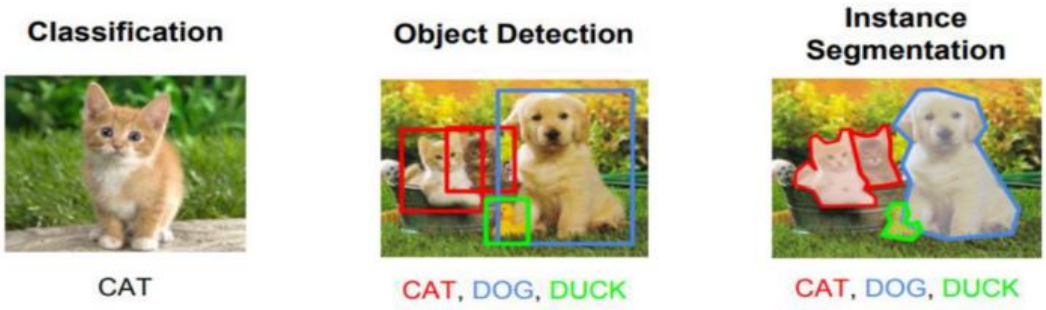


Figura 29. Per ordre d'aparició: Classificació, Object Detection i Segmentació d'Imatge
font: https://medium.com/ml-research-lab/what-is-object-detection-51f9d872ece7

L'elecció de la detecció d'objectes com a mètode d'identificació es basa en dos arguments molt senzills. El primer, la classificació d'imatges no ens és útil pel nostre projecte, ja que no serviria per detectar més d'un objecte dins d'una imatge, de fet no seria ni capaç de poder classificar-la. El segon és que la segmentació d'imatges és una detecció massa detallada. No ens interessa detectar exactament l'àrea de la imatge que ocupa un cert objecte, només ens interessa saber que aquell objecte està present. A més, aquesta segmentació detallada precisa de molts més recursos computacionals que no pas l'Object Detection. Tenint una placa limitada pel consum i la seva potència de còmput, no ens interessa posar-li més càrrega de treball, ja que aleshores el resultat seria menys satisfactori (pantalles negres, congelament de la imatge, apagades sobtades…)

La forma de dur a terme aquesta detecció d'objectes no és única, en el món de les ciències de computació s'han anat desenvolupant diverses formes de fer-ho. Cada una té els seus punts forts i punts febles i seran adients per situacions diferents. Aquestes diverses formes de dur a terme un mateix tipus de programa se'n diuen models i internament utilitzen llibreries (programació) i estructures (com s'organitzen les capes i les neurones) molt diferents entre si. Per sort per la comunitat informàtica, la majoria de models d'Object Detection són Open Source[33], per la qual cosa podrem accedir als models més desenvolupats del moment de forma lliure. Gràcies a això, podrem escollir i triar entre les diferents estructures que tenim disponibles i escollir les que millor s'adiguin al nostre projecte.

8.2.1. Elecció dels models

Si busquem una classificació general amb la qual poder dividir els diferents models d'Object Detection aquesta seria segons la llibreria principal que fan servir. Els principals models utilitzen TensorFlow o bé PyTorch.

[33] Open Source: Programes o materials que no són de domini privat, els desenvolupadors fan el material o programa públics per tal que tothom el pugui utilitzar. El seu ús és lliure, sempre que no infringeixis les normes de l'Open Source. En altres paraules, no pots treure profit econòmic d'aquest projecte.

TensorFlow és una llibreria creada i mantinguda per Google que se centra bàsicament a crear tècniques i programes per millorar l'àmbit del Deep Learning i el Machine Leraning. Tot i estar mantinguda per Google, la llibreria és Open Source, per la qual cosa qualsevol persona pot adoptar-la, crear els seus propis models i mantenir-la. Gràcies a això, TensorFlow ens ofereix una gran multitud de models els quals estan sempre actualitzats . D'entre les models que hi ha, podem trobar una gran varietat d'estructures que ens permetran balancejar entre rendiment i potència de còmput. Per contra, la seva implementació és molt més difícil, ja que requereix llibreries molt concretes, les quals a vegades porten a errors en el programa

A l'altre costat de la balança tenim la família de models YOLO[34]. A diferència de TensorFlow, aquests models no tenen a cap gran empresa darrere i es desenvolupa gràcies al treball de la gent que el vol desenvolupar. Per ser justos, cal dir que les primeres versions sí que complien els criteris anteriorment descrits, però les últimes versions d'aquest model estan programades de la mà d'Ultralytics, tot i que també segueixen mantenir-se gràcies a la comunitat que té al darrere. YOLO no utilitza les llibreries de TensorFlow així que utilitza les llibreries de PyTorch. Si destaca una qualitat d'aquests models és la rapidesa en què poden detectar i processar les imatges (temps d'inferència). Aquesta reducció del temps d'inferència ve donat per la forma en la qual analitza la imatge. A diferència dels altres models, YOLO divideix la imatge en petits trossos i cada un d'ells compta amb un programa individual que va detectant si hi ha alguna cosa en aquella casella en particular.

Un cop conegudes les dues grans branques de l'Object Detection, ara en cal escollir-ne un en concret. El model que escollim ha de detectar el millor possible els objectes seleccionats, però també ha de ser capaç de fer-ho en el menor temps. Tenir temps d'inferència elevats, farà que passi massa temps entre que es captura la imatge i la persona rep l'ordre per veu. Si tenim temps d'inferència molt grans, pot suposar un perill per la persona, ja que la informació que rebi ja estarà desactualitzada. Un altre

[34] Acrònim en anglès de You Only Look Once (Només Mires Una Vegada)

aspecte a tenir en compte és la senzillesa de la seva estructura i programació, perquè això ens permetrà poder manipular, si és necessari, tota la seva programació per afegir o treure característiques d'aquest model, tenint en compte que tenim un coneixement en Python de nivell baix-mitjà.

Primer, cal escollir un model en concret, el que millor ens convingui, basant-nos en els tests que s'han dut a terme en condicions iguals. És a dir, que han rebut les mateixes imatges, anotacions, un mateix període d'entrenament i en unes condicions de maquinari iguals (mateix processador, mateixa quantitat de memòria, mateixa targeta gràfica…).

Els resultats que ha obtingut el Blogger especialitzat en IA i Deep Learnig *Jonathan Hui*, amb una mostra d'alguns dels models són els següents:

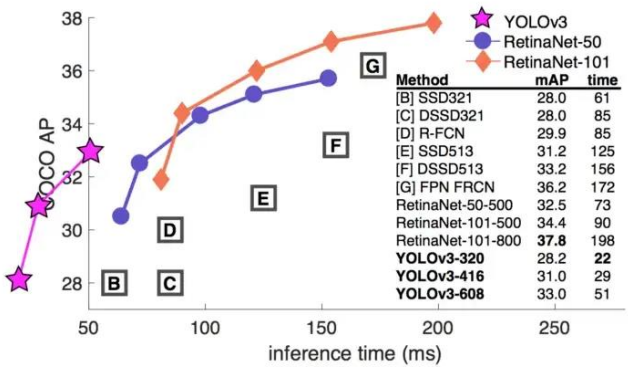


Figura 30. Comparació entre els models més destacats d'Object Detection.
font: https://jonathan-hui.medium.com/object-detection-speed-and-accuracy-comparison-faster-r-cnn-r-fcn-ssd-and-yolo-5425656ae359

Si observem la infografia, podem observar en primera instància una gràfica amb tres dels models avaluats. En l'eix d'abscisses trobem el temps d'inferència (com més a l'esquerra sigui, menor temps d'inferència, per tant, més ràpid anirà) i en l'eix d'ordenades trobem l'AP (Average Precision). En termes generals, el mAP (o AV) es refereix al rendiment que ofereix el model. Es basa en factors com la precisió amb què detecta els objectes, el percentatge de deteccions verídiques, el percentatge de falses, el percentatge d'objectes detectats com a negatius que eren positius i els

negatius detectats com a positius. En definitiva, és un valor que inclou molts aspectes que parlen sobre la capacitat del model en detectar els objectes.

El model que treu menys temps d'inferència és el YOLO (amb les seves variants YOLOv3-320, YOLOv3-416, YOLOv3-608), mentre que obté un màxim de 33 punts en AP. Comparat amb els altres models, podem veure que aconsegueix resultats similars en temps inferiors a altres models com el RetinaNet-50. Si bé el màxim d'AP l'aconsegueix el RetinaNet-101, el temps d'inferència que necessita per obtenir aquesta precisió és molt superior al demanat pel YOLO.

Segons aquesta gràfica, la millor opció per fer la detecció d'objectes seria utilitzar YOLO, ja que ofereix uns temps d'inferència molt baixos, mentre que també ens ofereix uns nivells de precisió acceptables (no perdem molta precisió en reduir el temps d'inferència). Ara bé, aquesta comparació s'ha fet amb el model v3 de YOLO, però actualment ja han sortit models del mateix YOLO molt més moderns i amb millors prestacions. Comparem les diferents versions per veure quina és la més adequada:

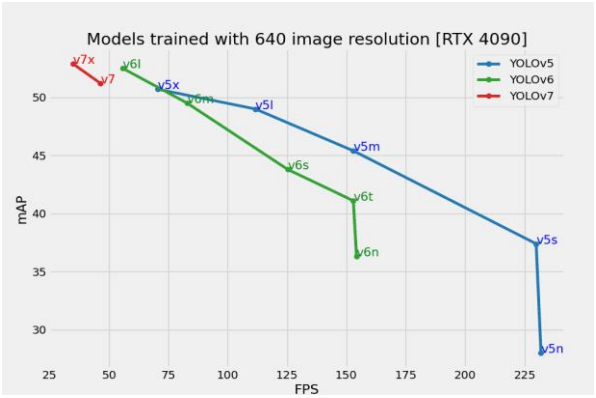


Figura 31. Comparació entre les tres últimes versions de YOLO (v5, v6, v7)
font: https://learnopencv.com/performance-comparison-of-yolo-models/.

En aquesta figura, amb mateixos indicadors que l'anterior[35] i també efectuada en iguals condicions entre ells, observem la comparació entre el model 7 (estrenat el juny

[35]Els FPS són el nombre de fotogrames que es poden mostrar en un segon, per tant, com més tinguem més ràpida serà la inferència.

de 2022), 6 (principis 2022) i model 5 (principis 2020). Cal destacar que els diferents punts que s'observen en cada model són els diferents nivells de complexitat que poden tenir cada un d'ells, parlarem d'ells més endavant. Si bé el model de YOLO anterior era el 3, hem de saber que aquest model està molt per sota en rendiment i prestacions que YOLOv5 i, a més ja és gairebé obsolet, ja que es va estrenar el 2018.

El model que obté major puntuació de mAP és YOLOv7 (amb una mica més de 50 punts), tot i que, a canvi, ens ofereix molt pocs fotogrames. YOLOv6 aconsegueix una puntuació bastant similar amb la seva versió més complexa, mentre que amb la seva versió més simple obté, aproximadament, un mAP de 36 punts, però també amb uns FPS bastant limitats[36]. El model més antic (el model 5) ens ofereix la quantitat més gran de fotogrames dels tres i també un mAP acceptable en les seves versions "m" i "s".

Vistos els resultats assolits per cada model, creiem que els models més indicats serien o el 6 (en la seva versió "n") o bé la versió 5 en la seva versió "s" o "n". Per desempatar la competició tindrem també en compte dos elements extres:

1. En programació és important esperar un temps prudencial un cop surt qualsevol mena de programa nou, ja que aquest pot contenir errors propis del llançament. Per tant, creiem que YOLOv6 encara no està prou provat com per ser estable i no donar problemes desconeguts.
2. El suport que té cada model també és rellevant, perquè davant de qualsevol inconvenient que la comunitat trobi (errors, mal funcionament, etc.) serà més fàcil que una comunitat gran o empresa puguin solucionar-ho que no pas una comunitat petita. En aquest cas, YOLOv5 disposa d'una empresa com ho és Ultralytics que s'encarrega de mantenir-lo, mentre que YOLOv6 només té una comunitat i el seu creador.

[36] Cal tenir en compte que la Raspberry Pi 4 no tindrà tanta potència com la que té l'ordinador en el qual s'ha entrenat aquests models. No hem d'esperar els mateixos fotogrames o temps d'inferència, hem d'esperar molts menys

8.2.2. Instal·lació dependències YOLOv5

Abans de poder començar amb el procés d'entrenament del programa, primer ens hem d'assegurar que aquest pugui executar-se en el nostre ordinador. A més de les dependències necessàries, també necessitarem instal·lar algunes llibreries opcionals, per aprofitar les característiques del nostre ordinador i disminuir el temps d'entrenament. És necessari instal·lar Python abans, però suposarem que aquest ja està instal·lat, per no allargar més l'explicació.

Primerament, hem de descarregar-nos tots els arxius que componen al YOLOv5. Per fer-ho només ens caldrà descarregar-los des de la pàgina de GitHub[37] de YOLOv5. Tenim vàries formes de descarregar-los, la més bona és obrir un terminal (sigui Windows o Linux) i executar el següent codi:

```
git clone https://github.com/ultralytics/yolov5
```

Aquest codi ens descarregarà tots els arxius necessaris per poder executar YOLOv5. El següent que hem de fer és instal·lar totes les llibreries que el programa necessita. En el cas de YOLO (qualsevol d'ells) és molt fàcil fer-ho, ja que totes les llibreries necessàries estan especificades en un arxiu anomenat requirements.txt. D'aquesta forma, no hem de perdre temps intentant instal·lar les versions correctes de cada llibreria. Només hem d'executar el següent codi per instal·lar-les:

```
pip install -r requirements.txt
```

Ara ja podríem iniciar un entrenament o una detecció amb els models preentrenats, però en el nostre cas, disposem d'una targeta gràfica bastant potent de la marca Nvidia. Les gràfiques d'aquesta marca estan també optimitzades per paral·lelitzar les tasques d'entrenament de tota mena d'intel·ligències Artificials. Per poder-la aprofitar

[37] GitHub és una plataforma d'allotjament de repositoris, on tothom pot penjar i compartir els seus arxius de programació: github.com

i disminuir el temps necessari respecte de fer-ho amb el processador, necessitem instal·lar les llibreries CUDA. Per això ens dirigim a la web de Nvidia per descarregar i instal·lar el CUDA toolkit[38]

Un cop acabi la instal·lació de CUDA, ja estarem llestos per començar a crear el nostre propi model.

8.2.3. Personalització del model

YOLOv5 ens permet utilitzar-lo sense haver d'entrenar-lo, mitjançant models preentrenats amb imatges i anotacions (dataset) creats per Microsoft. En el cas de YOLO, podem aconseguir models preentrenats amb el dataset COCO 2017, el qual conté milers d'imatges i anotacions de 80 classes d'objectes. No obstant això, aquestes 80 classes no són útils per a nosaltres, ja que incorporen objectes com "forquilles", "corbates" o "ganivets", objectes bastant prescindibles en la nostra detecció. Per aquest motiu, necessitem un dataset[39] personalitzat, on estiguin presents just els objectes que volem detectar.

En aquest punt tenim dues opcions. La primera és crear un dataset nosaltres mateixos i veure si aquest és viable. La segona consistirà a buscar si per internet existeix ja algun dataset que contingui els objectes que nosaltres necessitem.

Per dur a terme la primera opció hem de ser conscients de dues coses. La primera és que per crear un dataset hem de seguir un conjunt de normes o bones pràctiques que de seguida detallarem, de la mateixa manera que també necessitarem anotar i identificar en cada imatge els objectes que vulguem que detecti.

El conjunt de normes que hem de seguir per crear el nostre propi dataset són les següents:

- Hem de tenir una gran quantitat d'imatges de cada objecte a detectar, ja que si no el model no serà capaç de poder extreure les característiques de cada

[38]Pàgina de descàrrega: www.developer.nvidia.com/cuda-toolkit
[39] Dataset: Conjunt d'imatges amb anotacions on s'indica on es troba cada objecte.

objecte. De forma general, seria aconsellable tenir 1000 imatges de cada objecte (evidentment no és matemàtic, el nombre pot variar, però en total hauria d'haver-n'hi en gran quantitat).

- Les imatges que representin cada objecte han de ser fetes en diferents angles i condicions, ja que si no el model "s'acostumarà" a trobar l'objecte només en una posició i llavors no el podrà detectar en altres posicions (al carrer poden estar en multitud de posicions i condicions lumíniques diferents).

- Hem d'afegir imatges de variants d'un mateix objecte. Per exemple, si volem detectar pilones, haurem de tenir en compte que hi ha moltes mides i colors. Haurem d'incloure la quantitat més gran possible de variants per assegurar-nos que ho detectarà tot com a pilona.

- Si en una imatge hi ha més d'un objecte a detectar s'han d'anotar tots els objectes, ja que si no és probable que confonguem al model i que no aprengui tan de pressa.

- Caldrà afegir imatges distorsionades o modificades (que no es mostri l'objecte sencer o en condicions que no serien normals). D'aquesta forma aconseguirem que es fixi més en la forma de l'objecte. Alguns exemples serien rotar la imatge (vertical i o horitzontal) o bé invertir els colors o transformar-la a blanc i negre.

- Es podran afegir imatges sense cap objecte (anomenades background), però en poca quantitat.

Tenint en compte tots aquests consells, ja podríem passar a crear el nostre dataset:

En primer lloc, hem d'obtenir les imatges dels objectes que vulguem detectar[40]. Per fer-ho podríem fer-les nosaltres mateixos, o podríem descarregar-les d'internet

[40] Consultar la llista de la pàgina 66

mitjançant algunes llibreries de Python que descarreguen imatges d'internet de forma automàtica. En qualsevol cas, un cop ja tinguem les imatges recopilades, és el moment de començar a anotar-les. Per fer-ho podem optar per instal·lar un programa anomenat LabelImg[41] o bé optar per pujar totes les imatges al núvol i treballar amb plataformes web de gestió de datasets. En el nostre cas, creiem que és molt més senzill treballar al núvol, ja que presenta moltes facilitats que no incorporen programes com LabelImg. A part dels avantatges i facilitats que pot oferir un núvol, treballar amb certa quantitat d'imatges de forma local pot arribar a fer-se una feina complicada (llargs temps de càrrega, de traspàs d'informació entre unitats d'emmagatzematge) i fins i tot impossible si no disposem de molt espai d'emmagatzematge (com és el nostre cas).

Nosaltres hem optat per pujar totes les imatges al núvol de Roboflow[42], ja que segons el nostre punt de vista és una de les plataformes més intuïtives que es dediquen només a l'edició de datasets. Tot i ser una plataforma gratuïta (com moltes d'altres) ens ofereix una quantitat limitada d'imatges que podem pujar al núvol. En aquest cas ens deixa pujar un màxim de 8000 imatges, més que suficient per poder entrenar el nostre model.

El següent pas lògic que s'hauria de dur a terme seria el d'anotar els objectes que hi ha en cada imatge. Aquest pas consisteix a anar enquadrant els objectes que vulguem detectar i anar indicant a quina categoria pertanyen. Les categories que hem d'enquadrar són aquelles dels objectes que ens interessa detectar[43]. Veiem un exemple:

[41]Es pot descarregar des del següent enllaç: www.github.com/heartexlabs/labelImg
[42] Accés: www.roboflow.com
[43]Consultar llista pàgina 66

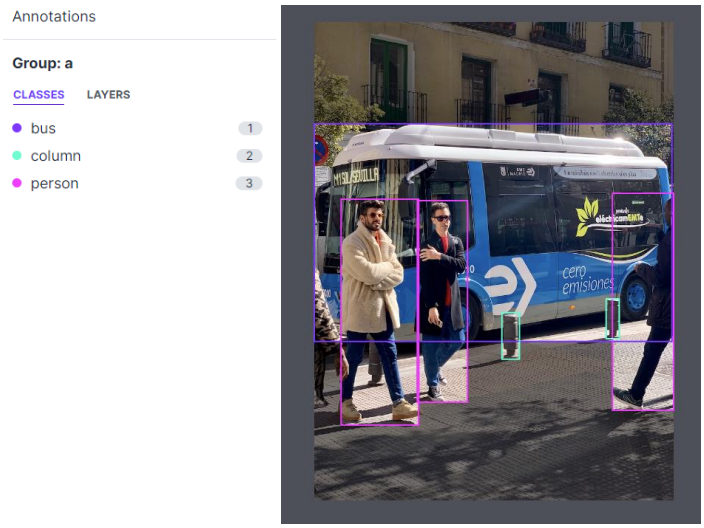


Figura 32. Procés d'anotació de les imatges

Font: Captura pròpia

En la figura 32 podem observar el procés que s'ha seguit per anotar una d'aquestes imatges d'un suposat dataset. Com podem apreciar, cal enquadernar els objectes que vulguem detectar i relacionar-los amb la categoria d'objectes al qual pertanyin. En aquesta figura, apareixen les categories a la columna esquerra i en aquest cas només podem trobar-hi les categories "person", "bus" i "column", relacionades amb els objectes de la imatge que compleixen com a objectes de la categoria.

Tot i semblar una tasca fàcil, cal remarcar que no ho és, ja que has de repetir el procediment tants cops com imatges tinguis. En el cas de tenir-ne el màxim admès per Roboflow (8000), el procés pot costar-te molt temps. Per aquest motiu, nosaltres vam intentar buscar, com a última opció abans d'haver d'anotar 8000 imatges, un dataset que ja estigués creat i anotat el qual inclogués, més o menys, les categories que nosaltres volíem.

El resultat el vam trobar, altre cop, en un repositori de la pàgina web GitHub, on un usuari ha penjat un dataset amb un nombre bastant extens d'imatges conjuntament amb les seves anotacions. El resultat el podeu trobar a la web citada en els repositoris GitHub a la part final del document amb el nom "Obstacle-dataset OD".

Tot i ser un dataset ja preparat, aquest no incorpora tots els elements que hem mencionat al principi. Si bé la gran majoria sí que els inclou, n'hi ha d'altres que n'haurem de prescindir. Aquesta és la llista dels objectes que serem capaços de detectar:

- Senyal de Stop
- Persones
- Bicicletes
- Bus
- Camió
- Cotxe
- Moto
- Cons reflectants
- Escombraries
- Pilones
- Esferes de ciment
- Pals (en general de ferro, podria ser que acabés detectant fanals i altres objectes verticals com a pal)
- Gos
- Tricicle (referit al vehicle motoritzat de tres rodes)
- Boca d'incendis

En total, serem capaços de detectar fins a 15 classes d'objectes. Segons el nostre punt de vista, mancaria poder detectar arbres, bancs, jardineres i vegetació en general, ja que van ser dels objectes que Anna Morancho ens va suggerir que més els molestaven quan anaven pels carrers. Per contra, obtenim nous objectes com les boques d'incendis, els cons reflectants, les esferes de ciment (molt presents al nostre poble), escombraries i senyals de Stop.

En descarregar el dataset, el pengem directament a Roboflow per comprovar si les anotacions no s'han mogut o desconfigurat i ja podrem guardar el dataset per poder exportar-lo. Abans, però, ens demanarà si volem modificar i aplicar algunes transformacions a les nostres imatges. Com hem vist en la llista anterior un dels consells era modificar les imatges i dona'ls-hi altres orientacions i colors. És just en aquest moment on podem fer-ho. Cal destacar que un cop fetes les transformacions s'han de seguir mantenint les imatges originals. Aquest procés es pot dur a terme a la secció "Augmentation" de Roboflow. En el nostre cas, hem decidit afegir imatges en blanc i negre, imatges girades horitzontalment i imatges amb espais en negre (requadres en negre). Tots aquests efectes que afegim contribuiran a fer el model més robust i que detecti objectes encara que estiguin mig tapats (per això els espais negres en les imatges) o bé per fixar-se més en les formes de l'objecte i no pas en el color d'aquesta (per això transformem algunes imatges a blanc i negre). La rotació de la imatge també ajuda al fet que el model es fixi més en la forma.

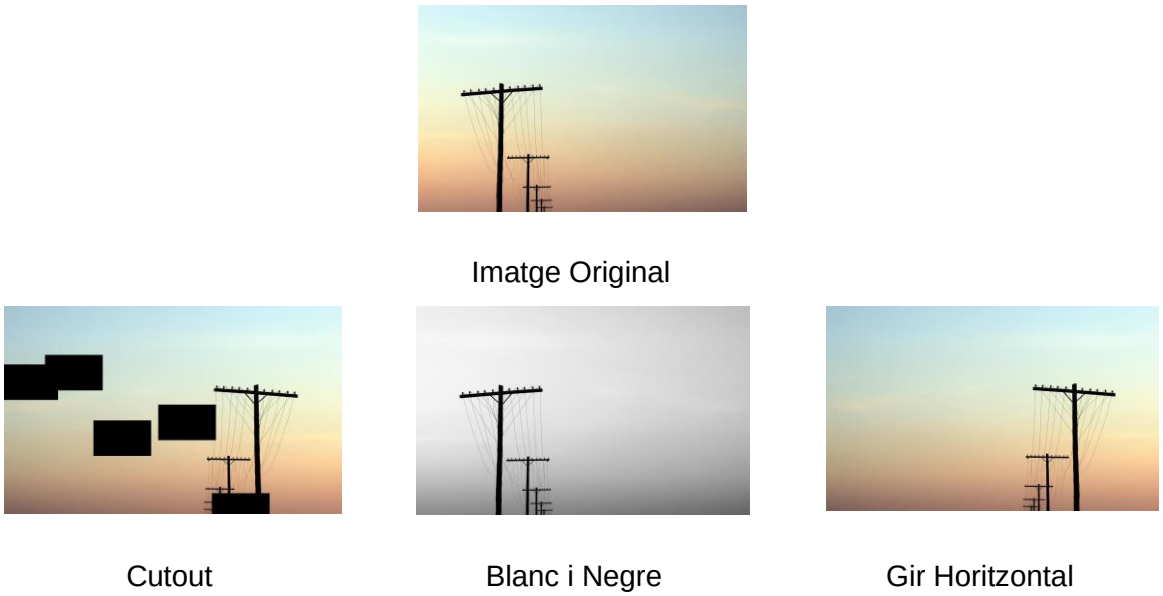


Figura 33. Modificacions que apliquem a les imatges del dataset.
Font: Imatges del dataset modificat

Després d'aplicar les transformacions, el descarreguem en el format YOLOv5 (en cada format canvia l'estructura de la carpeta que et descarregarà) i només ens

quedarà comprovar que l'estructura de l'arxiu de Roboflow descomprimit i incorporat dins de la carpeta que conté YOLOv5 coincideix amb el següent:

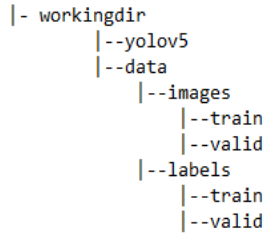


Figura 34. Estructura que hauria de tenir el nostre directori amb YOLOv5.
Font: https://towardsdatascience.com/yolo-v5-object-detection-tutorial-2e607b9013ef.

Dins de l'arxiu que ens descarreguem, hauríem de trobar tres carpetes i un arxiu anomenat data.yaml . Les tres carpetes corresponen a les divisions que Roboflow fa automàticament per dividir les imatges. La primera carpeta és train i en aquesta es troben totes les imatges (aproximadament un 70% del total) que el model utilitzarà per entrenar-se a ell mateix. La segona carpeta és val (equival a un 20% de les fotografies totals) seran les imatges que el model farà servir per avaluar-se després de cada epoch (cicle d'entrenament). L'última carpeta és la "test" i la seva funció és contenir totes les imatges que el model necessitarà per validar els resultats quan nosaltres li ho diguem. Per últim, tenim l'arxiu "data.yaml", en aquest arxiu, el model troba les direccions de dins l'ordinador on es troben les tres carpetes.

El següent pas a fer és molt més senzill. Consisteix a escollir quina estructura de YOLOv5 volem entrenar. Depenent de l'estructura que escollim, obtindrem uns resultats o uns altres, ja que variarà el nombre de "neurones" (unitats d'aprenentatge) que tindrà el model. Com més neurones tingui, millors resultats assolirem, però per contra aconseguirem un temps d'inferència molt superior (recordem que volem el menor temps d'inferència possible) i també requerirà més potència de còmput, la qual ja tenim limitada per la Raspberry Pi 4. Aquestes estructures venen determinades per l'arxiu .cfg que podem descarregar amb YOLOv5. Tot i ser un arxiu amb només paràmetres i variables, sense programació en si, aquests valors acabaran decidint l'estructura que farà servir el programa en entrenar. YOLOv5 disposa de les següents estructures:

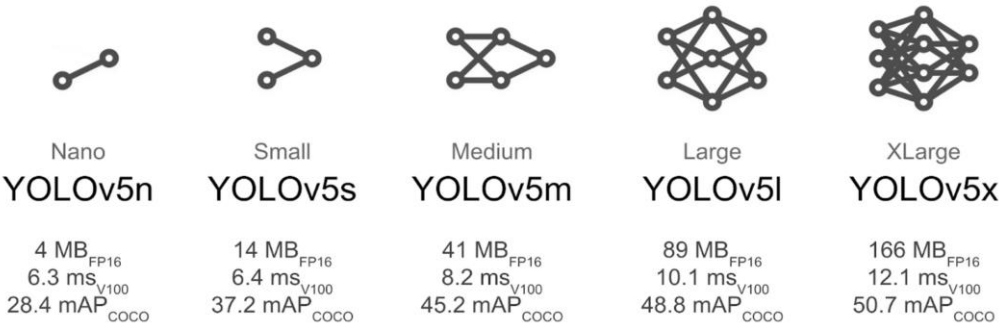


Figura 35. Els 5 models de YOLOv5 que hi ha actualment.

Font: https://machinelearningknowledge.ai/introduction-to-yolov5-object-detection-with-tutorial/.

Si ens referim al nostre projecte, podem descartar com a mínim dos dels models, que són el YOLOv5l i YOLOv5X, ja que la seva estructura és la més complexa i els temps d'inferència són bastant elevats. Podem acabar decidint-nos per un tot mirant el rendiment que obtenen en el mateix dataset que la gràfica anterior (COCO 2017):

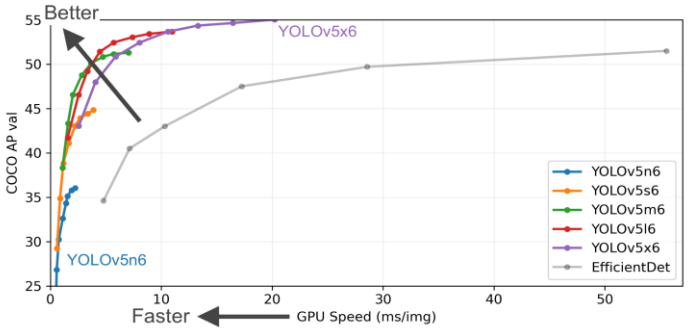


Figura 36. Comparació entre els models de YOLOv5 i EfficientDet

font: https://towardsdatascience.com/yolov5-compared-to-faster-rcnn-who-wins-a771cd6c9fb4.

En aquesta altra gràfica podem observar el rendiment (mAP) que tenen els models de YOLOv5. En aquest cas, ja hem descartat la variant "x" i "l" , per ser massa pesades computacionalment. Analitzant la situació que ens planteja la gràfica sembla que les versions més interessants pel seu temps d'inferència són la "s" i "n". Si investiguem per internet, podem veure com la majoria de pàgines web[44] on entrenen un model per la Raspberry Pi 4 de YOLO utilitzen la versió "s", ja que ofereix bon rendiment i poca necessitat de computació. No obstant això, segurament no contemplen la versió "n",

[44] Johnston, Jordan (consultat 5 desembre 2022). jordan-johnston271.medium.com. Tutorial: Running YOLOv5 Machine Learning Detection on a Raspberry Pi 4. https://jordan-johnston271.medium.com/tutorial-running-yolov5-machine-learning-detection-on-a-raspberry-pi-4-3938add0f719

perquè aquesta és molt recent. Així, creiem que el més convenient és entrenar la “s” principalment i després, si tenim temps, entrenar la “n” per veure quina s’adapta millor a la nostra situació.

Abans de passar a l’entrenament, també cal saber que no entrenarem el model des de zero. Ultralytics ens proporciona en el seu repositori uns models preentrenats amb milers de fotografies provinents del dataset COCO, els quals ens serviran de base pels nostres models. Aquesta operació s’anomena “transfer learning” i consisteix en només entrenar les dues últimes capes del model. Les altres es mantenen igual (tècnicament es coneix aquest procediment com a “freezing layers”), ja que són les que han après a buscar certes correlacions entre píxels per detectar objectes. A més, si no féssim servir la tècnica del “transfer learning”, necessitaríem encara més imatges per obtenir un resultat decent. En cada variant de YOLOv5 s’ha d’entrenar amb un model preentrenat de la mateixa variant. Ja hem descarregat els models preentrenats amb el mateix repositori del YOLOv5.

8.2.4. Entrenament

Ara que ja tenim tots els arxius i llibreries necessàries, ja ens podem dedicar a entrenar al nostre propi model. Com ja hem dit anteriorment, disposem de targeta gràfica dedicada, per la qual cosa el procés serà una mica més ràpid que no pas si ho féssim amb el processador únicament. Cal destacar que sempre quedaria l'opció d’entrenar-la amb el servei de Google Colaboratory (al núvol). Veiem, però com entrenem al nostre model.

Per iniciar el procés d’entrenament és molt senzill, només hem de cridar l’arxiu “train.py” que ens inclou la carpeta del model i donar-li totes les direccions dels diferents arxius que hem anat creant. La següent línia de codi resumeix tot el procés d’inici:

```
python train.py --weights
C:/Users/flocj/Desktop/yolo/yolov5backup/runs/train/yolov5s_V1_04/weight
```

```
s/yolov5s1.0.pt --data ./data/data.yaml --cfg ./models/yolov5s.yaml --
img 416 --epochs 300
```

Veiem que li estem dient amb aquesta línia de codi:

- `python` Amb aquesta part del codi estem indicant a l'ordinador que he d'executar l'arxiu train.py mitjançant el llenguatge Python
- `train.py` És el nom de l'arxiu, necessari per indicar a l'ordinador quin arxiu obrir
- `--weights` Indiquem la direcció del model preentrenat que ens proporciona Ultralytics amb YOLOv5 dins de l'ordinador.
- `--data` Indiquem a quina direcció dins de l'ordinador el programa pot trobar l'arxiu on s'especifica on estan les carpetes de "train", "val" i "test", on estan ubicades les imatges i les anotacions per entrenar.
- `--cfg` Li indiquem on es troba el fitxer .yaml on definim l'estructura que tindrà el model (en el nostre cas l'arxiu serà yolov5s.yaml, ja que farem servir la versió "s").
- `--img` Aquest valor fa referència a quina mida (en píxels) ha de transformar la imatge original per poder-se entrenar. Imatges amb mides molt grans seran més detallades, però costaran més d'entrenar, mentre que imatges molt petites serà molt difícil trobar algun objecte. Cal tenir en compte també quina resolució utilitzarà el model per fer inferència, ja que si l'entrenem amb una resolució molt gran, però fem una inferència amb una imatge amb poca resolució no serà capaç de trobar res. En el nostre cas pretenem fer inferència amb imatges de 416 píxels, així que l'entrenarem amb la mateixa resolució.
- `--epochs` Indiquem quants cicles d'entrenament ha de fer.

Els resultats que ens entrega el model mentre entrena són els següents:

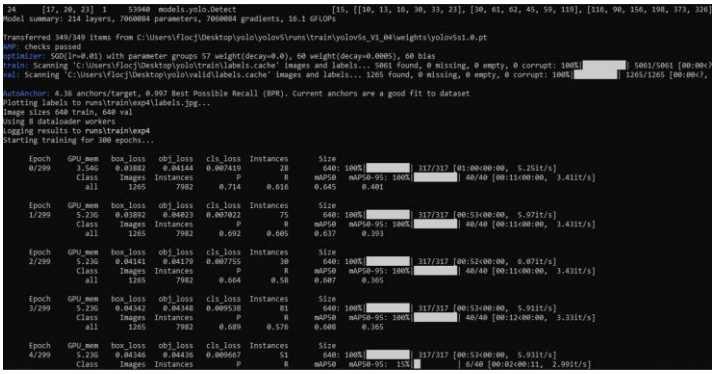


Figura 37. Missatges que ens mostra el programa mentre està entrenant.

Font: Captura Pròpia

8.2.5. Avaluació del model

Després d'haver entrenat el model hem de mirar com és de precís en identificar els objectes en imatges completament noves pel model. Aquestes imatges són les que hem guardat en la carpeta "train" i ara ens serviran com a marcadors per saber si estem obtenint un bon model.

El primer mètode que podem utilitzar és el mètode de passar-li les imatges de la carpeta "train" com a entrada i valorar nosaltres mateixos la imatge resultant. Aquestes són algunes de les imatges que hem extret:

Figura 38. Exemple d'imatge que dona com a sortida el nostre model

font: captura pròpia

Figura 39. Exemple d'imatge que dona com a sortida el model.

Font: captura pròpia

Com es pot apreciar en la figura 38, el model és capaç de detectar múltiples objectes en una imatge de forma correcta. No obstant això, podem observar també com la seva precisió no és molt bona en la figura 39. Aquest és un exemple de confusió del model, que ens indica que hi ha objectes que no acaba d'entendre. Per saber exactament quins objectes no ha après i quina és la precisió global, cal executar un altre codi de Python.

Per avaluar el model de forma objectiva hem d'executar l'arxiu anomenat "val.py". Aquest s'encarregarà d'anar monitorant tots els resultats que processi el nostre model i d'ajuntar els resultats en forma de gràfiques. Aquest és el codi que hem de posar:

```
python val.py --weights
C:\Users\flocj\Desktop\yolov5mod\runs\train\yolov5s_V1_04\weights\yolov5
s1.0.pt --data C:\Users\flocj\Desktop\yolov5mod\data\data.yaml --img 416
```

Veiem el que li estem dient a l'arxiu val.py:

- `--weights` Indiquem en quina direcció es troba el model acabat d'entrenar
- `--data` Direcció on es troba l'arxiu data.yaml, on indica la direcció de la carpeta "test", d'on traurà les imatges per avaluar.
- `--img` Mida de les imatges un cop entren en el model. YOLOv5 té una primera capa capaç de reduir la imatge a una resolució indicada

Un cop ha acabat l'avaluació ens guardarà un seguit de gràfiques que ens permetran saber diversos aspectes del nostre model de forma objectiva.

Confusion matrix:

Una de les principals gràfiques que ens dibuixa el model és la "Confusion Matrix". En termes generals, ens està dient quina precisió té el model en detectar cada categoria. Ens indica quin tant per cent de les vegades el model ha predit correctament la categoria i quin tant per cent de les vegades l'ha confós amb alguna altra categoria. Al mateix temps, també ens permet obtenir unes dades concretes que necessitarem en les següents gràfiques de forma molt senzilla. Aquests són els conceptes de "True Negative", "True Positive", "False Positive" i "False Negative". A part de totes les categories del model, s'hi afegeix una altra anomenada "background", aquesta categoria representa la identificació errònia de parts del fons (que no ens interessen) com objectes del nostre dataset o viceversa.

Per poder entendre de forma més fàcil aquesta gràfica, agafarem l'exemple d'una de les categories i l'analitzarem de forma individual. Cal remarcar que les categories posicionades en fila corresponen a la categoria real de l'objecte, mentre que les disposades en columna corresponen al que el model ha predit.

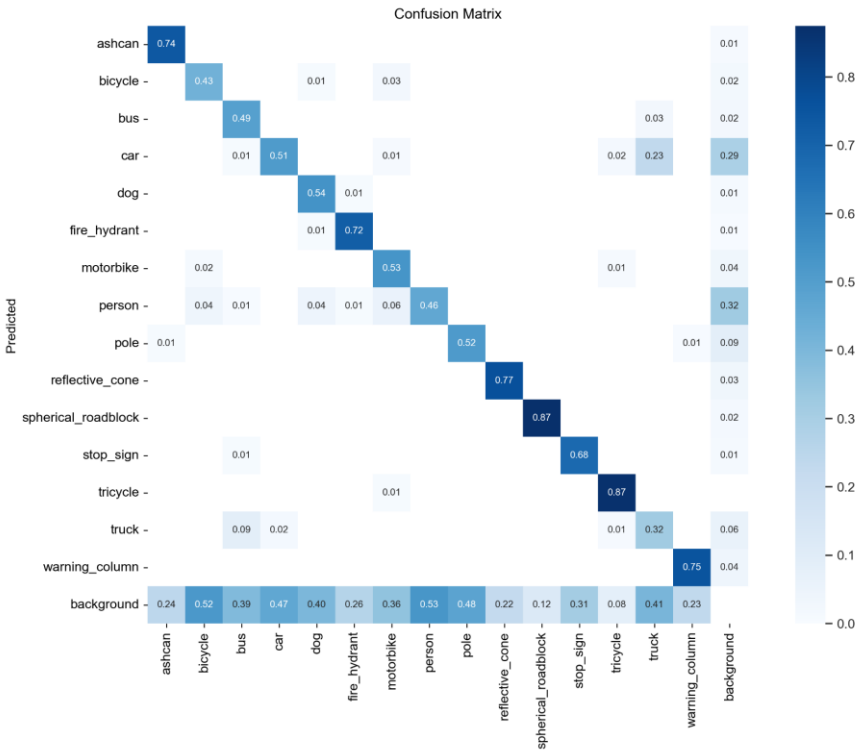


Figura 40. Confusion matrix del model YOLOv5s entrenat

font: producció pròpia

Fixem-nos ara en la categoria de l'objecte real (la de l'última fila) "bus". Podem observar com de cada vegada que el model havia de predir "bus", aquest només l'ha predit correctament un 51% (a la gràfica 0,51) de les vegades. Això ho sabem perquè la categoria de bus de la columna de noms (predit) coincideix amb el valor real de "bus" de la fila de noms (realitat) a aquests encerts els anomenem "True Positives". Si continuem analitzant aquesta columna, podrem veure que un 9% de les vegades, el model ha predit un camió (truck), quan en realitat era un "bus". D'aquest tipus de prediccions se les anomena "Fals Negative" (el model diu que ni hi ha l'objecte quan aquest sí que hi és). Més amunt trobem un altre "Fals Negatiu", ja que el model diu que no hi ha "bus", però sí que hi ha "stop_sign", quan en realitat hi ha "bus". Això li succeeix un 1% de les vegades que ha de detectar "bus". El mateix li passa amb "person", un 1% de les vegades que ha de sortir com a resultat "bus", el model diu que no hi ha "bus", però que hi ha "person". Cal tenir en compte que el resultat seria al revés si ho observéssim des del punt de vista de "person". Des d'aquest, podríem veure com el "False Negative" de la categoria "bus" es converteix en un "False

positive". En altres paraules, el model detecta "person" quan hauria de detectar "bus". Per tant, podem assumir que aquestes afirmacions varien segons l'objecte del qual avaluem la precisió. Per resumir:

- True Positive: Són totes aquelles vegades que el model encerta la categoria que ha de trobar.
    - Exemple: Avaluem "bus". El model ens indica que l'objecte és un bus i realment ho és.

- True Negative: Són totes les vegades que el model no detecta un objecte i la realitat tampoc és aquest objecte.
    - Exemple: Avaluem "bus". El model ens diu que l'objecte és un "truck" i és un "truck".

- False Positive: Les vegades que el model indica que existeix l'objecte, però que aquest no es correspon amb la realitat
    - Exemple: Avaluem "bus". El model ens indica que l'objecte és un "bus", però l'objecte real és un "car".

- False Negative: Cada vegada que el model diu que no hi ha un objecte (podria dir que n'hi ha un altre) però que realment el primer objecte sí que existeix.
    - Exemple: Avaluem "bus". El model ens indica que l'objecte és "dog", però l'objecte és en realitat "bus"

Un cop ja sabem com interpretar la gràfica, ressalta a simple vista que hi ha certes categories les quals el model no acaba d'apreciar les diferències. En concret preocupa bastant el "Fals Positive" entre "truck" i "car", ja que aquest succeeix un 23% de les vegades que el model ha de detectar la categoria "car". Els altres False Positive que podem trobar no són tan preocupants com el de "truck"; tot i això, també val la pena comentar-los. La següent categoria amb menys precisió és "person" que,

curiosament, té percentatges bastant distribuïts en moltes categories. Entre elles, podem trobar “bicycle”, “bus”, “dog”, “fire_hydrant” i “motorbike”. Algunes confusions que té poden arribar a tenir certa explicació lògica, com és el cas de “motorbike”, “dog”, “bicycle” i fins i tot “bus”. L’error aquí deu estar en el fet que aquestes categories gairebé sempre apareixen al costat o a les proximitats de la categoria “person”, a causa de l’ús habitual que les persones fem d’aquestes categories. Així, el model acaba entenent que aquestes categories acaben formant part de la categoria “person”. No obstant això, la confusió amb la categoria “fire_hydrant” no té una explicació lògica com les altres confusions. La solució passaria segurament per introduir més imatges de “fire_hydrant” al dataset, però per la petita quantitat d’error que provoca, no ho considerarem un error greu.

Un altre fet que també podem observar en aquest gràfic és la gran quantitat de confusió (False Negative) que té el model amb el “background” (fons). El model no és capaç moltes vegades de distingir les categories del que és el fons de les imatges i no acaba assenyalant els objectes del dataset que hi ha presents (els confon amb el fons). Una possible explicació al problema seria la no introducció de fotografies sense cap categoria dins del dataset[45], per tant, la solució exigiria introduir una quantitat considerable de fotografies sense cap categoria present.

Precision-Recall Curve:

La següent, i última, gràfica que també és interessant mirar és la gràfica de “Precision-Recall”. Abans caldrà entendre el concepte de “recall” i precisió, ja que de no fer-ho no podríem saber què estem observant:

Quan ens referim al “Recall”, estem referint-nos a quina quantitat de totes les deteccions que haurien de ser positives el model és capaç de detectar. En termes matemàtics, s’obté de la següent operació:

[45] Tot i haver mencionat aquest pas com a un dels termes més importants per la creació d’un dataset, cal dir que el dataset que vam poder obtenir no contemplava aquesta possibilitat.

$$\text{Recall} = \frac{TP}{TP + FN}$$

On TP representa la quantitat de “True Positives” que té una certa categoria del dataset i FN representa la quantitat de “False Negatives” total que té la mateixa categoria.

Per altra banda, quan ens referim a “Precision” ens estem referint a quina proporció de totes les deteccions positives d’una classe són realment verídiques. S’obté de la següent operació:

$$\text{Precision} = \frac{TP}{TP + FP}$$

Ja per acabar, podrem observar un valor anomenat “Average Precision”, aquest s’extreu, de forma simplificada, de fer la integral de la funció Precision-Recall de cada categoria. En altres paraules, correspon a l’àrea que descriu la funció Precision-Recall de cada categoria respecte dels eixos cartesians. Aquest valor és un molt bon indicador per saber el rendiment que obté un model perquè, com és lògic, reuneix en un sol valor els valors de “Recall” i “Precision”. A més, també és una molt bona forma de comparar el rendiment de cada model. Matemàticament es definiria de la següent forma:

$$\int_0^1 p(r)dr$$

La mitjana ponderada de totes les “Average Precision” de cada classe dona lloc al mAP, que seria la xifra que ens indica el rendiment general del model en el dataset concret. Hem de fixar-nos també en el número que acompanya al mAP. Aquest ens indicarà el valor a partir del qual el programa d’avaluació dona com a bo una predicció del model. Aquest valor és el resultat de calcular el “Intersection over Union” IoU. De forma resumida, es tracta de comparar el rectangle que el model posa per delimitar el què és un objecte amb el rectangle que nosaltres li proveïm amb les anotacions. Es pot calcular amb la següent fórmula:

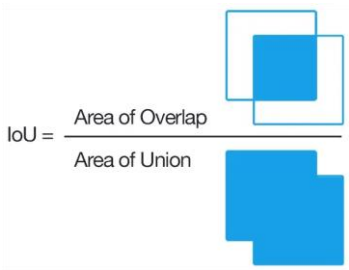


Figura 41. Formulà gràfica de com es calcula IoU

font: https://towardsdatascience.com/breaking-down-mean-average-precision-map-ae462f623a52

D'aquesta forma, si el mAP va acompanyat per un 0.5, voldrà dir que estarem donant com a bons tots els resultats en els quals hagi encertat el 50% de l'àrea total. En canvi, si va acompanyat de 0.95, voldrà dir que només donarà com a bons els resultats que pràcticament imitin els de les anotacions. De forma general, s'utilitza el mAP@0.5.

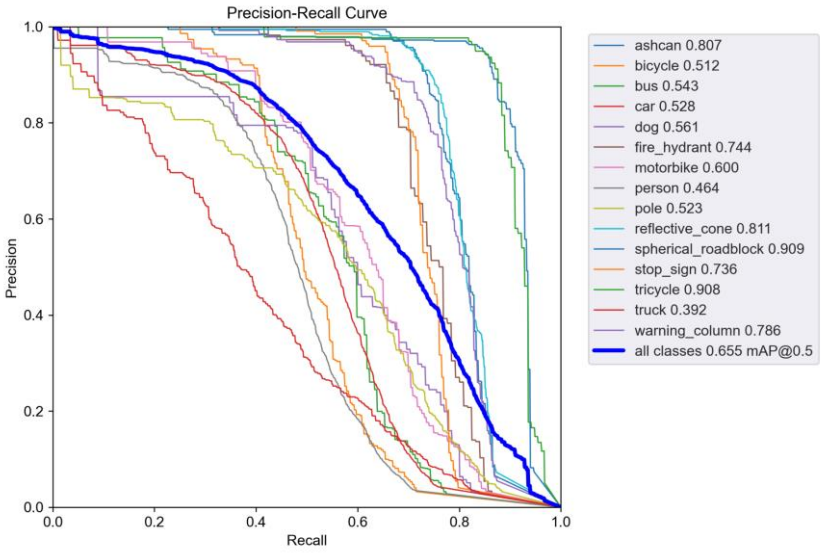


Figura 41. Gràfica Precision-Recall del model YOLOv5s1.0[46]

font: Gràfica generada pel model.

detect.py

Un cop plantejats els problemes, cal buscar una solució que s'adigui a les nostres necessitats. En un principi, el primer que vam pensar va ser en fer ús d'una llibreria que ens permetés executar dues funcions a la vegada, anomenada `asycio`, però les seves estructures eren complicades i difícils d'entendre (es podria catalogar com un

[46] Nom que li hem donat al primer model de YOLOv5s que hem entrenat.

nivell de Python avançat). Després de donar-hi moltes voltes vam arribar a la conclusió que el millor era utilitzar diferents programes, un per la detecció i un altre per reproduir el so. El programa de detecció, llavors, escriuria en un arxiu anomenat data.txt, situat a la mateixa carpeta que el programa detect, les categories que havia detectat en cada fotograma inferit. Això ho faria mitjançant una funció que ja incorpora Python que és la funció `open()` aquesta funció obrira el fitxer en un mode de només escriptura i escriuria les dades que conté la variable just abans que s'acabi la iteració del **for** en la que està situada. Aquest és el resultat:

```
with open("data.txt", "w") as data:
                data.write(",".join(uu))
                print("he escrit en data")
```

On `"data.txt"` és l'arxiu on es guardarà les variables, `"w"` el valor que necessita el programa per obrir l'arxiu en mode només escriptura, data.write() la funció que escriu les dades, `(",".join(uu))` la part on agafem les dades de la variable uu (la que conté les dades de cada cicle) i les unim en un string per tal que GTtS ho detecti correctament i `print("he escrit en data")`, un component per comprovar que realment està funcionant correctament.

main.py

Per altra banda, tenim un altre arxiu Python que es dedica a llegir les dades que l'arxiu detect.py escriu en el document data.txt. El seu procediment és senzill, definim una funció tts, la qual conté un **`while`** (bucle fins que la variable a deixa de ser **`True`**) infinit (fet a propòsit per tal que la funció no deixi mai de funcionar a no ser que l'usuari la pari). Dins d'aquest **`while`**, obrim l'arxiu data.txt només de lectura i guardem el que hi ha en la variable arg, la qual donem com a valor a la funció Speech.speak que executarà el procés de conversió a àudio. En el cas que no hi hagi res escrit en el document, assumirem que no s'ha detectat res i per això creem una estructura condicional (**`if`**) que si detecta que arg no conté cap valor li adjudica el valor `"no detection"`. Aquí el resultat:

```
    while a == True:
        with open("data.txt", "r") as data:
            print("open")
            arg=data.read()
            if arg == " ":
                arg = "no detection"
            Speech.speak(arg)
```

Speak.py

Per últim, ens queda definir el procés que segueix l'arxiu encarregat de transformar el text en veu. Aquest s'anomena Speech i a dins té una funció anomenada speak. En aquesta, donem valor a una variable anomenada tts amb els paràmetres que volem que tingui. Aquests paràmetres seran text, que serà el text a transcriure i lang, que serà la llengua en la qual li estem passant el text. Com que les anotacions estan en anglès, deixarem la llengua en anglès per millor comprensió. Un cop ha transformat el text a àudio, passa aquest àudio a la llibreria BytesIO, la qual funcionarà com un mena de disc dur intern del programa. Per reproduir l'àudio utilitzem la llibreria pygame, la qual necessita diverses línies de codi per poder-se inicialitzar. Aquesta llibreria reproduirà l'arxiu de veu guardat per BytesIO. Pel bon funcionament de la llibreria pygame, hem d'afegir un temps d'espera, si no ho féssim no reproduiria l'arxiu de veu. El resultat:

```
def speak(cls, text):
        mp3_file_object = BytesIO()
        tts = gTTS(text, lang='en')
        tts.write_to_fp(mp3_file_object)
        pygame.init()
        pygame.mixer.init()
        pygame.mixer.music.load(mp3_file_object, 'mp3')
        pygame.mixer.music.play()
        time.sleep(1)
```

En tots tres casos, es pot trobar la programació completa a l'Annex 2

8.3. IA Stairs Detection

8. 3. 1 Definició del model

Una de les carències de l'apartat anterior és que el programa fins ara només és capaç de detectar objectes treballant en imatges planes. Però a la realitat del nostre dia a dia hi ha obstacles que només amb 2D no podem localitzar, com ara les escales. És per això que necessitem poder treballar amb la tercera dimensió: la profunditat. Una càmera usual no és capaç de capturar aquesta dada, però les anomenades *depth cameras* sí. La Kinect entra dintre d'aquest grup, aquesta és una de les raons per la qual hem decidit treballar amb ella.

El següent objectiu és la segona funció a implementar: La detecció d'escales. Volem que l'usuari pugui saber a quina distància es troben les escales que està enfocant, si aquestes són ascendents o descendents i el nombre d'esglaons que tenen. Tota aquesta informació farà més senzilla la convivència dels cecs amb aquest tipus d'obstacle.

Cal recalcar que el disseny de les escales sempre varia en funció de la regió on ens trobem, si aquestes són d'un edifici públic o privat, si han sigut construïdes per una empresa multinacional o petita, etc.

8.3.2. Funcionament

El data emmagatzemat en aquest programa són els núvols de punts[47] generats aleatòriament per la Kinect. Cada píxel capturat conté la seva localització 3D respecte a la càmera i tota la informació RGB (nota al peu). Tenint en compte el tipus de data utilitzada com a input, utilitzarem com a framework Robot Operating System (ROS) i Point-Cloud Library (PCL) com a llibreria. Aquesta última ha sigut desenvolupada de cara al tractament computacional de núvols de punts, i la primera conté eines que faciliten el treball d'aquestes dades. Aquestes són:

- **rviz**: per simulació i visualització 3D.

[47] Veure pàgina 47, punt 7. 1. 2, Kinect.

- **rosbag**: per gravar i reproduir missatges de comunicació.
- **catkin**: eina de compilació, basada en CMake.
- **rosbash**: paquet d'eines per ampliar les funcions del shell bash.
- **roslaunch**: per executar nodes ROS de forma local o remota.

La determinació de la situació en l'espai es basa en un sistema 3D de mesura Euclidià. El primer que fa el sistema per tal d'establir els eixos XYZ és trobar el pla que correspon a terra. S'utilitza un algoritme RANSAC(nota al peu) per trobar els diversos plans i en funció de la seva distància i orientació respecte a la càmera es determina quin d'ells és el terra. Es comprova si hi ha coincidència amb l'eix Z' (el procedent de la perspectiva de l'usuari), si no és el cas, es repeteix el procés de recerca.
Un cop s'ha trobat el pla floor, es construeix la matriu que determina les projeccions dels eixos X'Y'Z'. Sabem que la normal[48] respecte a terra és paral·lela a l'eix Y, i l'origen de coordenades es troba a terra, altura zero. Tenim que l'equació del pla és Ax + By + Cz + D = 0, on D és la distància perpendicular que hi ha entre el pla i l'origen de coordenades i que defineix el vector normal com n = (A, B, C). Tenint en compte els angles respecte de l'eix de les X (α) i l'eix de les Z (γ), ens queda que la matriu que troba les projeccions dels eixos en la imatge a partir de les originals és:

$$T = \begin{pmatrix} \cos\gamma & -\sin\gamma\cos\alpha & \sin\gamma\sin\alpha & 0 \\ \sin\gamma & \cos\gamma\cos\alpha & -\cos\gamma\sin\alpha & -D \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Un cop tenim les projeccions dels eixos euclidians passem al següent pas: trobar possibles esglaons a partir de la segmentació de l'escena. El primer pas per segmentar és extreure les normals de diversos grups de punts per veure la forma i curvatura de les figures (*Normal estimation*). Seguidament, començant des del punt amb menys curvatura, estenem la seva regió als seus punts més propers amb curvatura similar i repetim el procés amb aquells punts fins que les diferencies de

[48] Definim la força normal Fn com la força que exerceix una superfície sobre un cos que es troba recolzat en aquesta.

curvatures són mínimes (*Region-growing*). A continuació el mateix algoritme RANSAC del principi analitza si els plans més grans de cada regió es corresponen a objectes que hem definit dintre del sistema com plans (parets i terra), si la resposta és afirmativa seran interpretats com a plans totals, en cas contrari ho seran com obstacles (*Planar test*). Els punts que no s'han pogut classificar en cap regió s'inclouen en aquells plans on l'angle entre la seva normal i la del pla és prou petit i la distància és baixa (*Planes extension*). Els punts que després del procés anterior continuen sense pertànyer a cap regió són interpretats com a obstacles (*Euclidean cluster extraction*).

Ara que ja hem segmentat l'escena, classifiquem els plans en dos grups en funció de l'angle que forma la seva normal respecte de la normal del floor. Si l'angle que formen és proper a 0° es classificaran com a horitzontals, mentre que si és proper a 90° seran anomenats walls. Per poder trobar els possibles esglaons, només ens queda evitar que es reconeguin les parets o el terra com a aquests. Per això, indiquem que es reconeguin només com a steps candidate aquelles regions on la Z' (l'altura) és negativa o positiva i que compleixen el Codi Tècnic d'Edificació a Espanya[49](Fig. 42).

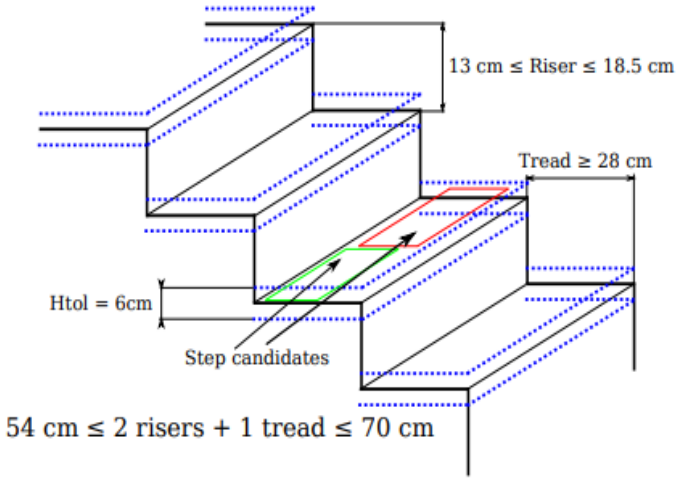


Figura 42. Esquema de les mesures que pren el model de les escales
Font: https://github.com/aperezyus/stairs_detection

Ja hem vist com trobar els step candidates, ara toca passar al sistema de Stairs detection, on l'input seran els candidats i l'output les veritables stairs amb les seves

[49] Hmax=18,5 cm, Hmin=13 cm, i per horitzontal Hmin - Htol/2 = 10 cm des del terra.

característiques completes. El primer que fa el detection algorithm és descartar els candidats que no pertanyen a l'escala a partir de les connexions que hi ha entre les diferents possibilitats per llavors classificar-los per nivells en funció del número de steps que hi ha des del floor fins a ells. El primer que es troba és el first step, aquest és el candidat que compleix que la distància entre el terra i ell és inferior a Hmax + Htol/2 = 21,5 cm. A més, si el terra està dintre del rang visualitzat el first step ha d'estar directament en contacte amb ell. El programa indicarà que no hi ha escales si no troba cap step candidate que encaixi com a first step. Un cop l'algoritme ha trobat el primer esglaó, comença a comparar les altures respecte al terra de la resta de step candidates per determinar si pertanyen al mateix nivell o no. Si el nombre de nivells resulta en més d'un, el programa comença a modelar les escales.

El modelatge de les escales es basa a tornar a transportar els nostres eixos X'Y'Z' a cada un dels steps candidates. Es repeteix el procés realitzat anteriorment per eliminar les concavitats que puguin quedar-hi entre uns i altres i es fa una comparació entre les àrees formades pels punts detectats inicialment i l'àrea formada finalment per mesurar la qualitat dels esglaons resultants. Escollim el millor esglaó com aquell que ha tingut una major extensió d'àrea plana i utilitzem les seves mesures i posició per corregir les de la resta de steps. Finalment, les stairs queden definides per esglaons amb forma de paral·lelograms. La seva amplada serà la del step seleccionat anteriorment com a millor, la seva altura serà la mitjana de les distàncies que hi ha entre els esglaons consecutius, i el seu llarg (la seva profunditat) la mitjana de la distància horitzontal entre cada dos passos. Un cop tenim tots aquests paràmetres es valida la detecció, i si aquesta és correcta s'executa el model.

8.3.3. Instal·lació prerequisits

En primer lloc, hem d'instal·lar un sistema operatiu (OS), específicament Ubuntu, si aquest encara no està present al dispositiu. Nosaltres ja comptem amb anterioritat de la versió Focal Fossa, així que obviarem aquest pas.

A continuació procedim a la instal·lació de ROS (Robot Operating System). ROS és una col·lecció de frameworks dirigits al desenvolupament de software de robots, el que es coneix com a middleware robòtic. El model original del programa indica que aquest està pensant per ser executat mitjançant ROS Melodic, una versió (també anomenada *distribution*) del sistema operatiu llançada el 2018 dirigida als usuaris d'Ubuntu 18.04. Però nosaltres treballem amb Ubuntu 20.04, versió amb la qual ROS Melodic no funciona. Per això adaptarem les instal·lacions i passarem a treballar amb ROS Noetic, la versió més nova d'aquest sistema. Tanmateix, hem comprovat que el model pot funcionar amb les versions Noetic, Melodic i Kinetic.

El primer que farem és configurar el nostre arxiu sources.list. Aquest és bàsicament el mapa que necessita el nostre sistema per saber on instal·lar el programa.

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu
$(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-
latest.list'
```

A continuació configurem les nostres *keys*, això per administrar i verificar els paquets que instal·lem.

```
sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80 -
-recv-key 421C365BD9FF1F717815A3895523BAEEB01FA116
```

Ara actualitzem el sistema junt amb els paquets instal·lats.

```
sudo apt-get update
```

El següent pas no és estrictament necessari, però ens evitarà futurs problemes a l'hora de treballar amb ROS.

```
sudo apt-get install ros-noetic-desktop-full
```

Creem un entorn virtual que ens servirà com a directori d'instal·lació aïllat. Així podem localitzar tot el que instal·lem sense haver de descarregar-lo en tot el sistema.

```
echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

Per poder treballar amb totes les dades, tant les proporcionades pel programa com les que un usuari generi més endavant, instal·lem el paquet de ROS OpenNI. Aquest paquet permet treballar amb les dades directament.

Nosaltres hem d'instal·lar la primera versió del paquet perquè és la compatible amb ROS Noetic. En canvi, si treballem amb les versions Melodic o Kinetic hem d'instal·lar la segona versió del paquet; ROS OpenNI2.

```
sudo apt-get install ros-noetic-openni-launch
```

8.3.4. Configuració de l'espai de treball.

Procedim a la creació d'un catkin workspace. Aquest és el nom que reben els espais de treball en ROS. Entenem com a espai de treball un conjunt de directoris o carpetes on emmagatzem tot el codi ROS.

Creem el nostre workspace (anomenat `catkin_ws`)

```
$ mkdir -p ~/catkin_ws/src
$ cd ~/catkin_ws/
$ catkin_make
```

Configurem, dintre d'aquest, un nou entorn virtual.

```
echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

Finalment, comprovem si ho hem configurat bé amb la següent instrucció, que hauria de donar-nos com a resposta `/home/user/catkin_ws/src:/opt/ros/noetic/share`

```
echo $ROS_PACKAGE_PATH
```

8.3.5. Instal·lació del Stairs detection

Hem d'assegurar-nos que instal·lem el següent paquet en l'espai de treball `catkin_ws/src`. Per això primer hi accedim amb la instrucció `cd`, que ens permet canviar de directori.

```
git clone https://github.com/aperezyus/stairs_detection.git
```

Procedim a compilar el programa:

```
cd ~/catkin_ws
catkin_make
```

8.3.6. Activació del programa amb la càmera Kinect

Utilitzem ROS per iniciar el controlador i que la càmera comenci a generar els núvols de punts. Amb la Kinect connectada (hem de comprovar que el sistema la detecti com a càmera) obrim en el terminal el següent:

```
rosrun stairs_detection stairs
```

Si no hi ha cap error com a resposta, hauria d'obrir-se una nova finestra de terminal que mostri dades sobre les dimensions de les escales detectades. A continuació obrim una altra finestra que mostri la gravació en directe amb les escales i els eixos indicats.

```
roslaunch openni_launch openni.launch depth_registration:=true
load_driver:=false
```

Durant la realització del treball ens vam trobar amb un gran problema en aquest apartat. Tal com hem mencionat, hem d'assegurar-nos de què la Kinect està sent detectada. Això vol dir que s'han de detectar el motor, la càmera i el micròfon. En el nostre cas, només estava detectant-se el motor, cosa que impossibilitava que el sistema funcionés. Després d'analitzar-ho i notar que el llum de l'adaptador de corrent de la càmera parpellejava en comptes de mantenir-se constant, vam sospitar que hi

havia un problema amb un dels condensadors del carregador. Arribats a aquest punt, vam decidir obrir el connector per veure què passava.

Figura 43. Imatge del transformador un cop obert.

Font: Imatge pròpia.

Però al trobar que l'extracció del condensador posava en perill tot el conjunt de l'adaptador i que el temps invertit en la seva reparació no valdria la pena, vam decidir tallar-lo i substituir-lo per un altre carregador que tingués el mateix voltatge que aquest. El seu amperatge és major, però això no hauria de ser un problema de cara al funcionament.

Figura 44. Imatge del nou transformador.

Font: Imatge pròpia

Era necessari un adaptador que ens permetés connectar el nou carregador i la resta de l'aparell, així que vam aprofitar un propi que teníem a mà. Només calia extraure'l de la placa on es trobava.

Figura 45. Imatge del donant del component.

Font: Imatge pròpia.

Un cop extret l'adaptador, cal comprovar la seva polaritat i la corresponent a cadascun dels cables per tal de realitzar correctament la seva connexió.

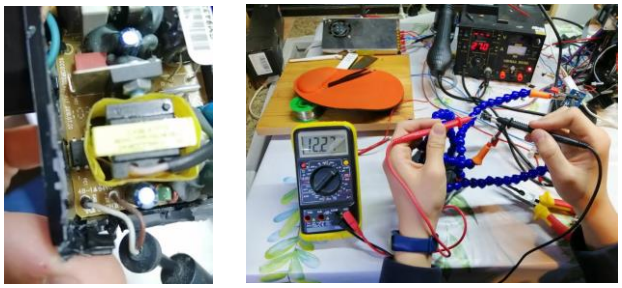


Figura 46. Procés de comprovació de la polaritat del transformador i la càmera.
Font: Imatge pròpia.

Un cop verificada la connexió i el pas correcte del voltatge, passem a comprovar la funcionalitat amb la Kinect. Afortunadament, ara sí que es detectaven totes les parts de la càmera, cosa que indicava que el voltatge que hi arribava anteriorment era massa baix, de manera que només s'arribava a localitzar el motor, que és el que menys voltatge requereix.

8.3.6. Prova del model

Com que en aquest cas tenim l'avantatge de què aquest model ja ha sigut entrenat anteriorment, podem ometre aquesta part. Passarem a veure directament si funciona correctament i quins són els errors més comuns que fa.

De cara a veure el seu rendiment en diferents graus de complexitat d'escales, compararem els seus resultats amb unes escales públiques en un espai obert (unes d'institut) i unes privades en un espai tancat (d'una casa).

Comencem amb les escales d'institut. Aquestes han sigut construïdes dintre de la normativa espanyola, així que les seves mesures haurien de ser detectades pel programa com a suficients dintre de les seves limitacions. A més, no presenten cap mena d'obstacles i la seva amplària és prou extensa per a poder tenir una millor referència dels plans. Els resultats obtinguts en aquesta prova són els següents:

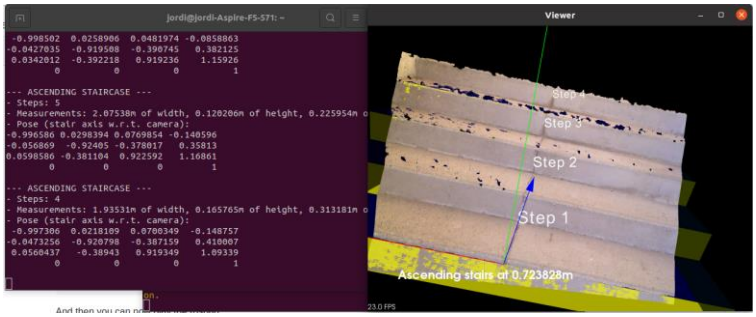


Figura 47. Captura del programa detectant les escales de l'institut.

Font: Captura pròpia

Com veiem, les escales són detectades correctament pel programa tan ascendentment com descendentment.

Passem a les següents. Les escales de la casa no han sigut construïdes per un organisme públic, així que no compleixen estrictament la normativa. A més, les dimensions dels seus esglaons varien, és corba en certes parts i l'amplada dels esglaons és bastant reduïda en comparació a les anteriors. La dificultat per a la detecció d'aquestes és molt superior. Els resultats de la segona prova són aquests:

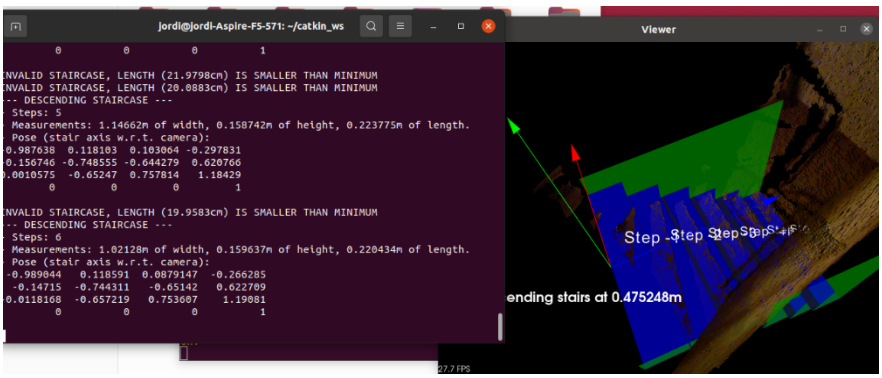


Figura 48. Imatge de la detecció d'escales que no compleixen estrictament la norma

Font: Captura pròpia

Malgrat les variacions de mides, l'algoritme ha sigut capaç de detectar gairebé tots els esglaons de les escales. Sembla que aquest no és un gran problema a l'hora de localitzar els steps i les relacions entre aquests. En canvi, la part corba final no ha sigut detectada, mentre que la inicial sí. El fet que hi hagi zones corbes torna imprecís el model, que pot o no detectar-les.

Si comparem els resultats de les dues proves, veiem que el model funcionarà millor com més se cenyeixin les escales a la normativa en la qual es basa l'algoritme.

Els errors trobats durant les proves que podem destacar són:

- Aleatorietat en la detecció d'escales amb zones corbes
- Necessitat d'un angle específic de la càmera per funcionar
- Augment de l'error amb poc nivell de lluminositat

8.3.7. Extensió per àudio

Ara que ja tenim el model funcionant, hem de fer que l'usuari invident pugui rebre la informació que el programa mostra en pantalla per forma de sortida d'àudio, com ja havíem fet anteriorment amb la primera funció de l'aparell (Object Detection). Però en aquest cas no podem recórrer a la mateixa opció que abans, perquè el model d'aquest apartat està escrit en C++ d'alt nivell, un llenguatge de programació que no dominem. Per tant, la primera idea que se'ns va ocórrer per realitzar la transmissió de la informació a àudio, que era trobar la variable que corresponia al nombre de *steps* i ordenar que el programa recitès, a través d'una llibreria que passés els grafemes a fonemes, tant el nombre d'esglaons com la distància a la qual les escales es trobaven i si aquestes eren ascendents o descendent, queda eliminat. Això no treu la possibilitat de poder-se realitzar d'aquesta manera, però nosaltres ens trobem limitats pels nostres coneixements mínims. La segona idea, llavors, és realitzar simplement una lectura de la pantalla en la qual apareixen les escales junt amb la distància, el tipus i els *steps*. El desavantatge d'aquesta opció enfront de la primera és que llegirà cadascun dels steps de dalt a baix en comptes de dir directament el número, però encara així compleix la seva funció.

Per aquesta funció aprofitarem un model enfocat en la lectura de llibres de la tauleta de lectura Kindle. Aquest programa fa una captura de la pantalla i llegeix el text que hi trobi amb una veu artificial. Està dissenyat per ser implementat en Windows, però

nosaltres en aquests moments treballem en Linux (per Ubuntu), així que haurem de fer algunes adaptacions.

El primer que hem de fer és descarregar pytesseract, una de les llibreries que utilitza, des de la seva pàgina web. A l'hora d'instal·lar-lo, guardem la seva ubicació d'instal·lació i canviem aquesta per la que apareix a la línia número 20 del codi del programa. Ara instal·lem també la versió binaria de la llibreria des del terminal.

```
sudo apt update
sudo apt install tesseract-ocr
sudo apt install libtesseract-dev
```

I per últim instal·lem totes les llibreries que utilitza. En principi, només amb l'ordre següent és suficient. Però en cas que doni error es pot fer la instal·lació individual de cadascuna de les llibreries, que són: PyAutoGUI (0.9.50), numpy (1.20.2), pytesseract (0.3.7), opencv-python (4.5.1.48), Pillow (7.2.0), pyttsx3 (2.90).

```
pip install -r requirements.txt
```

L'últim que queda és indicar la part de la pantalla on volem que es faci la captura d'imatge. Podem definir el requadre on es troba la finestra del Stairs detection o podem fer que aquesta ocupi tota la pantalla, les dues opcions funcionen.
Per modificar l'àrea capturada, anem a la línia 23 del codi, on els valors de les variables estàn ordenats començant per la cantonada superior esquerra i acabant en la inferior dreta.

```
cap = ImageGrab.grab(bbox =(5, 110, 1335, 1032))
```

8.4. Text-to-Speech

L'última funció que volem implementar no està enfocada en l'ajut a la mobilitat de l'usuari, sinó a la resolució d'un altre dels grans problemes de les persones invidents: la lectura. Encara que és cert que per a això ja hi ha alternatives com els textos en braille, les extensions d'Internet per la lectura de pantalla en cas de textos en

pantalles, entre d'altres, és interessant donar una possibilitat que permeti accedir a la informació de qualsevol text en qualsevol moment.

Afortunadament, ja existeix un tipus de IA que ens facilitarà enormement l'addició d'aquesta nova funció: el Text to Speech.

El procés es basa en rebre i seleccionar text per reproduir-lo (o en alguns casos fins i tot traduir-lo) per a continuació reproduir-lo amb una veu artificial sintetitzada. Per això, aquesta tecnologia és coneguda també com a Speech Synthesis.

Per ser més específics, el Text-to-speech no seria una IA en si, sinó una tecnologia assistida que utilitza algoritmes impulsats per IA. Concentra dos camps on la intel·ligència artificial és realment potent: el Text recognition i el Text-to-speech. El segon necessita a més, un sistema IA centrat en el Natural Language Processing (NPL). Aquest és el que s'encarrega de generar veus artificials que tinguin certa semblança amb una natural.

8.4.1. Network architecture

L'esquema de funcionament d'un sistema Text-to-speech típic (i el que utilitzarem) és el següent:

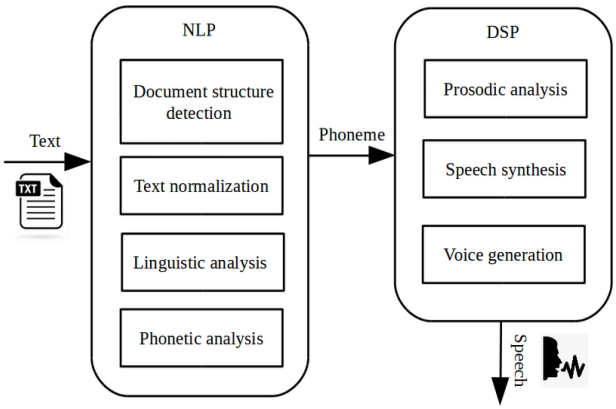


Figura 49. Esquema del funcionament d'un sistema TtS.
Font: https://www.researchgate.net/figure/Standard-Text-to-speech-system-architecture_fig2_343861915

On l'input és el text, que és processar pel NPL Aquest dona com a output el fonema corresponent, que acaba sent l'output final, la lectura del text.

Dintre del NPL es fan diversos passos. El primer, el Document structure detection s'encarrega d'identificar quines són les regions d'interès de la imatge que arriba (en el cas de fotografies, textos escanejats, etc.). Diferència les zones de text de les que no en tenen en funció de la disposició en la imatge. El següent pas és el Text Normalization, es tracta de transformar el text de manera que passi a tenir una única forma per tal de poder realitzar-hi funcions de manera consistent a tot el conjunt. Els següents passos, Linguistic and Phonetic analysis, són els mitjans per tractar d'entendre el conjunt del text per passar-lo a grafemes i després a fonemes

El DPL és una funció més avançada que dona entonació i pauses a la lectura, però que en aquest cas no utilitzarem.

8.4.2. Instal·lació de dependències

Nosaltres introduirem el Text-to-speech a partir d'un sistema de lectura amb la càmera del portàtil. D'aquesta manera, l'usuari pot capturar en imatge el text que vulgui i aquest serà llegit pel dispositiu.

El sistema que utilitzarem requereix les llibreries OpenCV, pytesseract i win32.

pytesseract: És una eina OCR per Python que pot reconèixer i llegir text en imatges.

opencv: Biblioteca lliure especialitzada en visió artificial que ens permet treballar amb imatges.

win32com.client: És un paquet de mòduls de Python que permet l'accés als automation objects. Facilita l'automatització d'aplicacions en Windows.

Cal destacar que, ja que aquesta última llibreria només es troba disponible en Windows, aquesta tercera funció de l'aparell no podrem fer-la amb el mateix SO que

les dues anteriors. Així que a partir d'ara treballarem amb el sistema operatiu Windows.

Des del terminal, comencem amb la instal·lació d'aquestes llibreries.

```
py -m pip install opencv-python
```

```
py -m pip install pypiwin32
```

En cas de que el sistema no trobi pytesseract en aquest instant, hem de canviar el PATH on es troba o fer el següent des del terminal:

```
pi -m pip install update
pi -m pip install tesseract-ocr
pi -m pip install libtesseract-dev
```

A continuació executem el programa.

```
python3 Main.py
```

8. 4. 3 Funcionament del programa

El codi complet del programa utilitzat es deixarà a l'annex, aquí procedim a l'explicació del procés que realitza.

Primer de tot, importem les llibreries i la carpeta `speak`, que definirem més endavant. També invoquem el PATH de pytesseract, abans de comencar a operar amb les imatges per evitar errors comuns que solen aparèixer treballant amb aquesta llibreria.

```
import cv2
import pytesseract
from speaker import speak
pytesseract.pytesseract.tesseract_cmd='C:\\Program
```

```
Files\\Tesseract-OCR\\tesseract.exe'
```

Capturem el data input de la càmera (per fer la captura de pantalla haurem de pressionar el botó “Q” del teclat de l’ordinador).

Ara que ja tenim l’input text, hem de segmentar-lo i modificar-lo per tal que pytesseract pugui processar-lo. La càmera fa arribar les imatges en model BGR, en aquest model les imatges s’emmagatzemen en àrees corresponents a tres colors, on el primer és el més rellevant i l’últim el de menys importància. Tal com indiquen les seves inicials (Blue Green Red), el vermell és el de l’àrea menys important. En canvi, pytesseract treballa només imatges que es trobin en model RGB (Red Green Blue), aquest funciona igual que l’anterior, però l’ordre de rellevància de les àrees és al revés.

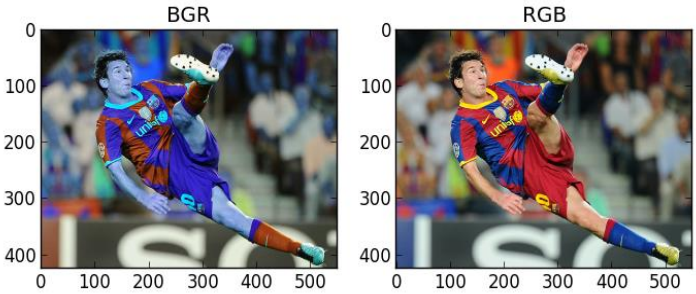


Figura 50. Comparació amb la mateixa fotografia en model RGB i BGR. A l’esquerra el BGR, a la dreta RGB.
Font: https://stackoverflow.com/questions/15072736/extracting-a-region-from-an-image-using-slicing-in-python-opencv

Tal que:

```
cap = cv2.VideoCapture(0)
while(True):
    _,img =cap.read()


    cv2.imshow("tee",img)
    k=cv2.waitKey(5) &0xFF

    if k == ord('q'):
        break
```

Una vegada completa l'adaptació de la data, comencem a definir el quadre que captura la informació. Dintre de les dimensions que la càmera ja té hem de definir quina àrea d'aquesta és la que pot detectar text. A més, hem d'indicar com funciona la disposició en l'espai del tipus de text que es rebrà perquè el programa no tingui errors en l'ordre de traducció.

```
hImg,wImg,_ = img.shape
Idata=pytesseract.image_to_data(img, lang='eng')
sentence=[""]
for x,b in enumerate(Idata.splitlines()):
    if x != 0:
        b=b.split()
##        print(b)
        if(len(b)==12):
##            x,y,w,h = int (b[6]),int(b[7]),int (b[8]),int (b[9])
##            cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),1)
```

Ara definim el llenguatge en que funciona el programa. És a dir, quin codi ha d'utilitzar sobre els textos que li arriben (les mateixes grafies tenen diferents fonemes en funció del llenguatge del que parlem). També indiquem què ha d'interpretar com a dimensions de la imatge que li arriba.

```
print("start")

print(pytesseract.image_to_data(img, lang='eng'))
hImg,wImg,_ = img.shape
Idata=pytesseract.image_to_data(img, lang='eng')
sentence=[""]
```

Per últim, definim què és `speak`. Utilitzem la funció Dispatch de dintre de la llibreria win32 per generar la veu artificial que utilitzarem per narrar i tornem a `speak` l'ordre de lectura.

```
from win32com.client import Dispatch

def speak(str):
    speak = Dispatch(("SAPI.SpVoice"))
    speak.Speak(str)
```

```
if __name__ == '__main__':
    speak(word234)
```

8. 4. 4 Prova del model

Posem a prova el programa amb textos de diferents mides, colors i tipus de lletra.

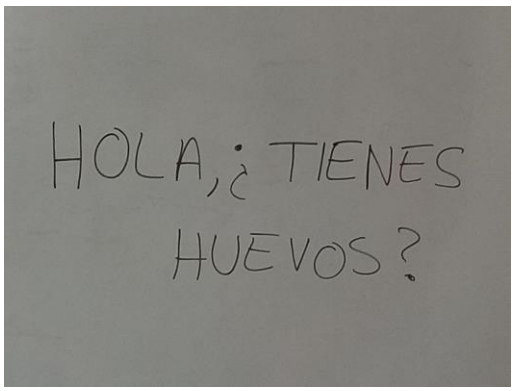
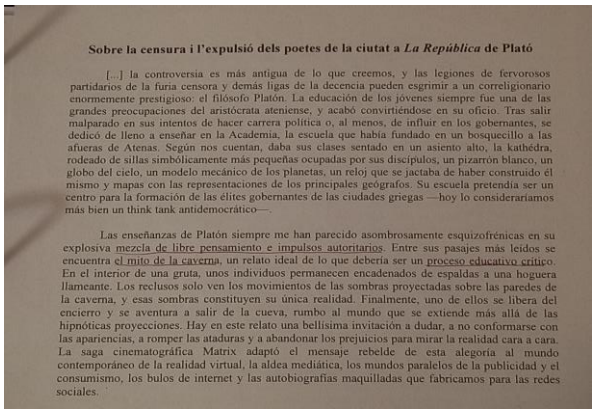


Figura 51. Text escrit a mà en una petita pissarra on es pot llegir "Hola, ¿Tienes huevos?".
Font. Imatge pròpia.

En aquest primer text fet a mà no hi ha lectura del text. La detecció és molt pobra, només detecta un sol caràcter.



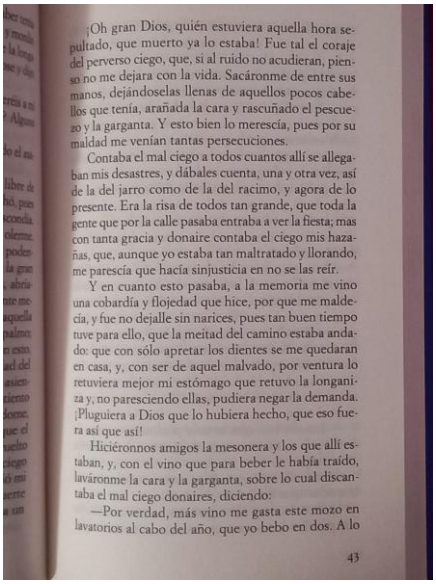
Sobre la censura i l'expulsió dels poetes de la ciutat a *La República* de Plató

[...] la controversia es más antigua de lo que creemos, y las legiones de fervorosos partidarios de la furia censora y demás ligas de la decencia pueden esgrimir a un correligionario enormemente prestigioso: el filósofo Platón. La educación de los jóvenes siempre fue una de las grandes preocupaciones del aristócrata ateniense, y acabó convirtiéndose en su oficio. Tras salir malparado en sus intentos de hacer carrera política o, al menos, de influir en los gobernantes, se dedicó de lleno a enseñar en la Academia, la escuela que había fundado en un bosquecillo a las afueras de Atenas. Según nos cuentan, daba sus clases sentado en un asiento alto, la kathédra, rodeado de sillas simbólicamente más pequeñas ocupadas por sus discípulos, un pizarrón blanco, un globo del cielo, un modelo mecánico de los planetas, un reloj que se jactaba de haber construido él mismo y mapas con las representaciones de los principales geógrafos. Su escuela pretendía ser un centro para la formación de las élites gobernantes de las ciudades griegas —hoy lo consideraríamos más bien un think tank antidemocrático—.

Las enseñanzas de Platón siempre me han parecido asombrosamente esquizofrénicas en su explosiva mezcla de libre pensamiento e impulsos autoritarios. Entre sus pasajes más leídos se encuentra el mito de la caverna, un relato ideal de lo que debería ser un proceso educativo crítico. En el interior de una gruta, unos individuos permanecen encadenados de espaldas a una hoguera llameante. Los reclusos solo ven los movimientos de las sombras proyectadas sobre las paredes de la caverna, y esas sombras constituyen su única realidad. Finalmente, uno de ellos se libera del encierro y se aventura a salir de la cueva, rumbo al mundo que se extiende más allá de las hipnóticas proyecciones. Hay en este relato una bellísima invitación a dudar, a no conformarse con las apariencias, a romper las ataduras y a abandonar los prejuicios para mirar la realidad cara a cara. La saga cinematográfica Matrix adaptó el mensaje rebelde de esta alegoría al mundo contemporáneo de la realidad virtual, la aldea mediática, los mundos paralelos de la publicidad y el consumismo, los bulos de internet y las autobiografías maquilladas que fabricamos para las redes sociales.

Figura 52. Fotocòpia de text acadèmic de caràcter filosòfic, reflexió sobre la censura i l'expulsió dels poetes a La República de Plató.
Font: Imatge pròpia.

Text acadèmic amb fragments subratllats amb vermell. Sí que ho detecta i ho llegeix. Segons sembla, només és capaç de detectar text imprès.



¡Oh gran Dios, quién estuviera aquella hora sepultado, que muerto ya lo estaba! Fue tal el coraje del perverso ciego, que, si al ruído no acudieran, pienso no me dejara con la vida. Sacáronme de entre sus manos, dejándoselas llenas de aquellos pocos cabellos que tenía, arañada la cara y rascuñado el pescuezo y la garganta. Y esto bien lo merescía, pues por su maldad me venían tantas persecuciones.

Contaba el mal ciego a todos cuantos allí se allegaban mis desastres, y dábales cuenta, una y otra vez, así de la del jarro como de la del racimo, y agora de lo presente. Era la risa de todos tan grande, que toda la gente que por la calle pasaba entraba a ver la fiesta; mas con tanta gracia y donaire contaba el ciego mis hazañas, que, aunque yo estaba tan maltratado y llorando, me parescía que hacía sinjusticia en no se las reír.

Y en cuanto esto pasaba, a la memoria me vino una cobardía y flojedad que hice, por que me maldecía, y fue no dejalle sin narices, pues tan buen tiempo tuve para ello, que la meitad del camino estaba andado: que con sólo apretar los dientes se me quedaran en casa, y, con ser de aquel malvado, por ventura lo retuviera mejor mi estómago que retuvo la longaniza y, no paresciendo ellas, pudiera negar la demanda. ¡Pluguiera a Dios que lo hubiera hecho, que eso fuera así que así!

Hiciéronnos amigos la mesonera y los que allí estaban, y, con el vino que para beber le había traído, laváronme la cara y la garganta, sobre lo cual discantaba el mal ciego donaires, diciendo:

—Por verdad, más vino me gasta este mozo en lavatorios al cabo del año, que yo bebo en dos. A lo

43

Figura 53. Pàgina del llibre El Lazarillo de Tormes, ho detecta i ho llegeix. Confirmem la idea que només detecta text imprès.

Font: Imatge pròpia.

Repetim de nou les tres proves amb els mateixos textos, però canviat l'entrada d'imatge, passem de la càmera del portàtil a una càmera web amb més resolució. Els resultats són els mateixos, verifiquem la teoria.

Errors trobats durant les proves realitzades:

· Gran pèrdua de capacitat de detecció amb baixa lluminositat.

· Veu artificial massa robòtica.

· Acostuma a confondre paraules.

9. Conclusions

Un cop hem acabat la investigació, toca mirar cap enrere i valorar el treball que hem realitzat. En una primera instància, podem afirmar que hem aconseguit complir amb l'objectiu principal del projecte, el qual era poder desenvolupar un aparell capaç de poder assistir a les persones amb problemes de visió a partir de mètodes basats en models de visió per computació. El resultat final ha estat un aparell dividit en tres parts. La primera, basada en la detecció d'objectes, que ens ha permès localitzar i reconèixer obstacles comuns en les vies urbanes dels nostres pobles i ciutats de forma prou satisfactòria. La segona part, ens ha permès detectar uns dels obstacles que més inconvenients creiem que porten a totes les persones cegues: Les escales i els desnivells pronunciats en el terra. I la tercera i última part, on hem aconseguit poder desenvolupar una de les principals característiques que tenen la majoria de productes similars al nostre projecte, poder llegir i interpretar textos mitjançant una càmera.

Si bé hem aconseguit tots els objectius plantejats, cal destacar algunes de les mancances que el nostre projecte posseeix. En primer lloc, destacar l'absència de certes categories d'objectes que ens hagués agradat incloure dins de la detecció d'objectes. Com ja s'ha dit en la corresponent secció[50], l'error és causat per l'ús d'un dataset extret d'internet. Si bé aquest compleix bastant amb la seva funció, hagués estat molt millor haver creat nosaltres mateixos el dataset, incloent-hi totes les categories que hem esmentat durant el transcurs del treball. Aquesta opció no ha estat escollida en aquest treball degut a la complexitat i el temps que aquesta requereix. Seria objecte de millora en futures versions. A part de les categories a detectar, també és important destacar el baix rendiment que el model ha obtingut en termes generals. Si bé hem intentat esbrinar els motius en l'Annex 1, no hem sigut capaços de poder implementar una solució al complet. Aquests dos problemes es podrien resumir en un mal dataset i la seva solució, la recomposició de les imatges que el formen. Un dels punts positius que hem obtingut, a part del funcionament

[50] Consultar pàgina 81 del treball.

general del model ha estat la possibilitat de poder narrar tot allò que el model detectava. Aquesta era una idea que en un principi no estava contemplada, però admetem que era necessària i el seu resultat ha estat més que satisfactori.

Un altre defecte a tenir en compte, aquest cop dintre de la segona funció, és el fet que l'aparell necessita enfocar les escales des d'un angle específic per tal de poder detectar-les. Això provoca que aquestes no siguin reconegudes en algunes ocasions si l'usuari no està en posició correcta. A més, l'extensió per àudio d'aquesta mateixa part no és com ens agradaria que fos. Tal com hem explicat anteriorment[51], condicionants com la nostra limitació de coneixements han resultat en fer un model que simplement llegeixi la pantalla en comptes de les variables, deixant-nos amb una solució no ideal.

Pel que fa a les mancances de la tercera part, podem senyalar que té un percentatge d'error bastant alt a l'hora de reconèixer paraules i que el seu rendiment empitjora amb situacions de poca llum. A més, el model està dissenyat per llegir només textos en castellà, encara que es deixa la possibilitat dintre del codi de canviar-ho a l'idioma que es vulgui substituint la llibreria d'idioma importada.

A part de les coses que es podrien millorar de cada component del prototip, també hem de tenir en compte els recursos i la situació econòmica en què ens trobem. El cor del nostre projecte ha estat la Raspberry Pi 4 de 4 GB. Aquesta no ha estat capaç de poder calcular totes les operacions necessàries com per poder executar els tres components a la vegada. Tot i que sembli que la Raspberry Pi 4 sigui un ordinador relativament nou, cal destacar que aquesta va sortir al mercat l'any 2019. En total, el model de la Raspberry Pi 4 ja fa uns 4 anys que va sortir. En un principi, això no hauria de suposar un gran problema, però en el món que avança tan ràpidament com en el món de la informàtica podríem dir que aquest model ja està gairebé obsolet. A més, també cal saber que si bé el model té l'edat que té, el seu processador és encara més

[51] Veure pàgina 111.

antic, ja que aquest va ser presentat el 2015. En termes teòrics, ja s'hauria d'haver presentat el nou model de Raspberry des de ja fa uns anys, però el context de pandèmia i les seves posteriors repercussions en la fabricació de microxips han cancel·lat la seva presentació. Si bé podríem haver utilitzat plaques més potents, aquestes tenien un preu massa elevat pel rendiment que realment oferien. És un punt a millorar pel futur actualitzar la central de computació d'aquest projecte.

Malgrat tots aquests inconvenients, podem dir que el resultat final del projecte ha sigut un èxit tenint en compte el punt de partida des del que iniciàvem: Un pressupost de 120€ i mínims coneixements sobre el tema. El fet que haguem pogut desenvolupar un aparell amb aquestes característiques demostra que amb una mica d'inversió en investigació en els camps en els quals hem estat treballant (IA i visió artificial) s'hi podrien aconseguir grans innovacions tecnològiques. A més, els nostres resultats són una prova de què els preus de productes similars que grans empreses tenen en circulació són massa abusius de cara al client. Per posar un exemple, un dels models citats anteriorment en aquest treball, les Envision Glasses, tenen un preu de 2.499,00$. Aquests alts preus fan que el producte no sigui accessible per a tothom, cosa impensable sabent que es tracta d'eines per afavorir la inclusió d'una minoria en el món actual.

Personalment, trobem que aquest projecte ha sigut una gran font de coneixement que ens ha brindat l'oportunitat de no només ampliar les nostres capacitats tecnològiques i nodrir les nostres ments, sinó també de créixer com a persones, podent contribuir a la humanitat utilitzant totes aquestes eines que hem anat aconseguint durant la nostra investigació.

Repositoris de GitHub

- Jocher, Glenn (2020) YOLOv5 by Ultralytics[Repositori GitHub]. https://github.com/ultralytics/yolov5.

- Tzutalin (2015) LabelImg [Repositori Github]. https://github.com/tzutalin/labelImg

- TW0521 (2021) Obstacle-dataset OD [Repositori GitHub]. https://github.com/TW0521/Obstacle-Dataset.

- Perez-Yus, Alejandro and Gutierrez-Gomez, Daniel and Lopez-Nicolas, Gonzalo and Guerrero, J. J. (2015) Stairs detection with odometry-aided traversal from a wearable {RGB-D} camera [Repositori GitHub]. https://github.com/aperezyus/stairs_detection.
- Rob, sftppgn (2021) Audible-clone [Repositori Github]. https://github.com/sftppgn/audible-clone

- Keval Prajapati (2021), Cameratospeech [Repositori Github]. https://github.com/KevalPrajapati/cameratospeech

Webgrafia

Agustí i Melchor, Manuel. Desarrollo de aplicaciones para interacción con el computador mediante el uso del Kinect. Universitat Politécnica de València Departamento de Informática de Sistemas y Computadores (DISCA) (consultat 27 setembre 2022). Recuperat de: https://riunet.upv.es/bitstream/handle/10251/79473/Agust%C3%AD%20-%20Desarrollo%20de%20aplicaciones%20para%20interacci%C3%B3n%20con%20el%20computador%20mediante%20el%20uso%20del%20Kinect.pdf?sequence=1&isAllowed=y

https://hardzone.es/tutoriales/componentes/procesador-arm/ (consultat 17 setembre 2022)

https://medium.com/@ANAI-DemocratizingAI/yolov6-explained-in-simple-terms-c46a0248bddc (consultat 4 Octubre 2022)

https://es.wikipedia.org/w/index.php?title=Arduino&oldid=147703046. (consultat 7 setembre 2022)

https://www.macula-retina.es/conos-y-bastones/ (consultada 21 agost 2022)

https://www.macula-retina.es/macula-lutea/ (consultada 21 agost 2022)

https://medium.com/mlearning-ai/confusion-matrix-for-multiclass-classification-f25ed7173e66 (consultada 6 desembre de 2022)

https://news.mit.edu/2019/mit-mini-cheetah-first-four-legged-robot-to-backflip-0304(consultat 30 agost 2022)

https://www.clinicabaviera.com/blog/que-es-la-retina-para-que-sirve (consultada 21 agost 2022).

https://medium.com/axinc-ai/yolov5-the-latest-model-for-object-detection-b13320ec516b (consultat 5 octubre 2022)

Collado, Sonia. Jiménez, Juan Antonio (2017). Tiflotecnología [Presentació de diapositives]. Universitat d'Alacant. https://web.ua.es/es/cae/documentos/noticias/2017/tiflotecnologia-para-deficit-visual-once-juan-antonio-gimenez-sonia-collado.pdf

https://descubrearduino.com/razones-las-usar-arduino (consultat 6 Setembre 2022)

.https://fundacionfrancina.org/herramientas-para-personas-con-discapacidad-visual/ (consultat 25 agost 2022)

https://es.wikipedia.org/w/index.php?title=Disco_%C3%B3ptico_(oftalmolog%C3%ADa)&oldid=143867924. (consultat 25 agost 2022)

https://medium.com/mlearning-ai/training-yolov5-custom-dataset-with-ease-e4f6272148ad (consultat 3 octubre 2022)

https://www.doctordiegoruizcasas.com/anatomia-globo-ocular/humor-vitreo (consultada 21 agost 2022)

https://towardsdatascience.com/yolov5-compared-to-faster-rcnn-who-wins-a771cd6c9fb4 (consultada 15 setembre 2022)

https://www.perrosguiaecuatorianos.org.ec/historia-del-baston-blanco/b (consultat 27 agost 2022)

https://www.xataka.com/componentes/cisc-frente-a-risc-una-batalla-en-blanco-y-negro (consultat 14 setembre 2022)

https://www.euroinnova.pe/blog/como-funciona-una-camara-de-video#iquestquieres-saber-coacutemo-funciona-una-caacutemara-de-viacutedeo (consultat 23 setembre 2022)

https://stackoverflow.com/questions/50390836/understanding-darknets-yolo-cfg-config-files (consultat 5 desembre 2022)

https://www.xataka.com/basics/que-arduino-como-funciona-que-puedes-hacer-uno/ (consultat 6 setembre 2022)

https://fundacionhomero.webnode.es/preguntas-sobre-la-ceguera/uso-de-baston (consultat 28 agost 2022)

https://www.hwlibre.com/pixy-una-camara-inteligente-para-nuestros-proyectos/ (consultat 8 setembre 2022)

https://www.adslzone.net/esenciales/preguntas/que-es-arm/ (consultat 16 setembre 2022)
https://es.wikipedia.org/w/index.php?title=George_Berkeley&oldid=147679804. (consultada 200 agost 2022)

http://www.xtec.cat/~mgisbert/projecte/lents1.htm (consultada 20 agost 2022)

https://towardsdatascience.com/the-practical-guide-for-object-detection-with-yolov5-algorithm-74c04aac4843 (consultat 3 Octubre 2022)

Instituto de Investigaciones Políticas y Sociales de Guatemala(consultat 27 Agost 2022). EL BASTÓN BLANCO Y SUS INVENTORES. Iips.usac.edu.gt. https://iips.usac.edu.gt/wp-content/uploads/2019/10/EL-BAST%C3%93N-BLANCO-Y-SUS-INVENTORES.pdf

http://oftalmologia-avanzada.blogspot.com/2013/06/como-funciona-nuestro-enfoque-o.html (consultat 20 agost 2022)

Juganaru Mathieu, Mihaela (2014). *Introducción a la programación.* Mèxic. Ed. Patria.

https://keepcoding.io/blog/ventajas-y-desventajas-de-python (consultat 15 setembre 2022)

https://robotics.stackexchange.com/questions/21520/what-are-the-purpose-of-pre-trained-weights-for-yolo-object-detector (consultat 3 decembre 2022)

https://es.wikipedia.org/w/index.php?title=Laura_Bridgman&oldid=145025948. (consultat 20 agost 2022)

https://es.wikipedia.org/w/index.php?title=Lenguaje_de_programaci%C3%B3n&oldid=147767733. (consultat 16 Octubre 2022)

https://www.msdmanuals.com/es/hogar/enfermedades-cerebrales,-medulares-y-nerviosas/trastornos-del-sistema-nervioso-aut%C3%B3nomo/introducci%C3%B3n-al-sistema-nervioso-aut%C3%B3nomo (consultat 19 agost 2022)

https://ecomorph.wordpress.com/2014/05/28/what-genes-tell-us-about-pinhole-and-camera-eye-evolution-in-cephalopods/ (consultat 25 agost 2022)

https://towardsdatascience.com/how-to-prepare-data-for-object-detection-34750c4d00da (consultada 30 setembre de 2022)

https://raspberryparatorpes.net/glossary/bcm2711/ (consultat 20 setembre 2022)

(consultat 27 agost 2022). Braille. Humanidades.com. Última edició: 25 de març de 2020. https://humanidades.com/braille

https://sanchom.wordpress.com/tag/average-precisio (consultada 7 desembre de 2022)

Miller, B. R. (2015, February 11). history of the blind. Encyclopedia Britannica. https://www.britannica.com/topic/history-of-the-blind-1996241

https://towardsdatascience.com/confusion-matrix-for-your-multi-class-machine-learning-model-ff9aa3bf7826 (consultat 6 desembre 2022)

https://www.nobbot.com/personas/baston-inteligente-discapacidad-visual-navegacion-integrada/ (consultat 30 agost 2022)

https://www.newscientist.com/definition/evolution-of-the-eye/ (consultat 24 agost 2022).

https://www.oftalvist.es/blog/que-es-el-iris-del-ojo-y-cual-es-su-funcion
(consultada 19 agost 2022)
https://www.oftalvist.es/blog/fovea-definicion-funcion-y-estructura (consultada 20 agost 2022)

https://www.oftalvist.es/blog/macula-ocular (consultada 20 agost 2022)

https://www.once.es/dejanos-ayudarte/la-discapacidad-visual/louis-braille (consultat 27 agost 2022)

https://www.once.es/servicios-sociales/braille (consultat 28 agost 2022)

https://www.once.es/noticias/los-libros-para-ciegos-de-la-once-si-salen-a-la-calle-para-mostrar-su-modelo-de-acceso-a-la-cultura-y-la-educacion (consultat 30 agost 2022)

https://www.faq-mac.com/2003/10/breve-historia-del-risc-por-alejandro-pena/
(consultat 14 setembre 2022)

https://perrosguiamurcia.org/nuestros-perros/historia-del-perro-guia/(consultat 31 agost 2022)

https://perrosguia.once.es/es/que-hacemos/nuestros-perros (consultat 31 agost 2022)

https://www.phos.co.uk/journal/the-evolution-of-sight (consultat 24 agost 2022)

Poejavlo. (5 de maig de 2012). *Richard Dawkins demonstrates the evolution of the eye*[arxiu de Video]. Youtube. https://www.youtube.com/watch?v=Nwew5gHoh3E

https://pypi.org/project/gTTS/ (consultat 5 desembre 2022)

https://www.quimica.es/enciclopedia/Bast%C3%B3n_%28c%C3%A9lula%29.html
(consultada 22 agost 2022)

https://www.quimica.es/enciclopedia/Conos_y_bastones.html (consultada 22 agost 2022)

https://www.quimica.es/enciclopedia/F%C3%B3vea.html (consultada 22 agost 2022)

https://learnopencv.com/performance-comparison-of-yolo-models/(consultat 4 Octubre 2022)

https://revistaseguridad360.com/destacados/sensor-infrarrojo/ (consultat 9 setembre de 2022)

https://hardzone.es/reportajes/que-es/registros-instrucciones-cpu/(consultat 16 setembre 2022)

https://cienciasdelsur.com/2018/01/02/los-ciegos-en-la-historia/ (consultat 25 agost 2022)

https://educandotumirada.es/que-es-el-enfoque-o-acomodacion/ (consultat 20 agost 2022)

https://machinelearningspace.com/yolov3-tensorflow-2-part-3/ (consultat 30 setembre 2022)

https://www.firstclassmed.com/articles/2018/evolution-of-the-eye(consultat 25 agost 2022)

Sánchez García, Jesús (2017). ACCIÓN SOCIAL. REVISTA DE POLÍTICA SOCIAL Y SERVICIOS SOCIALES, Número I/5, 97-107. ISSN 2341-4529. https://digitum.um.es/digitum/bitstream/10201/52562/1/acci%C3%B3n%20social%201-5.pdf

https://towardsai.net/p/l/precision-recall-curve (consultat 7 desembre 2022)

Schwab, I. The evolution of eyes: major steps. The Keeler lecture 2017: centenary of Keeler Ltd. Eye 32, 302–313 (2018). https://doi.org/10.1038/eye.2017.226

Serna Ruiz, Antonio. Ros García, Francisco Antoonio. Rico Noguera Juan Carlos(2010). *Guía práctica de sensores*. Ed. Creaciones copyright.

https://towardsdatascience.com/precision-and-recall-88a3776c8007 (consultat 7 desembre 2022)

https://towardsdatascience.com/multi-class-metrics-made-simple-part-ii-the-f1-score-ebe8b2c2ca1 (consultat 6 desembre 2022)

https://www.profesionalreview.com/2021/07/18/risc-vs-cisc/(consultat 15 setembre 2022)

https://www.profesionalreview.com/2021/07/17/que-es-risc/(consultat 15 setembre 2022)

https://appsilon.com/object-detection-yolo-algorithm/(consultat 6 desembre 2022).

https://histinf.blogs.upv.es/2011/01/12/computadores-de-8-bits/(consultat 6 setembre 2022)

https://www.v7labs.com/blog/mean-average-precision(consultat 8 desembre 2022)

https://viso.ai/computer-vision/coco-dataset/(consultat 1 octubre 2022).

https://en.wikipedia.org/w/index.php?title=Evolution_of_the_eye&oldid=1120142746 (consultat 8 desembre 2022)

Xiao, Anxing & Tong, Wenzhe & Yang, Lizhi & Zeng, Jun & Li, Zhongyu & Sreenath, Koushil. (2021). Robotic Guide Dog: Leading a Human with Leash-Guided Hybrid Physical Interaction. 11470-11476. 10.1109/ICRA48506.2021.9561786.

https://www.google.com/search?q=google+glasses+for+the+blind&rlz=1C1YTUH_esES1022ES1022&sourceid=chrome&ie=UTF-8 (Consultat el 2 de desembre)

https://docplayer.es/19036658-Vision-artificial-eloi-maduell-i-garcia-pid_00184756.html (Consultat el 2 de desembre)

https://www.curso-fotografia-digital.com/blogs-de-fotografia/tips-y-consejos/128-partes-de-tu-camara.html (Consultat el 2 de desembre)

https://www.solerpalau.com/es-es/blog/sensores-movimiento/#:~:text=Los%20sensores%20por%20infrarrojos%2C%20en,unos%20grados%20determinados%2C%20se%20activa. (Consultat el 2 de desembre)

https://eyesynth.com/que-es-eyesynth/ (Consultat el 2 de desembre)

https://www.folkstalk.com/2022/10/pytesseract-tesseract-is-not-installed-with-code-examples.html (Consultat el 2 de desembre)

https://www.crehana.com/blog/transformacion-digital/que-es-opencv/ (Consultat el 3 de desembre)

https://help.ubuntu.com/community/SourcesList (Consultat el 3 de desembre)

http://wiki.ros.org/Distributions (Consultat el 3 de desembre)

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7678785/ (Consultat el 1 de desembre)

https://en.wikipedia.org/wiki/Document_layout_analysis (Consultat el 1 de desembre)

https://www.researchgate.net/publication/343861915_DNN-based_grapheme-to-phoneme_conversion_for_Arabic_text-to-speech_synthesis (Consultat el 1 de desembre)

https://vitalflux.com/wp-content/uploads/2019/07/Neural-Network-Architecture-for-TTS-300x140.png (Consultat el 1 de desembre)

https://thehook.es/en/what-is-text-to-speech-and-how-does-it-work/ (Consultat el 1 de desembre)

https://www.readingrockets.org/article/text-speech-technology-what-it-and-how-it-works#:~:text=Parents%20or%20Schools%3F-,Text%2Dto%2Dspeech%20(TTS)%20is%20a%20type%20of,and%20convert%20them%20into%20audio. (Consultat el 1 de desembre)

https://www.askpython.com/python-modules/python-imread-opencv (Consultat el 28 de novembre)

https://www.jetbrains.com/es-es/pycharm/features/coding_assistance.html (Consultat el 28 de novembre)

https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_gui/py_image_display/py_image_display.html (Consultat el 28 de novembre)

https://www.jetbrains.com/pycharm/ (Consultat el 27 de novembre)

https://learn.microsoft.com/en-us/visualstudio/python/tutorial-working-with-python-in-visual-studio-step-05-installing-packages?view=vs-2022 (Consultat el 27 de novembre

https://anaconda.org/ (Consultat el 26 de novembre)

https://www.freelancer.com/u/afaqahmad100
https://www.python.org/downloads/ (Consultat el 25 de novembre)

https://code.visualstudio.com/docs/?dv=win (Consultat el 25 de novembre)

https://rstudio-pubs-static.s3.amazonaws.com/752943_8e2a5a8a45484f7a9b50b596be47f217.html (Consultat el 25 de novembre)

https://kripkit.com/ransac/ (Consultat el 2 de desembre)

https://www.hwlibre.com/ros/ (Consultat el 2 de desembre)

http://pointcloudlibrary.blogspot.com/2013/12/herramientas-de-la-libreria-pcl.html (Consultat el 2 de desembre)

https://www.hostinger.es/tutoriales/que-es-ubuntu (Consultat el 3 de desembre)

https://es.godaddy.com/blog/que-es-raspberry-pi/ (Consultat el 27 de novembre)

https://www.xataka.com/basics/que-arduino-como-funciona-que-puedes-hacer-uno (Consultat el 25 de novembre)

https://visualstudio.microsoft.com/es/ (Consultat el 25 de novembre)

https://riunet.upv.es/bitstream/handle/10251/83409/Agust%C3%AD%20-%20Uso%20del%20Kinect%20en%20el%20computador%20desde%20el%20punto%20del%20vista%20del%20usuario.pdf?sequence=1#:~:text=libfreenect%20es%20una%20blibioteca%20que,y%20el%20led%20de%20actividad. (Consultat el 24 de novembre)

https://es.wikipedia.org/wiki/OpenCV (Consultat el 24 de novembre)

https://www.tithink.com/es/2018/08/29/framework-o-librerias-ventajas-y-desventajas/ (Consultat el 3 de desembre)

https://creator.nightcafe.studio/ (Consultat el 6 de desembre)

https://www.microsoft.com/en-us/download/details.aspx?id=40278 (Consultat l'1 de desembre)

https://www.chakray.com/es/lenguajes-programacion-tipos-caracteristicas/ (Consultat el 12 de novembre)

Annexos

Annex 1. Yolov52.0

Després de veure el resultat obtingut en el primer entrenament, es fa evident que el dataset original necessita algunes modificacions. Per començar, necessita més imatges on no apareguin cap de les categories que hem de detectar, ja que com podem veure en la Confusion Matrix del model Yolov5s1.0[52]. La majoria de categories eren confoses amb el fons en gran manera. El segon fet que ressalta és la poca precisió que el model té en la categoria de truck. Aquesta falta de precisió es podria resoldre incorporant moltes més imatges de la mateixa categoria. No obstant això, aquest procés és feixuc i ens podria portar setmanes poder-lo dur a terme. En canvi, el problema de la confusió amb el fons és de fàcil solució, ja que només necessitem introduir noves imatges, sense haver-les d'etiquetar (un procés molt lent).

El primer pas a fer és obtenir les imatges. La forma més fàcil de fer-ho és mitjançant Google Maps. Com que el model es dedicarà a fer inferències de carrers, la millor forma de posar-li imatges del que realment és un fons sense res a detectar és passar-li imatges de carrers sense cap objecte d'interès per la detecció. Per tant, ens hem de dedicar a anar navegant pel Google Maps fins a trobar llocs que compleixin aquestes condicions. En el nostre cas hem començat per carrers del nostre poble que sabíem que no hi hauria cap objecte. Aquí tenim alguns exemples:

[52] Figura 40

Figura 53. Exemples d'imatges de "background", on no apareix cap categoria de les que hem de detectar.
Font: Google Maps.

En total hem reunit un total de 52 imatges d'aquest tipus, les quals afegirem al dataset original i aplicarem les mateixes transformacions que les altres imatges. Un cop fet, descarregarem el dataset i tornarem a entrenar-lo amb el nou dataset. No esperem solucionar el problema de la confusió amb el fons, ja que el nombre d'imatges és bastant reduït per poder fer-ho[53], però el nombre d'imatges hauria de ser suficient per notar una disminució de la confusió. Veiem els resultats:

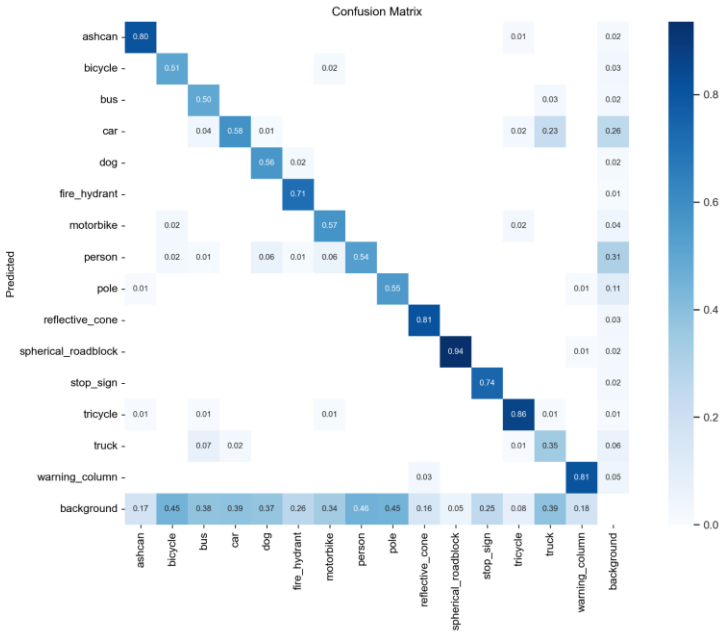


Figura 54. Confusion Matrix del model Yolov5s2.0
Font: Generat pel propi model

[53] Aproximadament, necessitaríem unes 600 imatges per disminuir-lo considerablement

Com podem apreciar, si bé encara hi ha bastanta confusió entre el fons i les categories, es pot veure una lleugera disminució d'aquesta en la majoria de categories. No és un gran descens, però és probable que amb més imatges de "background" es pugui arribar a solucionar o disminuir el problema fins a tenir un percentatge molt petit de confusió.

A conseqüència de la disminució de la confusió amb el fons, també podem observar un augment de l'AP d'algunes categories. A més, també podem afirmar que el mAP del model Yolov5s2.0 ha pujat unes dècimes. Vegem-ho en la gràfica Precision-Recall d'aquest model:

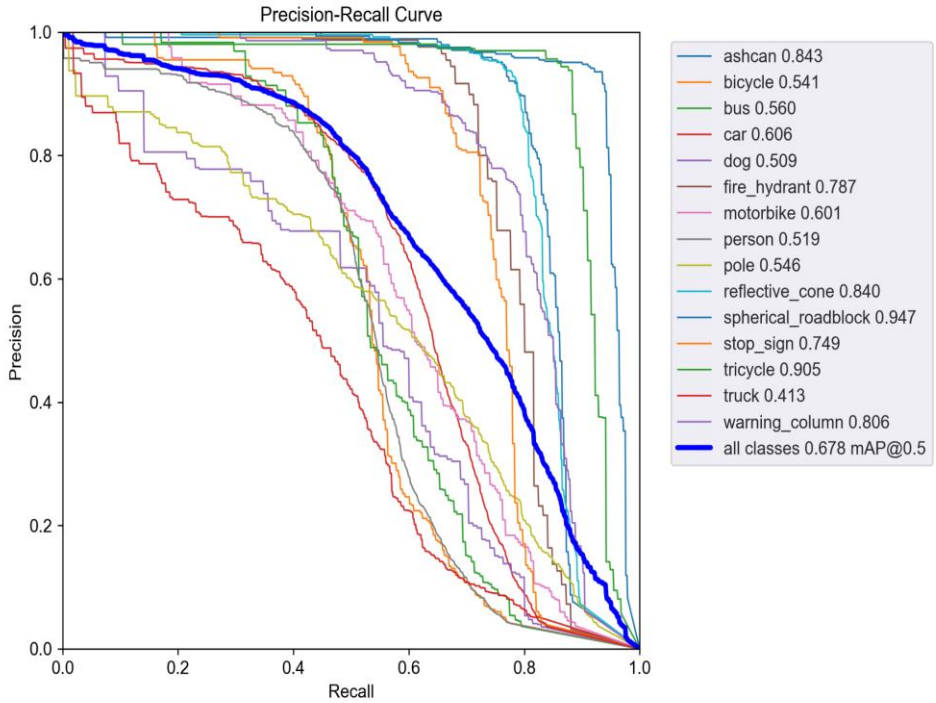


Figura 55. Gràfica Precision-Recall del model Yolov5s2.0

Font: Generat pel propi model.

En el nou model, podem veure una millora en les categories que anteriorment baixaven del 0.5, com és el cas de "truck", "car" i "person". En el cas de car, ara ha pujat fins al 0,606, en el cas de "person", fins al 0.519 i en el cas de truck segueix per sota del 0,5 i es posiciona en el 0.413.

Annex 2. Programació Object Detection

detect.py

```
# YOLOv5 🚀 by Ultralytics, GPL-3.0 license
"""
Run YOLOv5 detection inference on images, videos, directories, globs, YouTube, webcam,
streams, etc.

Usage - sources:
    $ python detect.py --weights yolov5s.pt --source 0                               #
webcam
                                                     img.jpg                         #
image
                                                     vid.mp4                         #
video
                                                     path/                           #
directory
                                                     'path/*.jpg'                    #
glob
                                                     'https://youtu.be/Zgi9g1ksQHc'  #
YouTube
                                                     'rtsp://example.com/media.mp4'  #
RTSP, RTMP, HTTP stream

Usage - formats:
    $ python detect.py --weights yolov5s.pt                 # PyTorch
                               yolov5s.torchscript          # TorchScript
                               yolov5s.onnx                 # ONNX Runtime or OpenCV DNN
with --dnn
                               yolov5s_openvino_model       # OpenVINO
                               yolov5s.engine               # TensorRT
                               yolov5s.mlmodel              # CoreML (macOS-only)
                               yolov5s_saved_model          # TensorFlow SavedModel
                               yolov5s.pb                   # TensorFlow GraphDef
                               yolov5s.tflite               # TensorFlow Lite
                               yolov5s_edgetpu.tflite       # TensorFlow Edge TPU
                               yolov5s_paddle_model         # PaddlePaddle
```

```
"""

if __name__ == "__main__":
    import argparse
    import os
    import platform
    import sys
    from pathlib import Path
    from IPython.display import Audio
    import os
    from speech import Speech
    import torch
    import time, sys
    ##import globals

    FILE = Path(__file__).resolve()
    ROOT = FILE.parents[0]  # YOLOv5 root directory
    if str(ROOT) not in sys.path:
        sys.path.append(str(ROOT))  # add ROOT to PATH
    ROOT = Path(os.path.relpath(ROOT, Path.cwd()))  # relative

    from models.common import DetectMultiBackend
    from utils.dataloaders import IMG_FORMATS, VID_FORMATS, LoadImages, LoadScreenshots,
LoadStreams
    from utils.general import (LOGGER, Profile, check_file, check_img_size, check_imshow,
check_requirements, colorstr, cv2,
                           increment_path, non_max_suppression, print_args, scale_boxes,
strip_optimizer, xyxy2xywh)
    from utils.plots import Annotator, colors, save_one_box
    from utils.torch_utils import select_device, smart_inference_mode

    uu = []
    print("Declaració Variables")
    Speech.speak("Welcome to YOLO5")


    @smart_inference_mode()
    def run(
            weights=ROOT / 'yolov5s.pt',  # model path or triton URL
            source=ROOT / 'data/images',  # file/dir/URL/glob/screen/0(webcam)
```

```
            data=ROOT / 'data/coco128.yaml',  # dataset.yaml path
            imgsz=(640, 640),  # inference size (height, width)
            conf_thres=0.25,  # confidence threshold
            iou_thres=0.45,  # NMS IOU threshold
            max_det=1000,  # maximum detections per image
            device='',  # cuda device, i.e. 0 or 0,1,2,3 or cpu
            view_img=False,  # show results
            save_txt=False,  # save results to *.txt
            save_conf=False,  # save confidences in --save-txt labels
            save_crop=False,  # save cropped prediction boxes
            nosave=False,  # do not save images/videos
            classes=None,  # filter by class: --class 0, or --class 0 2 3
            agnostic_nms=False,  # class-agnostic NMS
            augment=False,  # augmented inference
            visualize=False,  # visualize features
            update=False,  # update all models
            project=ROOT / 'runs/detect',  # save results to project/name
            name='exp',  # save results to project/name
            exist_ok=False,  # existing project/name ok, do not increment
            line_thickness=3,  # bounding box thickness (pixels)
            hide_labels=False,  # hide labels
            hide_conf=False,  # hide confidences
            half=False,  # use FP16 half-precision inference
            dnn=False,  # use OpenCV DNN for ONNX inference
            vid_stride=1,  # video frame-rate stride
    ):
        source = str(source)
        save_img = not nosave and not source.endswith('.txt')  # save inference images
        is_file = Path(source).suffix[1:] in (IMG_FORMATS + VID_FORMATS)
        is_url = source.lower().startswith(('rtsp://', 'rtmp://', 'http://', 'https://'))
        webcam = source.isnumeric() or source.endswith('.txt') or (is_url and not
is_file)
        screenshot = source.lower().startswith('screen')
        if is_url and is_file:
            source = check_file(source)  # download

        # Directories
        save_dir = increment_path(Path(project) / name, exist_ok=exist_ok)  # increment
run
        (save_dir / 'labels' if save_txt else save_dir).mkdir(parents=True,
```

```
exist_ok=True)  # make dir

        # Load model
        Speech.speak("Loading Model from source")
        device = select_device(device)
        model = DetectMultiBackend(weights, device=device, dnn=dnn, data=data, fp16=half)
        stride, names, pt = model.stride, model.names, model.pt
        imgsz = check_img_size(imgsz, s=stride)  # check image size

        # Dataloader
        bs = 1  # batch_size
        if webcam:
            view_img = check_imshow(warn=True)
            dataset = LoadStreams(source, img_size=imgsz, stride=stride, auto=pt,
vid_stride=vid_stride)
            bs = len(dataset)
        elif screenshot:
            dataset = LoadScreenshots(source, img_size=imgsz, stride=stride, auto=pt)
        else:
            dataset = LoadImages(source, img_size=imgsz, stride=stride, auto=pt,
vid_stride=vid_stride)
        vid_path, vid_writer = [None] * bs, [None] * bs

        # Run inference
        model.warmup(imgsz=(1 if pt or model.triton else bs, 3, *imgsz))  # warmup
        seen, windows, dt = 0, [], (Profile(), Profile(), Profile())
        for path, im, im0s, vid_cap, s in dataset:
            with dt[0]:
                im = torch.from_numpy(im).to(model.device)
                im = im.half() if model.fp16 else im.float()  # uint8 to fp16/32
                im /= 255  # 0 - 255 to 0.0 - 1.0
                if len(im.shape) == 3:
                    im = im[None]  # expand for batch dim

            # Inference
            with dt[1]:
                visualize = increment_path(save_dir / Path(path).stem, mkdir=True) if
visualize else False
                pred = model(im, augment=augment, visualize=visualize)
```

```
            # NMS
            with dt[2]:
                pred = non_max_suppression(pred, conf_thres, iou_thres, classes,
agnostic_nms, max_det=max_det)

            # Second-stage classifier (optional)
            # pred = utils.general.apply_classifier(pred, classifier_model, im, im0s)

            # Process predictions,
            for i, det in enumerate(pred):  # per image
                seen += 1
                if webcam:  # batch_size >= 1
                    p, im0, frame = path[i], im0s[i].copy(), dataset.count
                    s += f'{i}: '
                else:
                    p, im0, frame = path, im0s.copy(), getattr(dataset, 'frame', 0)
                p = Path(p)  # to Path
                save_path = str(save_dir / p.name)  # im.jpg
                txt_path = str(save_dir / 'labels' / p.stem) + ('' if dataset.mode ==
'image' else f'_{frame}')  # im.txt
                s += '%gx%g ' % im.shape[2:]  # print string
                gn = torch.tensor(im0.shape)[[1, 0, 1, 0]]  # normalization gain whwh
                imc = im0.copy() if save_crop else im0  # for save_crop
                annotator = Annotator(im0, line_width=line_thickness, example=str(names))
                uu=[" "]
                if len(det):
                    # Rescale boxes from img_size to im0 size
                    det[:, :4] = scale_boxes(im.shape[2:], det[:, :4], im0.shape).round()
                    for c in det[:, 5].unique():
                        n = (det[:, 5] == c).sum()  # detections per class
                        s += f"{n} {names[int(c)]}{'s' * (n > 1)}, " # add to string
                        cnv=str(n.item()) ## numero de deteccions x classe
                        strlvl=(names[int(c)])
                        summery = cnv +" " + strlvl
                        uu.append(strlvl)
                    # Write results
                    for *xyxy, conf, cls in reversed(det):
                        if save_txt:  # Write to file
                            xywh = (xyxy2xywh(torch.tensor(xyxy).view(1, 4)) / gn).view(-
1).tolist()  # normalized xywh
```

```
                            line = (cls, *xywh, conf) if save_conf else (cls, *xywh)  #
label format
                            with open(f'{txt_path}.txt', 'a') as f:
                                f.write(('%g ' * len(line)).rstrip() % line + '\n')

                        if save_img or save_crop or view_img:  # Add bbox to image
                            c = int(cls)  # integer class
                            label = None if hide_labels else (names[c] if hide_conf else
f'{names[c]} {conf:.2f}')
                            annotator.box_label(xyxy, label, color=colors(c, True))
                        if save_crop:
                            save_one_box(xyxy, imc, file=save_dir / 'crops' / names[c] /
f'{p.stem}.jpg', BGR=True)
                # Stream results
                im0 = annotator.result()
                if view_img:
                    if platform.system() == 'Linux' and p not in windows:
                        windows.append(p)
                        cv2.namedWindow(str(p), cv2.WINDOW_NORMAL | cv2.WINDOW_KEEPRATIO)
# allow window resize (Linux)
                        cv2.resizeWindow(str(p), im0.shape[1], im0.shape[0])
                    cv2.imshow(str(p), im0)
                    cv2.waitKey(1)  # 1 millisecond

                # Save results (image with detections)
                if save_img:
                    if dataset.mode == 'image':
                        cv2.imwrite(save_path, im0)
                    else:  # 'video' or 'stream'
                        if vid_path[i] != save_path:  # new video
                            vid_path[i] = save_path
                            if isinstance(vid_writer[i], cv2.VideoWriter):
                                vid_writer[i].release()  # release previous video writer
                            if vid_cap:  # video
                                fps = vid_cap.get(cv2.CAP_PROP_FPS)
                                w = int(vid_cap.get(cv2.CAP_PROP_FRAME_WIDTH))
                                h = int(vid_cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
                            else:  # stream
                                fps, w, h = 30, im0.shape[1], im0.shape[0]
                            save_path = str(Path(save_path).with_suffix('.mp4'))  # force
```

```
*.mp4 suffix on results videos
                            vid_writer[i] = cv2.VideoWriter(save_path,
cv2.VideoWriter_fourcc(*'mp4v'), fps, (w, h))
                        vid_writer[i].write(im0)

            # Print time (inference-only)
            LOGGER.info(f"{s}{'' if len(det) else '(no detections), '}{dt[1].dt *
1E3:.1f}ms")
            with open("data.txt", "w") as data:
                data.write(",".join(uu))
                print("he escrit en data")
            ## aqui acaba el bucle per inferencia

        # Print results
        Speech.speak("Finishing Inference. Turning off")
        t = tuple(x.t / seen * 1E3 for x in dt)  # speeds per image
        LOGGER.info(f'Speed: %.1fms pre-process, %.1fms inference, %.1fms NMS per image
at shape {(1, 3, *imgsz)}' % t)
        if save_txt or save_img:
            s = f"\n{len(list(save_dir.glob('labels/*.txt')))} labels saved to {save_dir
/ 'labels'}" if save_txt else ''
            LOGGER.info(f"Results saved to {colorstr('bold', save_dir)}{s}")
        if update:
            strip_optimizer(weights[0])  # update model (to fix SourceChangeWarning)

    def parse_opt():
        parser = argparse.ArgumentParser()
        parser.add_argument('--weights', nargs='+', type=str, default=ROOT /
'yolov5s.pt', help='model path or triton URL')
        parser.add_argument('--source', type=str, default=ROOT / 'data/images',
help='file/dir/URL/glob/screen/0(webcam)')
        parser.add_argument('--data', type=str, default=ROOT / 'data/coco128.yaml',
help='(optional) dataset.yaml path')
        parser.add_argument('--imgsz', '--img', '--img-size', nargs='+', type=int,
default=[640], help='inference size h,w')
        parser.add_argument('--conf-thres', type=float, default=0.25, help='confidence
threshold')
        parser.add_argument('--iou-thres', type=float, default=0.45, help='NMS IoU
threshold')
        parser.add_argument('--max-det', type=int, default=1000, help='maximum detections
```

```
per image')
        parser.add_argument('--device', default='', help='cuda device, i.e. 0 or 0,1,2,3
or cpu')
        parser.add_argument('--view-img', action='store_true', help='show results')
        parser.add_argument('--save-txt', action='store_true', help='save results to
*.txt')
        parser.add_argument('--save-conf', action='store_true', help='save confidences in
--save-txt labels')
        parser.add_argument('--save-crop', action='store_true', help='save cropped
prediction boxes')
        parser.add_argument('--nosave', action='store_true', help='do not save
images/videos')
        parser.add_argument('--classes', nargs='+', type=int, help='filter by class: --
classes 0, or --classes 0 2 3')
        parser.add_argument('--agnostic-nms', action='store_true', help='class-agnostic
NMS')
        parser.add_argument('--augment', action='store_true', help='augmented inference')
        parser.add_argument('--visualize', action='store_true', help='visualize
features')
        parser.add_argument('--update', action='store_true', help='update all models')
        parser.add_argument('--project', default=ROOT / 'runs/detect', help='save results
to project/name')
        parser.add_argument('--name', default='exp', help='save results to project/name')
        parser.add_argument('--exist-ok', action='store_true', help='existing
project/name ok, do not increment')
        parser.add_argument('--line-thickness', default=3, type=int, help='bounding box
thickness (pixels)')
        parser.add_argument('--hide-labels', default=False, action='store_true',
help='hide labels')
        parser.add_argument('--hide-conf', default=False, action='store_true', help='hide
confidences')
        parser.add_argument('--half', action='store_true', help='use FP16 half-precision
inference')
        parser.add_argument('--dnn', action='store_true', help='use OpenCV DNN for ONNX
inference')
        parser.add_argument('--vid-stride', type=int, default=1, help='video frame-rate
stride')
        opt = parser.parse_args()
        opt.imgsz *= 2 if len(opt.imgsz) == 1 else 1  # expand
        print_args(vars(opt))
```

```
        return opt


    def main(opt):
        check_requirements(exclude=('tensorboard', 'thop'))
        run(**vars(opt))


    if __name__ == "__main__":
        opt = parse_opt()
        main(opt)
```

main.py

```
from speech import Speech
import time, sys
def tts(a):
    print ("def")
    while a == True:
        with open("data.txt", "r") as data:
            print("open")
            arg=data.read()
            if arg == " ":
                arg = "no detection"
            Speech.speak(arg)
            print(arg)
            time.sleep(3.5)
if __name__ == '__main__':
    tts(True)
```

Speech.py

```
from gtts import gTTS
from io import BytesIO
import pygame
import time, sys

class Speech():
```

```
    @classmethod
    def speak(cls, text):
        mp3_file_object = BytesIO()
        tts = gTTS(text, lang='en')
        tts.write_to_fp(mp3_file_object)
        pygame.init()
        pygame.mixer.init()
        pygame.mixer.music.load(mp3_file_object, 'mp3')
        pygame.mixer.music.play()
        time.sleep(1)
```

Annex 3. Entrevista Anna Morancho

Anna M.:"*Bona tarda, sóc l'Anna Morancho, que hem estat parlant abans. M'has demanat que t'expliqui una mica que fem aquí a l'entitat i que després et resolgui unes preguntes, veritat?*"

Jordi: "Exactament. La idea és que aprofitant tot l'estudi que estem fent sobre aquest tema, ens expliquessis una mica com funcioneu per dins, en definitiva com ajudeu a tota aquesta gent, i en acabat m'agradaria plantejar-te un seguit de preguntes que tenim per veure si les podries contestar."

Anna M: "*Doncs mira, nosaltres som una associació sense ànim de lucre i som una entitat que tenim actualment 28 anys d'història. Va néixer el 1994 i l'entitat des dels seus inicis ha tingut una evolució bastant important en funció de les necessitats que s'han anat detectant. Inicialment, es va crear perquè les persones que van fundar l'entitat van veure que les persones amb discapacitat visual, especialment cecs totals, es quedaven a casa tancats en el seu temps d'oci, perquè no hi havia cap activitat ni res programat per a ells. Llavors es va crear aquesta entitat per ajudar-los a fer activitats accessibles, amb l'objectiu que aquestes persones no es quedessin a casa. Això estava fet a base de voluntariat i a poc a poc es va anar professionalitzant i afegint més servis. Els servies que s'han anat afegint, sense que es creessin en*

aquest ordre, actualment sense tenir en compte el servei d'animació sociocultural, que sí que és important, però que amb el temps ens hem adonat que n'hi ha d'altres que són més prioritaris, tenim el servei de psicologia que és important per quant a una persona se la diagnostica d'alguna malaltia o per quan una persona que ja té una malaltia fa una davallada de visió i s'ha d'adaptar a una nova situació, o una persona que es queda cega de cop, o en general una persona que té problemes o té inestabilitat psicològica provocada per la discapacitat visual. Després tenim l'àrea d'integració laboral, que és bastant important, sobretot pels temps que corren, ja que en l'àmbit de la discapacitat les xifres d'atur en les persones amb discapacitat visual sol ser a la inversa de les xifres d'atur normals. Per exemple: Si hi ha un 25% d'atur en la societat, de discapacitat n'hi ha un 75%. En general solen ser xifres molt altes i generalment, encara avui en dia, a les empreses els hi fa por contractar a persones amb discapacitat. Després també tenim la treballadora social que, en realitat, no es ven ve una àrea, però és una assistència que també oferim. Quan una persona arriba nova a l'associació, la treballadora social és la primera persona amb la qual té contacte. Ella valora quina és la seva situació i com els podem ajudar. Un cop avaluat a la persona se la deriva cap al servei que més li convingui: Psicologia, Integració laboral o al servei de Tiflotecnologia que seria tecnologia adaptada. Inclou, per una banda, tot allò que seria instal·lació de programes i també poder fer ús de xarxes socials o programes del tipus Excel, Word, ofimàtica i informàtica en general. Després també tenim el tècnic de rehabilitació bàsica que és el tècnic o la persona que t'ensenya a aprendre tècniques que puguis desenvolupar-te en la teva vida quotidiana en el que seria la teva situació actual, seria el que t'ajudaria a fer servir el bastó, per exemple. T'ensenya a com fer-lo servir, ja que no és només donar-li a una persona un bastó i que s'espavili, li has d'ensenyar tècniques i les coses que ha de tenir en compte quan l'utilitza. Hi ha una sèrie de recursos que has d'aprendre i el tècnic de rehabilitació bàsica t'acompanya fins que els assoleixes. Ja no només és pel bastó, sinó que també altres tipus de coneixements que una persona ha de tenir per poder-se desenvolupar correctament. S'assegura que la persona sigui capaç de poder desenvolupar-se de forma autònoma."

“Aquests són els serveis que oferim. A partir d’aquí, si alguna persona necessita algun altre servei que estigui vinculada amb la discapacitat visual, intentem donar resposta a tot. Aquestes necessitats extres les solem cobrir en forma de tallers i activitats extra en les quals la gent pot participar. Si hi ha molta demanda per part dels nostres socis doncs intentem donar-hi resposta i buscar tallers per solucionar la demanda.”

“A part de donar servei a les persones amb discapacitat visual, també intentem fomentar la investigació en el camp de la salut visual a qui a Catalunya. També intentem donar assistència a països del tercer món. El primer motiu de ceguesa al món són les cataractes, que aquí es poden operar en uns minuts, però allà ja ho sabem que coses que aquí trobem assolidíssimes, allà són coses més complexes.”

Jordi: “Has mencionat que dueu a terme activitats culturals. Què teniu en compte a l'hora d’organitzar aquestes activitats? En què s’han de diferenciar de les activitats normals?”

Anna M: “*En realitat no es diferencien en res sinó que només han de ser accessibles. Es tracta de fer activitats que siguin el màxim accessible. Moltes vegades són activitats que s'organitzen des de fora de l’entitat. Per exemple, si un museu munta una exposició accessible i ens truca, nosaltres ho organitzem tot per poder anar-hi. També es donen casos en el que muntem nosaltres les activitats senceres, si volem anar tots a un cert lloc, doncs ho organitzem i contractem a un guia que ens pugui ajudar. L’associació es va crear perquè no hi havia absolutament res per a persones amb discapacitat visual, en el temps lliure et quedaves tancat a casa teva. Si havies d’anar a comprar sorties, si havies d’anar al metge sorties, però fora d’això no feies res més. Ara la veritat és que hi ha motes ofertes i cada cop hi ha més espais accessibles. En part també és gràcies als nostres socis, els quals a alguns ja els hi està bé sortir del seu àmbit familiar i no dependre de la família per fer activitats. Llavors, saben que nosaltres sempre els hi posem voluntaris perquè puguin fer les activitats. Al cap i a la fi, aquestes activitats els hi donen autonomia personal*.”

Jordi: “La vostra associació quin abast té? Tota Catalunya, només a Barcelona…?”

Anna M: “*L’entitat té la intenció de tenir un abast de nivell Catalunya i sempre que rebem una petició de qualsevol lloc de Catalunya també [bueno, i de fora de Catalunya també si fa falta], però clar, a nivell de recursos econòmics no ens arriba per poder atendre d’igual manera a una persona que és de Barcelona que a una altra que és d’una altra província. Intentem resoldre les peticions, sobretot si són coses molt específiques. Per exemple el tema de psicologia el podem fer a distància, el tècnic de rehabilitació bàsica alguna vegada s’ha desplaçat, però clar, el cost que té això per nosaltres és molt elevat i no ho podem sustentar de forma general, només de forma puntual.*”

Jordi: “Com us financeu?”

Anna M: *“Nosaltres tenim per un part finançament públic, que prové per part d’Ajuntament, generalitat i Diputació i després par part de finançament privat tindríem les quotes de socis, els col·laboradors de l’entitat, que són fixes i que ens van donant diners de forma periòdica. També intentem buscar finançament en empreses, oferint tallers, xerrades, el que podem en general…*”

Jordi:“Ja fora de l’associació, entrem dins de les preguntes que et voldria formular relacionades amb la ceguera. M’agradaria que m’expliquessis els problemes amb què es troben la gent amb problemes de visió en el seu dia a dia. A partir del moment que se'ls detecta el problema de visió. “

Anna M: “*Dins de la discapacitat visual, i d’aquí va el B1, B2, B3 del final del nom de l’associació, hi ha un ventall molt gran de visió. Si t'estàs referint als cecs totals que no veuen res de res és una cosa, però després hi ha les persones amb baixa visió, els quals veuen totalment diferent als cecs totals. Nosaltres hem fet aquesta separació a nivell social, ja que són diferents les necessitats de cada categoria. Els B3 són les persones que tenen més del 50% d’agudesa visual [hi ha un paràmetre que es diu*

agudesa visual que mesura més o menys la qualitat de visió que una persona té]. Del 50 cap al 100 són persones B3, els quals nosaltres considerem que no tenen cap problema de visió. Són persones que poden desenvolupar-se correctament sense necessitats de gent externa. Els B2 són del 50-10. Dins d'aquests tindríem els del 30 -50 % que serien els més lleugers i després del 30 -10 que són els que ja ho estan passant pitjor i després els B1 que legalment són cecs totals, tot i que encara poden arribar a tenir una mica de visió (poca) alguns d'ells."

"Aquests grups tenen necessitats molt diferents. De fet, et recomano que et descarreguis una aplicació que s'anomena "tengo baja visió VR", en aquesta aplicació hi ha diferents formes de tenir baixa visió. Pots veure com veu una persona que té problemes de diabetis, que el que veu són taques negres. Aquesta persona, per exemple, necessitarà unes coses que altres no. Després pots tenir una persona que té la visió borrosa la qual necessitarà altres coses i podrà fer coses que el primer no."

"Cada tipus de problema de visió té les seves coses que has d'aprendre per poder desenvolupar-se. El problema més gran pel desplaçament de les persones cegues és la no normalització de l'arquitectura."

"Els impediments més importants que ens trobem són, entre d'altres, que la gent posa jardiners fora de les portes, queden molt macos, però si vas amb bastó potser detectaràs el test, però a la que te n'adones ja t'has menjat la part d dalt de la planta. També és preocupant els cartells dels restaurants, les terrasses, les obres, la bastida, canvis de relleu no senyalitzats… En l'àmbit de la vida quotidiana, és més o menys el mateix, hi ha gent que és capaç de fer la compra tota sola, d'altres que no, i necessiten a un voluntari de l'entitat per fer-la… El món no està fet per tenir un problema de visió, la veritat… A casa pots anar fent, ja que saps on està tot, però, tot i això, necessites suport per acabar de dur-les a terme de forma completa. Tenir una ajuda et facilita la vida, ho fas més ràpidament i ho fas millor."

“En el món laboral segueix havent-hi molts dubtes de les capacitats d’una persona amb problemes de visió…”

Jordi : “Però la por que tenen les empreses és perquè creuen que no en serà capaç o por perquè no pot donar el màxim de si?”

Anna M: “*Jo crec que és de tot una mica. Tot i això, és un tema de desconeixement per part de l’empresari. Si una persona amb problemes de visió ha pogut passar una carrera, no ha d’haver-hi cap motiu pel qual no podria fer una feina relacionada amb aquesta carrera. Evidentment, les persones amb problemes de visió no ens posarem a treballar en oficis que sabem que no podem fer, ja buscarem aquells que ens siguin més còmodes. També és cert que hi ha oficis que podem fer parcialment. El periodisme tu com a persona cega el pots fer, però si ha s de buscar una foto en concret, necessitarà a algú que t’ho busqui. Dins d’una empresa hi ha moltes posicions, i tot està molt digitalitzat. Hi ha molts programes d’ordinador que et permeten fer accessibles tots els documents de l’empresa. Encara queda molt per fer en els dos mons. El món de l’accessibilitat i en la sensibilització empresarial*. “

Jordi: “Abans has mencionat això dels ordinadors. M’agradaria preguntar-te sobre com ajuda la tecnologia a les persones amb problemes de visió.”

Anna M: “*A nivell tecnològic està ajuden moltíssim. Hi ha moltes aplicacions que no estan específicament dissenyades per a nosaltres, però que ens ajuden a ser més autònoms. També existeixen aquelles que són específiques per a nosaltres. També he de dir-te que actualment el gruix de persones amb discapacitat visual són d’edat avançada i aquestes no són molt afines a la tecnologia. Aquestes persones no estan disposades a aquestes alçades de la seva vida a aprendre a utilitzar la tecnologia.*”

Jordi. “És l'oïda un sentit essencial per a les persones amb problemes de visió?”

Anna M:"*Sí que ho és. Tot i això, no és que s'aguditzi, simplement li prestes més atenció. Les persones amb problemes de visió aprenem a escoltar tot el que passa i a identificar les coses a través de l'oïda. També utilitzem molt el tacte. A Barcelona tenim molts d'aquests senyals. Tenim objectes urbans amb relleu que quan les notes saps que allà hi ha alguna cosa. Pots notar-ho amb els peus o amb el bastó, per exemple. Fins i tot ens fixem en la textura del terra per saber per on anem*."

Jordi: "Creus que en general les ciutats estan ben adaptades pels cecs?"

Anna M: "*No, en general no. Jo tinc la sort de viure a Barcelona i aquesta ciutat és una de les més accessibles en el món respecte d'aquesta disciplina. Tot i això, encara li falten coses. L'ajuntament han de tenir-nos en compte a l'hora de fer coses noves, per fer les coses més accessibles.*"

Jordi: "De forma general, hi ha algunes coses que se situen a la via pública que us molestin a totes les persones amb problemes de visió per poder desenvolupar-vos de forma no ramal?"

Anna M:"*Ara s'està acabant d'aprovar una llei del 2014 la qual dona algunes pistes de les coses que més nosa ens fan. Tot i això, no crec que estiguin totes les coses o, com a mínim, no crec que estigui ben explicat. Cada ciutat és diferent i cadascuna té les possibilitats que té per adaptar el seu espai. Per norma general, no tenim un llistat estàndard amb les coses, però de forma general són útils les senyalitzacions bodo tàctils, les advertències sonores dels semàfors. Estaria bé que també s'assenyalessin els noms dels carrers en braille i que en els transports públics hi hagués unes senyalitzacions en llocs com els transports públics, etc… En definitiva cadascú té els seus objectes que li molesten*."

Aquesta ha estat una transcripció de les parts més importants de l'entrevista. Si voleu escoltar-la sencera, accediu al següent enllaç:

https://drive.google.com/file/d/1el_32O_A5VJ2ghHIMCyuBy4T6d-L7LXH/view?usp=sharing

Annex 4. Model Stairs detection and Stairs modeling

Aquí procedim a l'addició del codi relacionat amb la segona funció de l'aparell. Podem dividir l'inici d'aquest model en dues parts: RGB-D (RGB and Depth *data* capturada pel sensor RGB) i construcció de *stairs*.
En les carpetes següents es pot trobar tot el codi del model.

https://github.com/aperezyus/stairs_detection/tree/master/include
https://github.com/aperezyus/stairs_detection/tree/master/src

El codi del programa per l'extensió d'àudio és el següent:

```
import pyautogui
import time

# while (True):
      # print (pyautogui.position())

import numpy as nm
import pytesseract
import cv2
from PIL import ImageGrab
import pyttsx3
engine = pyttsx3.init()

textarray=[]

def imToString():

    # Path of tesseract executable
    #pytesseract.pytesseract.tesseract_cmd ="tesseract.exe"
    pytesseract.pytesseract.tesseract_cmd ="C:\\Program
Files\\Tesseract-OCR\\tesseract.exe"
```

```
    # change depending on your desired capture area, use the pyautogui
commented out portion above to check xy coords - first set is top left,
2nd send is bottom right. (xtop, ytop, xbot, ybot)
    cap = ImageGrab.grab(bbox =(5, 110, 1335, 1032))

    # Converted the image to monochrome for it to be easily
    # read by the OCR and obtained the output String.
    tesstr = pytesseract.image_to_string(
            cv2.cvtColor(nm.array(cap), cv2.COLOR_BGR2GRAY),
            lang ='eng')
    return(tesstr)

while (True):
      textarray=imToString()

      engine.say(textarray)
      engine.runAndWait()
      time.sleep(1)
      pyautogui.click(x=1315, y=583)
      time.sleep(1)
```

Annex 5. Preparació Kinect

La Kinect utilitzada per la realització d'aquest projecte és la primera versió de la Kinect 360 per Windows. Necessitarem instal·lar els seus drivers tant per Linux com per Windows per poder treballar amb ella.

Drivers per Windows: https://www.microsoft.com/en-us/download/details.aspx?id=40278

Drivers per Linux: http://openkinect.org/
http://openni.org/

Annex 6. Programació Text to speech

```
import cv2
import pytesseract
from speaker import speak
pytesseract.pytesseract.tesseract_cmd='C:\\Program Files\\Tesseract-
OCR\\tesseract.exe'


# Function to convert :: used later
def listToString(s):

    # initialize an empty string
    str1 = ""

    # traverse in the string
    for ele in s:
        str1 += ele

    # return string
    return str1



#Capturing camera feed and when user presses q the feed at that momment
is used further
cap = cv2.VideoCapture(0)
while(True):
    _,img =cap.read()


    cv2.imshow("tee",img)
    k=cv2.waitKey(5) &0xFF



    if k == ord('q'):
        break

## BGR to RGB as pytesseract needs RGB
img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
```

```
print("start")

print(pytesseract.image_to_data(img, lang='eng'))
hImg,wImg,_ = img.shape
Idata=pytesseract.image_to_data(img, lang='eng')
sentence=[""]
for x,b in enumerate(Idata.splitlines()):
    if x != 0:
        b=b.split()
##        print(b)
        if(len(b)==12):
##            x,y,w,h = int (b[6]),int(b[7]),int (b[8]),int (b[9])
##            cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),1)
##
cv2.putText(img,b[11],(x,y),cv2.FONT_HERSHEY_COMPLEX,1,(50,255,50),2)
            wordhai = b[11]
            sentence.append(wordhai)
            sentence.append(" ")
##            print(wordhai)

        ##print(pytesseract.image_to_string(img))
speak(listToString(sentence))
cv2.imshow("tee",img)
cap.release()
##cv2.destroyAllWindows()

from win32com.client import Dispatch

def speak(str):
    speak = Dispatch(("SAPI.SpVoice"))
    speak.Speak(str)


if __name__ == '__main__':
    speak(word234)
```

Annex 7. Entorns de programació

Microsoft Visual Studio

És un entorn de desenvolupament integrat (IDE) disponible per Windows, Linux i macOS i compatible amb diversos llenguatges de programació. Permet desenvolupar aplicacions, webs, etc., que podran ser executades en qualsevol plataforma de Windows. Proporciona al programador facilitat a l'hora de desenvolupar el software.