ANHYDRA:

Joan Marc Samó Rojas, Enric Ferrera González,
Miguel López Manzanares

Abstract— El nostre objectiu amb aquest projecte ha sigut implementar un sistema de visió per computador per controlar un pàrquing, detectant places, vehicles, matrícules, colors i marques dels cotxes mitjançant tècniques clàssiques i d'aprenentatge profund. Aquestes dues parts del projecte han sigut molt diferents, ja que amb les metodologies clàssiques hi ha més treball per arribar al mateix punt que amb les d'aprenentatge automàtic. De tots els obejctius mencionats prèviament hem aconseguit la gran majoria d'ells, només deixant a mitges la detecció de ma marca, que finalment si que hem sigut capaços d'identificar a quina part de la imatge està i retallar-la però ens ha faltat fer la part de fer la correspondència amb la base de dades de les marques.

Keywords—Computer vision, parking management, vehicle detection, license plate recognition, YOLOv5, EasyOCR, classical image processing, CNN, smart surveillance.

1 INTRODUCCIÓ

El nostre objectiu és la creació d'un sistema de control d'un pàrquing. Aquest projecte s'ha lligat amb l'assignatura de Robòtica, Llenguatge i Planificació (102785) de la menció de computació del grau d'enginyeria informàtica de la UAB. Es basa en un vídeo dels cotxes aparcats en bateria, que capta un hipotètic robot quadrúpede de vigilància. Partirem d'un fotograma del vídeo que capta el robot, per exemple:

Fig.1. Imatge original de cotxes aparcat.

La principal motivació d'aquest projecte és implementar les tècniques bàsiques apresses durant el curs en un cas real i explorar només tècniques modernes.

Els objectius d'aquest projecte són:

- La detecció de les places d'aparcament
- La detecció de cotxes aparcats a les places
- L'extracció del color, la matrícula i la marca.

Per tal d'obtenir imatges amb les quals poder entrenar i provar els nostres models siguin clàssics o automàtics vam anar al pàrquing de l'escola d'enginyeria i vam gravar uns vídeos on es veien les múltiples places d'aparcament i els seus respectius cotxes aparcats o si eren places lliures.

1.1 Títol

Després d'estar pensat en múltiples noms que definissin el millor possible el nostre sistema hem pensat el següent: ANHYDRA → Automated Neural HYbrid Detection for Recognizing Automobiles

Eslògan: Detecta, entén i controla el teu pàrquing com mai abans.

2 ESTAT DE L'ART

2.1 Part Clàssica

En aquesta part hem volgut plantejar el nostre projecte des d'una visió més manual, això significa que no hem utilitzat xarxes neuronals o sistemes de detecció avançats basats en aprenentatge automàtic, sinó que ens hem centrat en tècniques clàssiques de visió per computador.

Principalment ens basem en:

- Aplicar transformacions a escala de grisos.
- Aplicar filtres i deteccions de contorns.
- Utilització de transformacions com identificar línies amb el HoughLines.
- Aplicar màscares i regions d'interès per identificar zones concretes.

2.2 Part Moderna

En aquesta part hem volgut plantejar el nostre projecte utilitzant tècniques modernes de visió per computador, utilitzant models d'aprenentatge profund (deep learning) per automatitzar la detecció de vehicles i la lectura de matrícules.

Principalment ens basem en:

- Utilització de models com YOLOv5 per detectar vehicles dins de les imatges.
- OCR modern com EasyOCR per a l'extracció de text d'imatges. El més interessant és que fa servir xarxes neuronals convolucionals.
- Classificació del model del vehicle amb xarxes CNN.

3 PROPOSTA

Es planteja aconseguir els objectius utilitzant mètodes clàssics apressos a classe, però, per certes parts del problema també és planteja una solució amb mètodes moderns.

3.1 Detectar places d'aparcament i cotxes

Per la detecció de les places es planteja utilitzar detecció de voreres de Canny per marca les línies del terra i la transformada de Hough per línies per detectar-les.

Per detectar els cotxes establirem dos mètodes clàssic i modern. Al mètode clàssic farem un retall vertical entre cada parell de línies per trobar els cotxes, i al modern, utilitzarem xarxes neuronals com la YOLO per detectar-los.

3.2 Detectar color

Un cop detectat el cotxe es guardarà com imatge per treballar. Sobre aquesta s'implementarà un KMeans per detectar el grup amb més píxels i es buscarà el seu nom aproximat en una llista de colors coneguts.

3.3 Detectar matrícules

Per tal de poder detectar i identificar els caracters de la matrícula ho hem plantejat de dues maneres que al final del procés

van molt lligades.

D'una banda, tenim la part clàssica, que consisteix a detectar la zona de la matrícula mitjançant cerca de contorns i trobar una àrea en forma de rectangle amb la proporció d'una matricula. Un cop detectada, s'aplica un preprocessat (binarització, morfologia) i es detecten els contorns dels caràcters. Aquests caràcters es comparen amb plantilles prèviament carregades (una per a números i una altra per a lletres) i es reconstrueix la matrícula de manera manual.

Per altra banda, s'utilitza també un enfocament modern basat en EasyOCR, que és una llibreria d'OCR que utilitza xarxes neuronals. Un cop generada la imatge de la matrícula amb els caràcters alineats i preprocessats, aquesta s'envia a EasyOCR per fer una validació automàtica i obtenir el text reconegut.

Aquest enfocament híbrid ens permet validar manualment els resultats obtinguts per plantilla i alhora comprovar si EasyOCR millora la lectura o reforça el que ja s'ha detectat. Això fa que el sistema sigui més robust i precís, i es pugui adaptar a diferents qualitats d'imatge o situacions ambientals.

3.4 Detectar marca

A l'hora de detectar la marca em començat per detectar i enquadrar la posició de la insígnia. Per això em pensat en detectar les zones amb moltes voreres utilitzant Harris i un cop tinguem les diferents zones detectar la de la insígnia.

Un cop trobada i enquadrada la insígnia utilitzarem, de manera clàssica ORB per trobar similituds amb imatges de referència. De manera moderna utilitzarem CNN per classificar les imatges.

4 EXPERIMENTS, RESULTATS I ANÀLISI

4.1 Detectar places d'aparcament

De manera clàssica em utilitzat Canni i Hough per detectar les línies, però, ens em adonat que necessitàvem alguna manera de comptabilitzar el total de línies. Per això em utilitzat cv2.findContours, per comptabilitzar les línies.

A l'hora de provar si funcionava ens em adonat que degut als cotxes i a les ombres dels cotxes no detectàvem correctament les línies verticals del terra. Llavors em decidit utilitzar les línies verticals per comptabilitzar les places.

Ens em trobat un parell de problemes. Primer, a l'hora de detectar les línies, l'asfalt tenia un color semblant i creava línies que no existien. Per solucionar-ho em jugat amb les paràmetres de Canni, augmentant el threshold inferior i superior per acceptar voreres més fortes.

Després, em decidit dibuixar les línies en un fons negre per facilitar el contour.

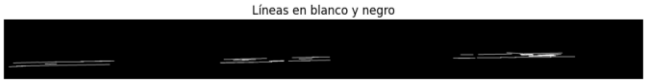

Fig. 2. Líneas en blanc i negre després d'aplicar Canni i Hough.

El problema ara era que el contour detectava massa regions, per tant, teníem que omplir les regions de cada línia de blanc. Per això em utilitzat un close, una operació morfològica, que primer dilata, expandeix les zones blanques i després erosiona, encongeix les zones blanques però manté tancats els petits forats.

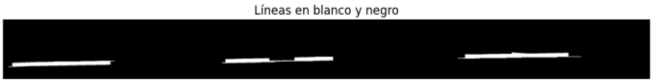

Fig. 3. Líneas després d'omplir-les morfologicament.

Un cop fet això, es comptabilitzen les línies perfectament. Per trobar el número de places d'aparcament, restem un al número de línies horitzontals.

4.2 Detecció de cotxes

De la manera clàssica, detectem l'espai entre contours i ens quedem amb les coordenades horitzontals.

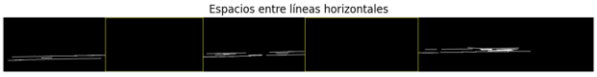

Fig. 4. Detecció d'espai entre contours.

Un cop fet això agafem la imatge original i retallem verticalment la zona on estan els cotxes. Després retallem horitzontalment les regions on hi han vehicles basant-nos en els càlculs previs.

Fig. 5.6. Cotxes detectats amb mètode clàssic.

De la manera moderna, hem utilitzat el model YOLO per fer la detecció dels cotxes a l'aparcament. Vam decidir utilitzar la versió 5 d'aquest model ja que la 8 requeria de llibreries addicionals com Ultralytics o Detectron2, cosa que complicaria més el projecte El primer problema que va aparèixer és que necessitàvem tenir el model en format onnx i al descarregar-lo siguin massa a prop ni massa llunyans, ja que al repositori de git no estava en aquest format. Per tant, hem hagut de descarregar les dependències corresponents (torch torchvision onnx onnxruntime). Aquest pas ha donat molts errors, ja que les versions de onnx i altres llibreries com numpy no eren compatibles. Al final, vam haver de reinstal·lar la majoria de llibreries que fallaven amb una versió de onnx més antiga que no donava error. Un cop descarregada, hem hagut d'exportar el model YOLO al format prèviament mencionat i afegir aquest a la carpeta del projecte.

Un cop programada la funció que donada una imatge detecta els cotxes que hi han a ella, només quedava ajustar els paràmetres d'aquesta fins a que no hi haguessin errors.

Al principi ens passava que detectava molts cops el mateix cotxe degut a que l'àrea mínima que buscava el model era massa petita en comparació al cotxe, el que conduïa a error. També hem hagut d'ajustar el paràmetre que indica la confiança del model, ja que si era massa baix, a vegades detectava objectes amb formes "similars" com petits murs i els marcava com a cotxes. A part d'això hem afegit un altre paràmetre que controla el nivell de soroll permès al model. Aquest ens interessa que sigui baix, ja que sinó molts cops es solapaven les deteccions. Però sense cap mena de dubte el gran salt qualitatiu va ser quan vam afegir paràmetres per controlar el mínim i el màxim de les àrees dels

contorns dels cotxes, perquè no volem que detecti cotxes ni que siguin massa a prop (és informació no rellevant pel projecte) ni massa lluny (pot portar a errors a l'hora de comptar els cotxes a la imatge). Per últim hem implementat dos paràmetres per controlar el mínim i el màxim aspect-ratio que han de tenir els cotxes a l'hora de ser detectats, ja que sinó per la distorsió de la lent als costats de la imatge, si hi hagués un cotxe allà es veuria estirat, i encara amb això el model ho detectaria com a positiu tot i que no està a la zona d'interès de la imatge. Un cop fet tot això, aquest és el resultat final:

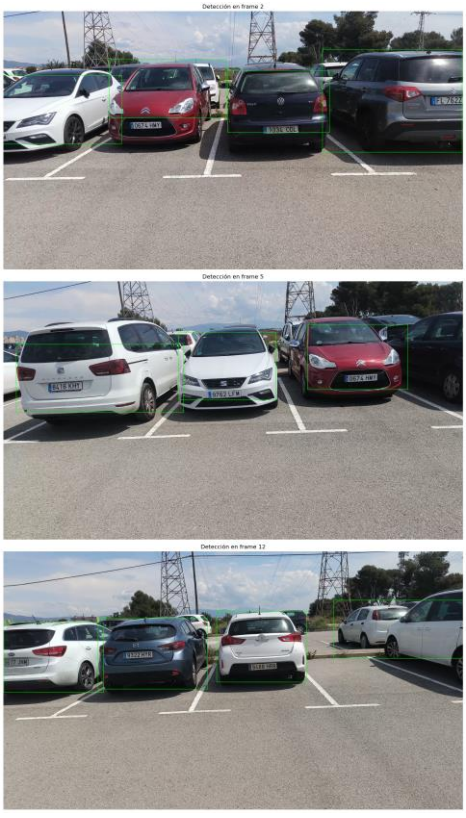

Fig. 7. Cotxes detectats de manera moderna.

4.3 Detecció de color

Utilitzant l'algoritme KMeans amb K=3 i una base 50 colors no s'aconsegueixen un resultats molt bons, detecta molts negres i grisos i no s'aconsegueix apropar al color.

Un cop vistos aquests resultats em implementat diverses millores:

- Filtrar del fons i zones irrellevants
- Provar altres espais de color
- Augment del número de clústers del KMeans
- Ignorar clústers minoritaris (-5%)
- Visualització de barra proporcional als clústers

Després de totes aquests intents s'ha aconseguit millorar notablement la detecció del color del cotxe.

4.4 Detecció de matrícules

Per a la detecció clàssica de matrícules, primer es detecta la zona de la matrícula a partir de contorns. S'utilitzen condicions sobre l'àrea i la relació d'aspecte per filtrar únicament rectangles que siguin de proporcions típiques de matrícules.

Un cop trobada la regió, s'aplica un preprocessat d'imatge amb binarització inversa, tractament morfològic i inversió de colors per fer els caràcters blancs sobre un fons negre.

Després el que fem és detectar els contorns dels caràcters i s'ordenen d'esquerra a dreta. Per a identificar que representa cada caràcter el que fem és que a partir d'una sèrie de plantilles generades anteriorment les compararem amb els caràcters utilitzant la funció cv2.matchTemplate.

Aquest sistema ens ha donat molt bons resultats quan la matrícula està centrada i fins i tot, si està una mica torta. El problema més greu al qual hem hagut de lluitar és amb els reflexos de i els errors en la segmentació i identificació posterior.

Fig. 8.9. Matrícula detectada i lletres detectades

En aquest cas hem pogut detectar el següent:
Caràcters detectats: 7
Matrícula detectada Manual: 6322HYF
Matrícula detectada Automàtica: 9322HYR (confiança: 0.96)

Per tal de fer la detecció moderna s'ha utilitzat el EasyOCR, una llibreria OCR basada en xarxes neuronals convolucionals. Aquesta eina permet llegir text de manera automàtica, tolerant inclinacions, deformacions i variacions de llum.
El primer que hem fet és a partir de la imatge processada del mètode manual li passem els caràcters individuals per tal d'evitar-li el possible soroll ja eliminat anteriorment i aquest amb les seves xarxes neuronals és capaç d'identificar molt millor que el model manual. Sobretot si ens fixem en situacions com matrícules inclinades ombres parcials o text amb problemes de segmentació.

4.3 Detecció de marca

Per la detecció de la marca hem utilitzat Harris per detectar cantonades i punts d'interès a la imatge. Hem hagut de jugar amb els paràmetres per poder recobrir totalment les regions. Em augmentat el block size a l'hora de calcular els gradients locals i em augmentat la sensibilitat a les cantonades k fins a 0.06.

Fig. 10. Taques fetes per Harris

Un cop marcades les diferents zones amb cantonades, em enquadrat les regions i de manera manual em buscat l'alçada i l'amplada de 3 insígnies diferents. Utilitzant aquestes 3 com a referencia, es busquen a les imatges rectangles similars per tal de detectar el requadre amb la insígnia.

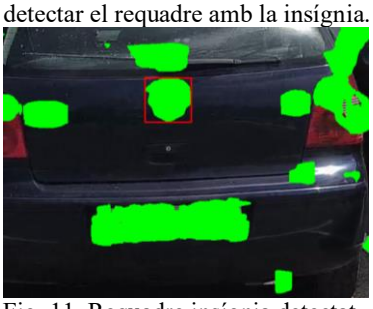

Fig. 11. Requadre insígnia detectat

Un cop trobat el requadre més proper és selecciona com a insígnia i es guarda per al posterior anàlisis.

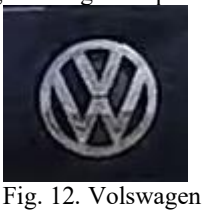

Fig. 12. Volswagen

Queda pendent de traslladar aquesta insígnia a un nom escrit.

5 CONCLUSIONS

Gràcies a aquest projecte, hem après molts conceptes i mètodes nous del camp de la visió per computador que segur que en un futur farem servir. El fet de desenvolupar el projecte seguint dues metodologies diferents (la clàssica i la moderna) ens ha donat una millor perspectiva sobre l'impacte dels models d'aprenentatge automàtic al camp de la visió per computador.

Hem viscut en primera persona la gran quantitat de problemes i complicacions tècniques que hi han a l'hora de fer servir la metodologia clàssica i com hem hagut de fer alguns trucs com retallar les imatges per tal d'aconseguir els resultats esperats. En canvi, amb la metodologia moderna, tot i que també hi han hagut complicacions i reptes a superar ha sigut més ràpida d'implementar. Hem vist la gran ajuda que suposen els models d'aprenentatge automàtic i el gran canvi que han suposat. Respecte als objectius que teníem amb el projecte, creiem que hem superat les nostres expectatives, ja que ens hem sapigut "treure les castanyes del foc" quan ho hem necessitat i aquest és el millor aprenentatge. Hem implementat un sistema de visió per computador capaç de detectar les línies d'un pàrquing, els cotxes que hi han aparcats, les matrícules dels cotxes i els colors dels cotxes.

6 EXTRES

6.1 Reptes Esperats

6.1.1 Condicions ambientals

Un repte important era assegurar-se que el sistema es podria enfrontar a escenes amb llum solar directa, ombres i reflexos. S'ha pogut mig resoldre aplicant tècniques de preprocessament com equalització, binarització o treball amb negatius.

6.1.2 Vehicles mal posicionats

Alguns cotxes aparcaven fora dels límits de la plaça, envaint altres espais. S'ha intentat resoldre detectant les línies de cada plaça i comprovant si el vehicle envaeix l'àrea d'altres.

6.1.3 Lectura de matrícules

Les matrícules borroses, amb angles difícils o amb reflexos van complicar la lectura. Es va millorar la detecció tallant millor la zona d'interès i aplicant EasyOCR.

BIBLIOGRAFIA

[1] https://github.com/ultralytics/yolov5/tree/master
[2] https://github.com/JaidedAI/EasyOCR
[3] OpenCV: cv::text::OCRTesseract Class Reference