Detecció de defectes en impressions d'una impremta

Sergi Madueño Noguer

Resum—En un món cada vegada més automatitzat, on es busca més l'excel·lència i la reducció de costos, cada cop hi ha menys lloc per als controls de qualitat on la intervenció humana és la principal. Ja fa força temps que les capacitats de les màquines van sobrepassar les dels humans en molts camps, com el del control de qualitat, no hi ha manera que un humà pugui competir contra una màquina intentant dur a terme el control de qualitat de la impressió d'una impressora ràpida. Però el que sí que pot fer un humà és -amb els coneixements i les habilitats pertinents- crear la tecnologia (o fent servir la tecnologia actual) i programar un sistema per a que realitzi el control de qualitat per a ell. I d'això tracta aquest treball de fi de grau, fent servir la tecnologia de la que disposem per a controlar la qualitat d'impressió d'una impressora de bobina, amb l'ajut d'una càmera connectada a un PC, Python i OpenCV.

Paraules clau—Impressió, defecte, arts gràfiques, control, OpenCV, visió per computador, xarxes neuronals, homografia, punts característics, data set, binarització, blob, contorn.

Abstract—In a ever more automated world, where execellence is the goal, among costs reduction, less space is left for human powered quality control. Long ago human capabilities were clearly overtaken by machines (at least in many fields as quality control), a human cannot compete with a Machine when trying to control the possible defects on a fast printing machine. But a human can -with the proper knowledge and skills- create the technology (or using the current technology) and program a system to do that quality control for him/her. And that is what this final degree project is about, using the technology we have at hand this days to control the printing quality on a paper roll, with the aid of a digital camera connected to a PC, Python and OpenCV.

Index Terms—Print, defect, graphic arts, control, OpenCV, computer vision, neural networks, homography, key points, data set, binarization, blob, contour.

1 Introducció - Context del treball

Amb l'arribada des telèfons «intel·ligents», les tauletes i els llibres electrònics, molts van ser els qui van predir o fins i tot profetitzar la desaparició de qualsevol tipus d'impressió sobre paper (molts d'ells probablement per a promocionar la venta dels seus dispositius electrònics), el que es coneix com a «*paperless office*».

El fet és que certs tipus d'impressions sobre paper se'n van ressentir (particularment la dels diaris), però hi ha un sector de la impressió que no ha estat amenaçat per la irrupció d'aquests aparells electrònics, i aquest és el sector de la impressió dels prospectes dels medicaments.

El prospecte d'un medicament[1] és un full de paper on es dóna informació molt important sobre el medicament, entre tota la informació que hi podem trobar hi ha quin tipus de medicament és i la seva utilitat (els tipus de malalties per a les quals està recomanat), com s'ha de conservar el medicament, els possibles efectes adversos (en cas de tenir-ne), advertències sobre possibles incompatibilitats si s'estan prenent altres medicaments, com desfer-se de la resta del medicament un cop acabada la prescripció del medicament i finalment (i potser la part més important) la posologia, que és la forma en que s'ha d'administrar un medicament.

Aleshores, com en qualsevol impressió (ja sigui amb una impressora domèstica o a una impremta), poden haver-hi errors com ara que apareguin taques en la impressió o també que s'arrugui el paper i la impressió surti malament.

- *E-mail de contacte: sergi.maduenon@autonoma.cat*
- *Menció realitzada: Computació*
- *Treball tutoritzat per: Coen Antens (Centre de Visió per Computador)*
- *Curs 2023/24*

[1] Ús d'un prospecte de medicament: https://www.farmaceuticonline.com/prospecte-medicament/

En el cas de que sortissin taques a la impressió, això podria ser molt preocupant, mentre que una taca en un espai blanc de la impressió (part no impresa del document) només tindria efectes visuals, si aquesta mateixa taca aparegués a una zona important com són les contraindicacions o, sobretot, a la manera d'administrar-lo (posologia), podria fer que la gent es prengués una dosi inadequada del medicament, i això pot posar en risc la vida dels usuaris del medicament. Per exemple imaginem-nos que a un medicament on ens hauríem de **prendre 3 càpsules cada 24 hores** hi aparegués una taca sobre el 4 i nosaltres llegíssim **prendre 3 càpsules cada 2 hores**.

A la impremta on s'estan imprimint els prospectes, actualment tenen el següent flux de treball:

1. Fent servir una impressora digital imprimeixen els prospectes en unes bobines de paper.
2. Un cop s'acaba la bobina agafen una mostra de la impressió (suposadament el darrer prospecte imprès) i l'escanegen.
3. Fent servir un programa que compara imatges (Image Compare) fan la comparació entre la imatge original (el PDF) i la imatge escanejada.

D'aquesta manera només es realitza un únic control de qualitat per cada bobina de paper impresa, el que resulta ser una mostra molt petita respecte a la quantitat de prospectes que hi caben a una bobina sencera; segons aquest control de qualitat, si la mostra escanejada es pot donar per bona, també ho serà la resta de prospectes de la bobina.

Aleshores, la utilitat d'aquest TFG és la de poder detectar aquests errors a mesura que van sortint les impressions, de manera que es vagin comparant els fulls impresos amb el document original, i quan es detecti algun error avisar per a que es puguin prendre mesures correctores. Això faria que:

1. No arribessin prospectes mal impresos als usuaris dels medicaments.
2. Que es detectés quan hi ha algun error i evitar haver de reimprimir els treballs.

Això s'aconseguiria fent uns canvis al flux de treball. S'hauria de col·locar una càmera digital amb la seva font d'il·luminació de manera que pogués capturar una imatge de cada prospecte imprès; cada imatge capturada es posaria en correspondència amb la imatge original i es compararia per a indicar si la qualitat de la impressió és bona.

2 OBJECTIUS

En el món de les arts gràfiques acostuma a haver-hi un alt control de qualitat sobre tot el procés, des del disseny gràfic, la maquetació d'un document, la correcció ortogràfica i d'estil, les proves de color i la preparació dels documents per a ser enviats a una impremta per a que facin les planxes d'impressió, etc. Però tot i així sempre pot haver-hi alguna cosa que s'escapi del control de qualitat.

En el cas d'aquest TFG només ens interessaran els errors que es puguin produir en la fase de la impressió, deixant de banda els errors que es puguin haver donat en els processos anteriors (donem per fet que aquests errors no hi són presents en aquesta fase).

Aleshores la llista d'objectius que m'he proposat és:

1. La captura de la imatge de la impressió un cop surt de la màquina d'impressió, en el meu cas fer captures d'unes mostres ja impreses.
2. Preparar un algorisme per a posar en correspondència la imatge escanejada amb la imatge rasteritzada del document original, buscar els errors i dibuixar el seu contorn.
3. Preparar un data set.
4. Comprovar els resultats imatge correcte vs imatge amb defectes.

3 METODOLOGIA

Les metodologies de la Enginyeria del Software estan pensades principalment per a equips de diverses persones, però com en el TFG només hi ha una persona treballant-hi, aquests mètodes s'han d'adaptar a aquesta peculiaritat.

Per al TFG he pensat en seguir la metodologia de treball SCRUM, en la qual jo seré l'«**equip**» de treball i el tutor farà de **Product Owner**; com les trobades o reunions obligatòries es fan aproximadament cada 4 o 5 setmanes, he pensat que fóra una bona manera de que cada Sprint durés això, el temps entre reunió i reunió, de manera que pugui mostrar el meu progrés al tutor i concretar què pensa que hauria de tenir per al següent Sprint (tot i que aquesta durada pot ser que es vagi modificant a mesura que el TFG avança).

4 PLANIFICACIÓ

En la metodologia SCRUM es discuteix entre el Product Owner i l'equip què és el que el primer vol que estigui fet un cop acabat el següent sprint i l'equip determina què és el que es pot comprometre de tenir acabat durant aquest període. Com es pot veure, essent un mètode de treball àgil, no s'acaba d'adaptar a una planificació a llarg termini, on poguem donar unes dates ben específiques per a cada tasca. Tot i així he fet una estimació, tot tenint en compte la llista d'objectius marcada i la durada del TFG.

Per a dur un control del projecte he pensat en fer servir l'eina de Microsoft Project, ja que aquesta va ser la que vaig fer servir quan cursava l'assignatura de Gestió de Projectes.

Per a un major detall de la planificació mirar l'annex.

5 ESTAT DE L'ART

El control de qualitat a una impremta és essencial ja que d'ell en depèn -en molta part- el prestigi d'una empresa; si s'entrega un treball mal imprès dona molt mala imatge ja que el client pensarà que en aquesta empresa no s'hi pot confiar, encara que després li tornin a imprimir bé.

Un mal control de qualitat o l'absència d'aquest pot fer que els errors a la impressió passin desapercebuts, ocasionant això haver de tornar a fer les comandes de paper; l'espera fins que es puguin posar en marxa les màquines d'impressió un altre cop (ja que sempre s'intenta tenir les màquines parades el temps més curt possible per a augmentar-ne la productivitat), degut a que molt probablement en el moment que es detectin els errors a les màquines d'impressió ja hi hagi un altre treball imprimint-se.

Tot això, a més de la mala imatge que dona l'empresa són costos que aquesta ha d'assumir, començant pel paper que haurà de pagar com també les hores que les seves màquines d'impressió no estaran generant beneficis, per no esmentar que el client no estarà satisfet ja que la impremta no haurà complert amb els terminis d'entrega. És per això que s'ha de tenir uns bons estàndards de qualitat.

Tradicionalment, en la impressió es feia la impressió d'uns quants exemplars i es feia un control de qualitat sobre la densitat del color, el balanç cromàtic, que no hi haguessin defectes a la impressió, el guany de punt, el trapping i sobretot el color. Aleshores si tot sortia bé, es continuava amb la impressió, realitzant controls aleatoris durant la tirada de tot el treball d'impressió.

El trapping és una tècnica que es fa servir a la preimpressió per a combatre un efecte no desitjat en una impressió quan hi ha errors de registre entre les planxes dels diferents colors. Posem que tenim un text en color magenta sobre un fons de color cian (com en el cas de la Figura 1) de manera que el text no superposi el fons (aleshores la planxa del cian tindrà una part retallada corresponent a les lletres de color magenta), doncs si es produeix un error de registre tindrem que es veurà una part sense tinta (mostrant el color del substrat d'impressió, generalment blanc). Per a que això no passi es fa servir el trapping o rebentat que consisteix en estendre una mica la impressió dels objectes que s'imprimeixen sobre aquest altre color de fons de manera que, tot i que hi hagi una mica de sobreimpressió, no es vegi aquesta part blanca no desitjada.

Figura 1 Mostra de trapping o reventat.

El guany de punt és quan la mida de la trama a la impressió resulta més gran que s'esperava, de manera que es produeix que tots els colors apareguin més foscos.

Però aquest tipus de control de qualitat era totalment manual, on la presència d'un operari especialitzat era essencial; degut a la velocitat d'aquestes impremtes (cada cop més ràpides) feia que fos impossible dur a terme un control de qualitat basat en la visió humana, per això van ser necessaris els controls de qualitat automatitzats.

Més endavant, en els darrers 20 anys s'ha anat fent servir la visió per computador clàssica (que seria la que estem realitzant en aquest TFG), on se situa una càmera digital que va fent captures de la impressió i fa la comparativa amb un original per a determinar si hi ha errors a la impressió i indicar-ho a un operari per a que realitzi mesures correctores per a acabar amb els errors.

Però en els darrers 10 anys ja s'ha començat a fer servir la intel·ligència artificial per a detectar els errors a la impressió.

Mentre que certs errors a la impressió són temporals, altres, deguts per exemple a errors en la màquina d'impressió o planxes d'impressió que s'estiguin gastant, continuaran existint durant tota la impressió. La visió per computador tradicional era capaç de detectar els errors però no de classificar-los, ja que aquests poden tenir moltes formes i mides diferents, de manera que sempre marcarà on hi ha un error però no ens dirà de quin tipus es tracta (si és temporal o no). Entenem per visió per computador tradicional a aquells sistemes de visió artificial que no fan servir la intel·ligència artificial.

Per això és interessant tenir classificadors que puguin discernir sobre quin tipus de defecte hi ha a la impressió i poder analitzar les causes del defecte. Així es fa servir el Deep Leaning per a entrenar una xarxa neuronal convolutiva.

A partir d'unes mostres d'entrenament s'entrena el model de xarxa neuronal per a que extregui les característiques dels defectes d'impressió. En la següent fase es fan servir diferents models de sobremostreig per a entrenar un classificador basat en una Support Vector Machine. Un problema en les mostres d'entrenament és que no tots els tipus de defectes tenien el mateix nombre de mostres, ja que és molt difícil obtenir una gran quantitat de mostres de tots els tipus de defectes d'impressió, i amb aquest sobremostreig es pot augmentar el nombre de mostres dels tipus de defecte que menys en tenen i així millorar el rendiment del classificador.

En el paper que em va enviar el meu tutor, van classificar els errors d'impressió en 4 grans grups segons la brillantor, la forma i el color, de manera que van quedar quatre grups o classes de defectes, defecte clar sobre fons fosc, defecte fosc sobre fons clar, defecte de ganivet (un error en el qual apareix una línia que no s'ha imprès en una sola

planxa de color) i la darrera que és desviacions de color. Ho podem veure a la Figura 2.

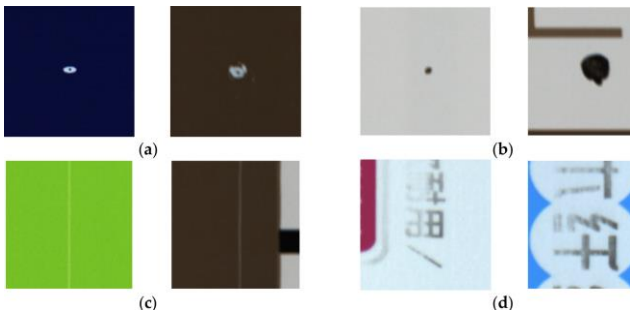


Figura 2 Catàleg de defectes inicial: (a) defecte clar; (b) defecte fosc; (c) defecte de ganivet; (d) defecte de desviació de color.

Els defectes d'impressió poden ser causats per falles als equips d'impressió, degut als materials d'impressió, o a la tecnologia d'impressió. Per a analitzar d'una manera més acurada les causes dels defectes d'impressió van agafar i van afinar els tipus de defectes i el que al principi eren només 4 grups de defectes van pujar-ho fins a 11 tipus de defectes. Als 4 tipus originals (clars, foscos, ganivet i desviació de color) se'ls hi van afegir 6, el crystal, en punt, sense impressió, tacat, sobreimpressió, trailing; però del tipus de defecte que en dèiem fosc sobre clar es va dividir en dos tipus, el de la pols de paper i el de la tinta (causat per baixa viscositat de la tinta). Podem observar els diferents tipus a la Figura 3.

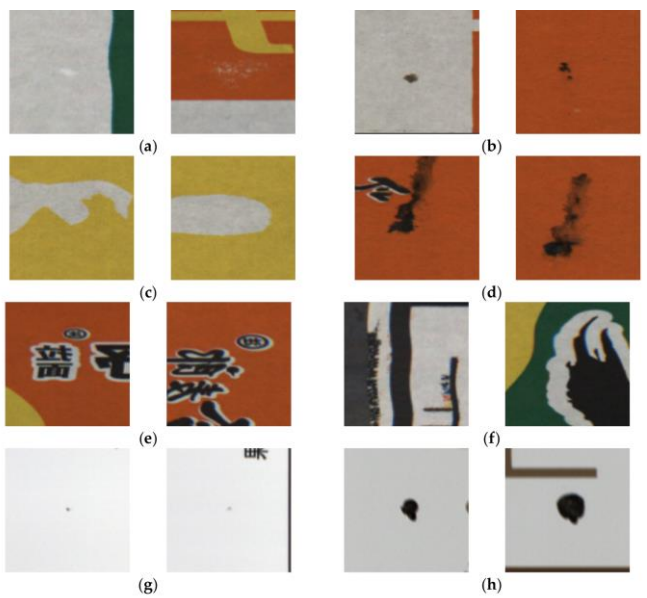


Figura 3 Nous tipus de defectes: (a) crystal; (b) punt; (c) sense impressió; (d) tacat; (e) sobreimpressió; (f) trailing; (g) pols de paper; (h) tinta.

A la següent fase, un cop ja estan entrenats tant la xarxa neuronal i el classificador, s'obtenen els resultats de la classificació fent servir les mostres de test.

A part de fer la recerca sobre l'estat de l'art davant d'un ordinador, també vaig poder assistir en la fira Advanced Factories aquest abril. Es tracta d'una fira orientada a les empreses i centrada en la visió per computador i la intel·ligència artificial.

Tot i que no vaig trobar cap estand on estiguessin promocionant algun sistema per a detectar errors en les impressions, sí que n'hi havia molts que feien servir la visió per computador amb intel·ligència artificial per a detectar errors en la producció de les fàbriques.

Un exemple és el d'una empresa de visió per computador de Girona que es diu Indeep amb qui vaig estar parlant (mentre hi parlava a l'estona també vaig coincidir amb el tutor Coen i resulta que entre ells ja es coneixien) i que tenien un sistema capaç de reconèixer quan hi havia una «contaminació» d'algun producte en una safata on no hi havia de ser. Per exemple a la Figura 4, on hi ha un conjunt de safates amb fruits secs, aleshores el software de Indeep detecta i marca quins fruits sec hi ha a cada safata.

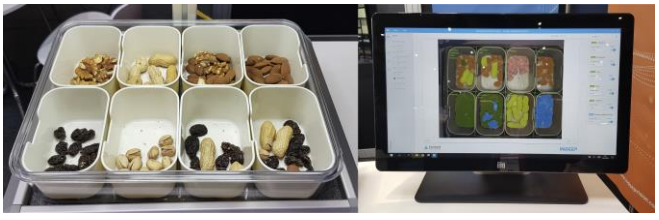


Figura 4. Software de Indeep per a detectar productes no desitjats.

5.1 Explicació del flux de treball

Un cop hem capturat una imatge d'una impressió, hem d'aconseguir casar aquesta imatge amb la imatge de referència que tenim del PDF original (el fitxer del qual s'ha fet la impressió), això és, les hem de posar totes dues en correspondència, de manera que si posem la imatge capturada

Figura 5. Esquema simplificat del flux de treball.

sobre l'original aquestes dues coincideixin. Un esquema simplificat del nostre flux de treball seria el de la Figura 5.

1- La impressora imprimeix el document.
2- Aquest document és capturat per la càmera industrial.
3- Aquesta captura és enviada al PC que s'encarrega del control de qualitat.

La manera de fer tradicional amb un programa de tractament d'imatges (com per exemple Photoshop) seria obrir la imatge original, enganxar-li la imatge capturada (a una altra capa), rotar la capa de la imatge capturada de manera que tinguessin el mateix angle, escalar la imatge capturada per a que tinguin la mateixa mida i finalment moure-la (translació) per a que estiguessin en correspondència.

Però això és inviable, de cap manera es pot fer això de manera manual; aleshores hem de veure de quina manera podríem realitzar aquesta tasca automàticament.

5.1.1 Coordenades homogènies

En el món dels gràfics per computadora i la visió per computador es fan servir el que s'anomenen coordenades homogènies o coordenades projectives.

En aquestes coordenades el que fem és afegir una dimensió més al nostre espai vectorial, per exemple, les imatges tenen dues dimensions que denotem per **x** i **y**, cada punt de la imatge està definit per les seves coordenades cartesianes com ara el punt **(13, 8)**, doncs aquest mateix punt en coordenades homogènies el podem definir amb les coordenades **(13, 8, 1)**. Hem afegit una nova dimensió **z** a la qual li hem donat un valor arbitrari **w = 1**, però podria haver estat qualsevol altre valor (menys **w = 0**); a aquest valor que li hem donat a **w** se li diu escalat i quan passem a coordenades homogènies hem de multiplicar els valors tant de **x** com de **y** per aquest valor, d'igual manera que quan volem retornar de cord. homog. a cart. haurem de dividir cada component per aquest **w**, s'escolleix normalment **w = 1** ja que simplifica molt els càlculs.

Tenim el punt **p** $\begin{pmatrix}13\\8\end{pmatrix}$ en cord. cart. i el volem en cord. homog., aleshores haurem de multiplicar cada component del punt **p** per aquest valor pel valor d'escalat (p. e. **w = 2**) i tindrem $\begin{pmatrix}13\times 2\\8\times 2\\2\end{pmatrix}=\begin{pmatrix}26\\16\\2\end{pmatrix}$, amb això podem veure que un punt en cord. cart. pot tenir inifites representacions en cord. homog., tot dependrà del valor que li volem donar a **w**, el nostre punt $\begin{pmatrix}13\\8\end{pmatrix}\equiv\begin{pmatrix}26\\16\\2\end{pmatrix}\equiv\begin{pmatrix}130\\80\\10\end{pmatrix}$; un punt i els seus escalats són el mateix punt, ho podem veure com una recta que passa pel punt $\begin{pmatrix}0\\0\\0\end{pmatrix}$ *i també pel punt p* $\begin{pmatrix}13\\8\\1\end{pmatrix}$.

Un avantatge de l'ús d'aquest sistema de coordenades és la facilitat de fer servir matrius per a realitzar transformacions com escalats, translacions, rotacions, etc., només multiplicant els punts d'una imatge per una matriu 3 x 3, així com la possibilitat d'apilar les transformacions, com veurem en el següent exemple.

Tenim una imatge a la qual li volem fer una rotació, un escalat i després una translació, aleshores primer haurem de fer una translació de manera que la imatge quedi centrada a l'origen de coordenades, farem servir la matriu T_1, després aplicarem la matriu de rotació R_1, tot seguit aplicarem la matriu d'escalat E_1, tot seguit desfarem la primera translació aplicant la matriu T_2 per a finalment aplicar una darrera matriu de translació T_3, això quedaria de la següent manera:

$$\begin{pmatrix}1&0&15\\0&1&20\\0&0&1\end{pmatrix}\times\begin{pmatrix}1&0&5\\0&1&5\\0&0&1\end{pmatrix}\times\begin{pmatrix}3&0&0\\0&3&0\\0&0&1\end{pmatrix}\times$$

$$\times\begin{pmatrix}\cos\theta&-\sin\theta&0\\\sin\theta&\cos\theta&0\\0&0&1\end{pmatrix}\times\begin{pmatrix}1&0&-5\\0&1&-5\\0&0&1\end{pmatrix}\times\begin{pmatrix}x\\y\\1\end{pmatrix}$$

La seqüència d'operacions que tenim just a dalt és la següent: $T_3 \times T_2 \times E_1 \times R_1 \times T_1 \times p$

5.1.2 Homografia

Al muntatge final la càmera estarà fixa i prendrà captures de les impressions de manera que en cada captura la part impresa quedi al mateix lloc dins la captura, però tot i això pot ser que quedi lleugerament desplaçada aquesta part impresa en cada captura.

Això segons el muntatge final, però en les captures que he fet, amb una càmera que es trobava fixa, el que no hi havia manera era fer que en cada captura quedés la part impresa al mateix lloc de la captura. La distància de la càmera a la impressió era sempre la mateixa, però no la part del prospecte que entrava dins de cada captura ni tampoc l'angle en el qual feia cada captura.

Així que necessitava alguna manera de fer que totes les captures concordessin, i això ho vaig poder resoldre amb l'ajut de les homografies, també conegudes com a transformacions de perspectiva.

Quan fem una captura amb una càmera estem agafant una escena en l'espai de 3 dimensions i l'estem projectant al sensor de la càmera que té 2 dimensions.

Agafem i fem una foto frontal a un quadre que només té rectes paral·leles com el de la Figura 6 esquerre, tot seguit ens desplacem cap a l'esquerra i n'agafem una nova captura però d'una altra perspectiva (ara ja no perpendicular al quadre sinó d'un costat) com a la Figura 6 dreta.

En projecció perspectiva una línia en l'espai 3D sempre es projectarà com una línia, i les línies que són paral·leles al sensor continuaran sent paral·leles a la imatge en 2D, això sí, la resta de línies que eren paral·leles al món 3D ja

no ho seran a la imatge. Com ja sabem, mentre un objecte es trobi més lluny de la càmera més petit semblarà, cosa que es pot observar a l'anterior il·lustració, on la part dreta del quadre es veu més petita (en aquest cas també veiem com ja no es conserven els angles ni la relació d'aspecte dels quadrats originals).

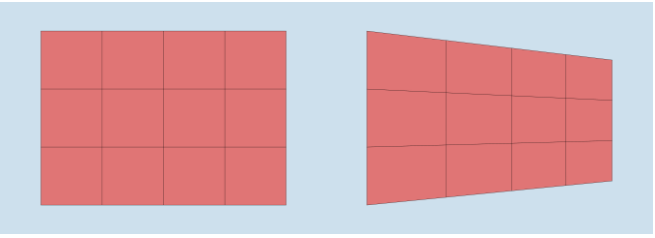


Figura 6 Vista frontal i vista en perspectiva.

Per a explicar el que és l'homografia partirem preguntant-nos si hi hauria una manera d'obtenir la imatge de Figura 6 esquerre a partir de la imatge de la Figura 6 dreta, i en cas afirmatiu de com es podria aconseguir.

La resposta és que sí (d'altra banda ja ni esmentaríem el problema), i és fent servir les homografies (que també es coneixen com transformació de perspectiva).

Al camp de la visió per computador quan tenim dues imatges d'un mateix pla (en el nostre cas serien la imatge original i la capturada), encara que s'hagin agafat des de diferent perspectives, diferent distància (escalat), hi ha una relació entre elles, una *homografia*. Una homografia és una matriu de transformació de perspectiva de mida 3 x 3 que ens porta d'un pla a un altre, d'un punt del pla A ens donarà les coordenades del mateix punt al pla B.

$$\begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} x_s \\ y_s \\ 1 \end{bmatrix}$$

Equació 1 Homografia

A l'equació matricial Equació 1 x_s *i* y_s són les coordenades dels punts d'origen (els punts de la imatge capturada a la Figura 7 dreta on està torta); x_d *i* y_d són les coordenades de la mateixa imatge després de multiplicar-los per la matriu homografia, on la perspectiva ja està corregida i la imatge no es troba torta (Figura 7 esquerra). Els elements h_{13} *i* h_{23} controlen la translació dels punts; h_{11}, h_{12}, h_{21} *i* h_{22} controlen la rotació; h_{11} *i* h_{22} controlen també l'escalat, i finalment h_{31} *i* h_{32} controlen la perspectiva. En aquesta equació hem establert el factor d'escalat **w = 1**, però pot ser que no el sabem i que tinguem Equació 1 de la següent manera, on no coneixem aquesta **w**.

$$w\begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} x_s \\ y_s \\ 1 \end{bmatrix}$$

Equació 2

Aquestes h_{11} *fins* h_{32} tindran el mateix valor per a cada punt que volem transformar d'un pla a l'altre, però **w** serà diferent per a cada punt, de manera que per a calcular la homografia tindrem les 8 incògnites de h_{11} *fins* h_{32} més una més (cada **w**) per a cada punt.

Si tenim 4 punts tindrem **8 + 4 x 1 = 12** incògnites; aleshores si cada punt ens genera 3 equacions, tindrem **4 punts x 3 equacions per punt = 12** equacions per a 12 incògnites, així podrem trobar la homografia.

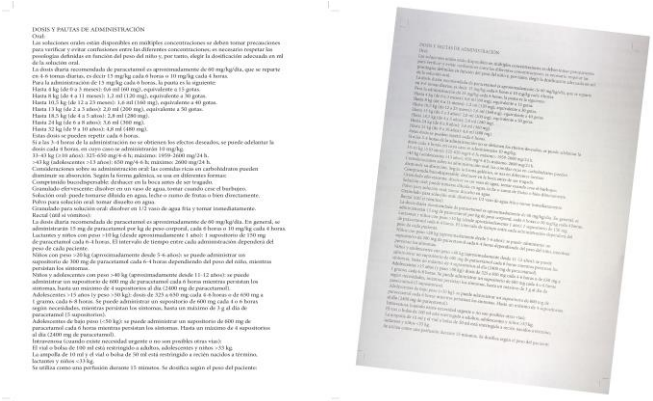


Figura 7. (esquerra) imatge original; (dreta) imatge capturada.

Aleshores, com acabem de dir, per a calcular aquesta homografia ens faran falta com a mínim 4 punts diferents que es trobin a ambdues imatges, 4 és el mínim tot i que quants més punts tinguem més robusta serà aquesta matriu. A la Figura 7 tenim la imatge original (esquerra) i la imatge capturada (dreta, exagerant l'angle de la captura per a demostrar l'eficiència de l'homografia).

5.1.3 SIFT i punts característics

Un punt característic podem dir que és una àrea d'una imatge que és especialment valuosa (variació de la brillantor, o del color, etc.) de manera que la fan única i que ens pot servir per a aparellaments (matching). També ha de tenir una posició ben definida a la imatge i ha de tenir unes característiques que permetin calcular-ne una firma pròpia (característiques que només tingui aquesta àrea). A més ha de ser invariant a l'escalat i la rotació així com als canvis en la il·luminació. Podem dir que un punt característic ha de ser únic dins la imatge.

Com hem dit al punt anterior, per a calcular la homografia que ens permeti anar o associar els punts d'una imatge als mateixos punts a una altra imatge, necessitarem de com a mínim 4 punts que coincideixin a les dues imatges.

Aleshores, com podem fer per a trobar aquests 4 o més punts característics dins les imatges?

La resposta és fent servir l'algorisme SIFT (scale-invariant feature transform), un algorisme de visió per computador que ens serveix per a trobar punts característics dins d'una imatge (Figura 9). Aquest algorisme va ser inventat per l'informàtic canadenc David G. Lowe al 1999.

Figura 9. Punts característics entre l'original i la captura

Aquest algorisme SIFT es fa servir per al reconeixement d'objectes a les imatges, la navegació o moviment de robots, la unió d'imatges (coneguda com stitching) per a fer panoràmiques (com per exemple les que es fan amb els telèfons mòbils), etc.

Quan tenim dues imatges diferents però d'un mateix objecte, pot passar que tinguin diferent mida (diferent escala), diferent orientació, diferent brillantor i diferent il·luminació. Aleshores, qualsevol detector de punts d'interès ha de poder compensar, atenuar o fins i tot eliminar aquestes variacions, i l'algorisme SIFT és capaç d'atenuar o eliminar aquestes variacions; així podrà aparellar una característica d'una imatge en una altra imatge.

A la Figura 10 podem veure 4 punts característics concordants tant a la imatge de l'esquerra com a la de la dreta.

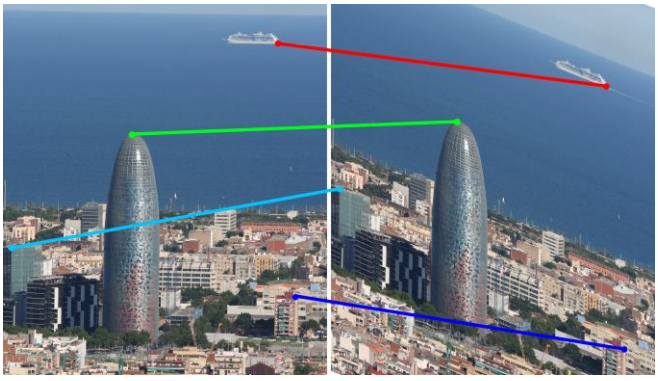


Figura 10. Exemple de correspondència.

5.2 Càlcul de l'homografia

Un cop ja disposem dels punts característics de les dues imatges procedim a calcular l'homografia amb OpenCV passant-li tant els punts característics de la imatge original com la de la capturada. Ja amb aquesta homografia podem calcular la imatge capturada tot seguit d'haver-li aplicat l'homografia. A la Figura 8 podem veure la imatge capturada quan ja ha estat rectificada.

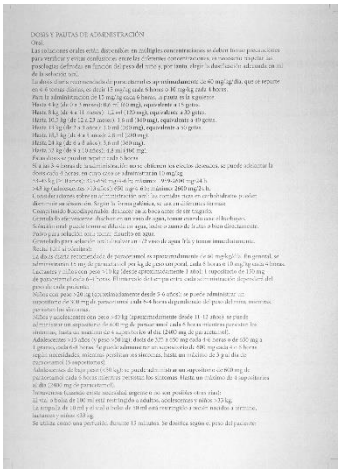


Figura 8 Imatge rectificada amb l'homografia.

5.3 Diferència entre imatges

Ara ja disposem de les dues imatges amb la mateixa mida i rotació, doncs agafem i a una imatge li restem l'altra de manera que puguem veure les diferències entre elles.

Per a obtenir aquest resultat agafem la imatge original i la posem com a capa del color vermell, mentre que a les capes del color verd i el blau hi posem la imatge rectificada; a més hi hem afegit unes taques en Photoshop a la imatge rectificada simulant uns defectes a la impressió, això ho podem veure a la Figura 11.

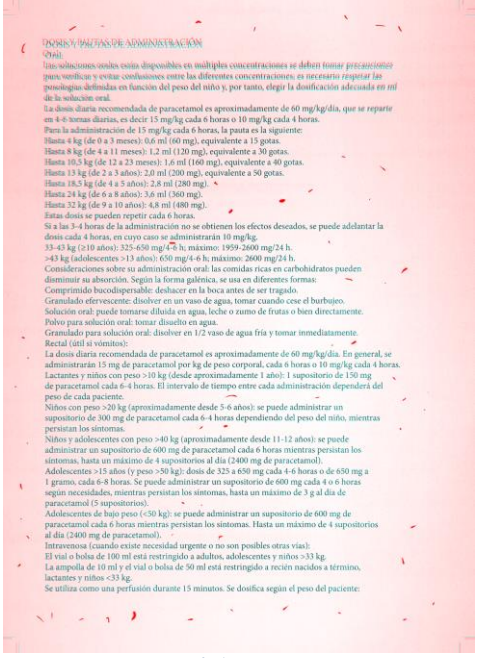


Figura 11 Diferència entre imatges.

6. Problemes trobats i possibles solucions

Al plantejament original del projecte partíem d'un original en PDF del document a imprimir, de manera que aquest document era la nostra referència, convertíem el PDF en imatge de bits i cada captura d'un prospecte imprès (no s'arriba a fer captures de cada prospecte ja que la impressora imprimeix a una velocitat massa elevada per a que es pugui capturar i processar cada prospecte) era processada i comparada amb aquesta referència per a decidir si la impressió era prou bona. Aleshores quan vàrem tenir mostres impreses d'un prospecte vam veure que tindríem un problema, el paper utilitzat per a imprimir no és de prou qualitat (té un gramatge no molt gran) i fàcilment transparenta la impressió de la cara posterior.

Amb aquesta manca d'opacitat en el paper fa que, quan comparem la nostra referència contra la imatge capturada, ens marqui moltes zones com a diferents un respecte l'altre com podem veure a la.

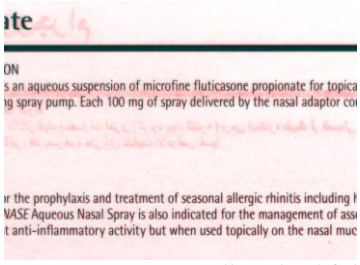


Figura 12. Errors marcats en vermell per les diferències entre el PDF i la captura amb transparència.

Aleshores se'ns va acudir fer les captures amb algun tipus de cartolina o plàstic negre a la part posterior de manera que la part transparentada de la cara posterior fos uniforme (no tindríem text i imatges transparentades sinó un bloc sòlid), però hi havia un altre problema amb aquesta solució, i és que la impressió -que es fa en bobines- va suspesa de manera que, si poséssim una cartolina sota la impressió, hi hauria un espai entre totes dues i no ens serviria.

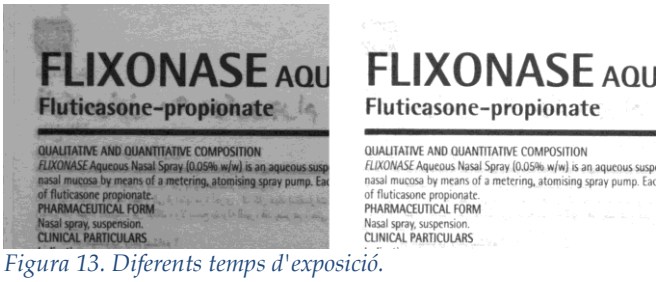


Figura 13. Diferents temps d'exposició.

Una altra solució és la de jugar amb el temps d'exposició, quan aquest temps és més elevat més llum s'està capturant al sensor i això fa que les parts transparentades es perdin, es cremin (Figura 13), tanmateix això fa que també «s'erosionin» els contorns de totes les lletres i il·lustracions, així com part dels defectes d'impressió. Un altre problema que ens planteja l'agafar i fer una exposició més llarga (per a eliminar la transparència) és que el temps d'exposició ens vindria, en gran part, limitat per la velocitat a la que es mou la bobina de paper on s'imprimeix; aquestes impressores treballen a grans velocitats i si l'exposició és prou llarga provocarà moviment a la imatge. La manera de solucionar aquest problema és jugant amb la intensitat del llum, de manera que canviaríem el control del temps d'exposició per la intensitat del llum que il·lumina el paper imprès.

Una alternativa que va sorgir (per part del tutor) és la d'agafar i començar a imprimir, llavors en un moment l'operari que està amb la màquina fa una inspecció visual de les impressions i n'escolleix una que considera bona (absent d'errors), doncs ara aquesta impressió serà la nostra referència (en comptes del PDF original). Mentre que aquesta solució sembla la més adient per a la dificultat que ens planteja la transparència, tampoc està exempta de problemes, i un possible contratemps seria un error de registre en la impressió. Aquesta solució es basa en que totes les impressions seran correctes respecte al registre entre una cara impresa i l'altra, però si passes que en alguna pàgina totes dues cares no estiguessin impreses al mateix lloc, la taca que transparenta cauria en una posició diferent a la impressió que hem considerat com a referència i ens marcaria com a possible error d'impressió.

A la Figura 14 podem observar com la transparència de la impressió posterior entre dos prospectes pot arribar a ser força diferent, quan això no hauria de passar, hi ha un defecte de registre a la impressió.

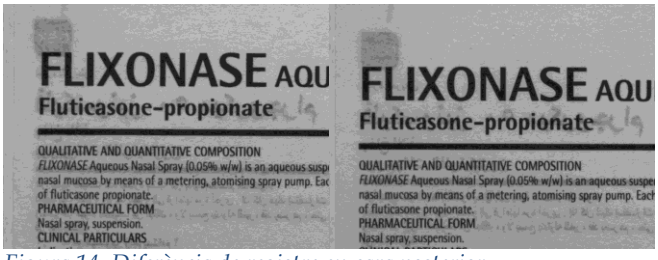


Figura 14. Diferència de registre en cara posterior.

Per a la captació d'imatges al muntatge final es pretén fer servir un encoder (codificador rotatiu o generador d'impulsos), que és un aparell electroòptic (també hi ha electromecànics) amb una roda que va girant sobre un eix, aquesta roda té una sèrie de forats per on passa el llum d'un díode que hi ha a un costat i que és rebut per un fotodíode just a l'altre costat, d'aquesta manera es pot saber quin és l'angle d'aquesta roda que va girant, que en el nostre cas seria la bobina d'impressió. Llavors aquest encoder va generant polsos que serviran per a que amb cada pols es faci una captura d'una impressió i sempre al mateix moment, capturant sempre la mateixa part impresa.

Un exemple ben senzill d'encoder que tenim la majoria de gent als nostres ordinadors és la roda d'un ratolí (Figura 15).

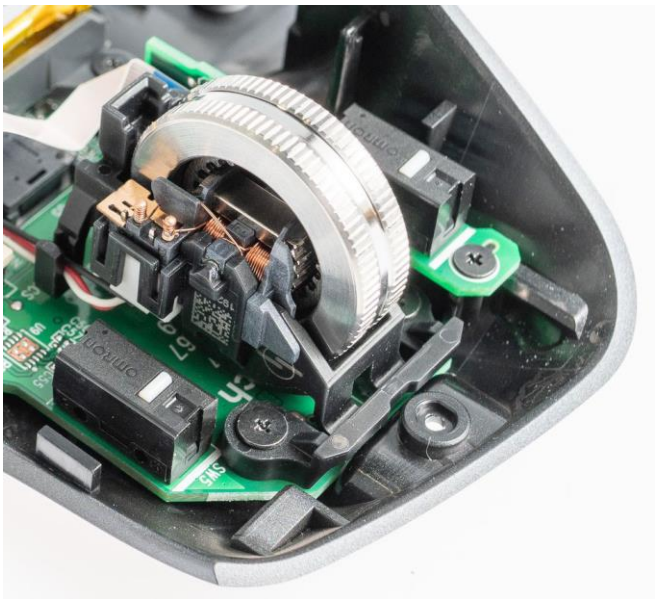


Figura 15. Encoder en la roda d'un ratolí d'ordinador.

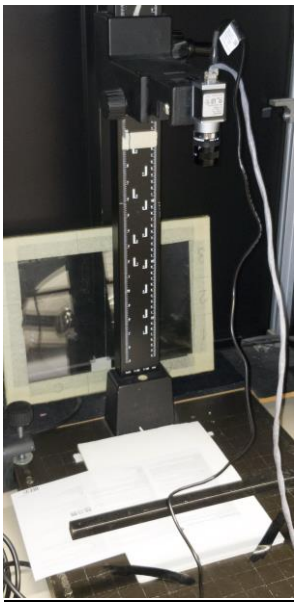


Figura 16 Preparació data set al CVC

6.1 Creació del data set

Per a poder realitzar la part pràctica del TFG necessitava un conjunt de mostres impreses amb les impressores on s'ha de realitzar el muntatge final (el del Centre de Visió per Computador, CVC). L'ideal hagués estat tenir mostres tant amb errors com sense, però com això no va ser possible (ens mancaven mostres amb errors ja que aquests no són molt comuns) i només teníem unes bobines impreses amb els prospectes de medicaments, vàrem haver de simular nosaltres els errors.

Aleshores per a preparar el data set hi havia un muntatge al CVC amb una càmera industrial (de 5 mega píxels i blanc i negre) situada de manera fixa i perpendicular a una base on ficava els prospectes impresos (veure Figura 16). Amb aquest muntatge i la lent que tenia la càmera no donava per a que entrés tot un prospecte (a més, 5 Mpx potser no era prou qualitat per a un prospecte sencer), de manera que per a cada prospecte es necessitaven 6 captures.

En total vaig fer captures de 18 prospectes diferents (13 amb defectes i 5 amb), que van resultar 648 captures, de les quals 468 són amb defectes i 180 sense; per a cada prospecte necessitàvem 6 captures, i per a tenir més mostres, de cada part de prospecte la vaig fotografiar amb 6 temps d'exposició diferents.

Com a defectes que vam introduir (amb tinta negre) van ser principalment taques de tinta, ja fos en una zona sense impressió o sobre alguna part impresa, les taques podien ser taques sòlides o arrossegades (smear ink stain). Una mostra d'aquestes taques la podem veure a la Figura 17.

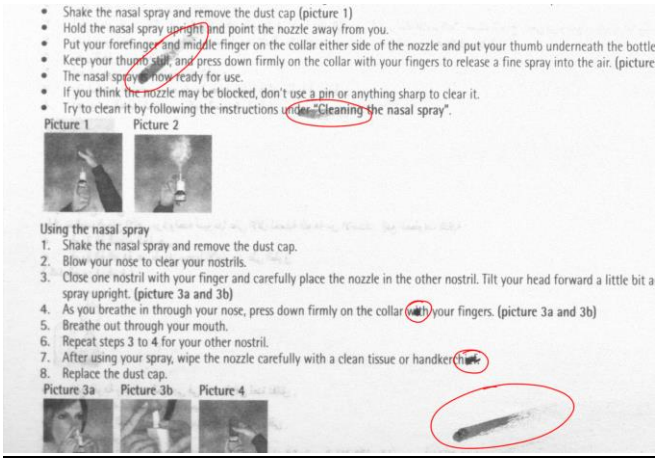


Figura 17 Mostra de defectes artificials

6.2 Binarització i thresholding

Ara que ja tenim les imatges per a treballar i la manera de posar-les en correspondència, necessitem simplificar les imatges de manera que podem comparar-les entre elles.

Si tenim un parell d'imatges per a comparar serà més senzill si hem de comparar dos píxels quan poden tenir només el valor 0 o 1 (imatges binàries) que si poden tenir un valor entre 0 i 255; per això agafem i convertim les captures en nivells de grisos a imatges binàries.

Una altra raó i molt important de fer aquesta binarització és intentar eliminar al màxim la taca impresa de la cara posterior de la impressió, sobretot quan vaig comprovar que aquesta impressió no acabava sempre al mateix lloc (veure Figura 14).

Per a fer això he fet servir el que es coneix com a thresholding, dels diferents tipus de thresholding que ens ofereix OpenCV vaig provar-los tots, des dels simples fins als adaptatius, passant per OTSU (que intenta trobar el millor valor de trheshold de manera automàtica), i els millors resultats els vaig obtenir amb la binarització simple. Aquesta binarització consisteix en agafar una imatge en grisos convertir-la en binària de la següent manera, es defineix un valor de llindar (threshold), aleshores, es llegeix cada píxel

de la imatge i si el valor del píxel no arriba a aquest llindar se li assigna com a valor un 0, si hi arriba se li assigna el valor màxim (255 en aquest cas).

Això sí, com les captures dels prospectes les vaig realitzar amb 6 temps d'exposició diferents, la diferència de lluminositat entre les diferents velocitats de captura era massa gran per a que un sol valor de llindar servís per a totes les captures, de manera que vaig escollir un llindar diferent per a cada un dels 6 temps d'exposició (podem veure aquesta diferència de lluminositat a Figura 13).

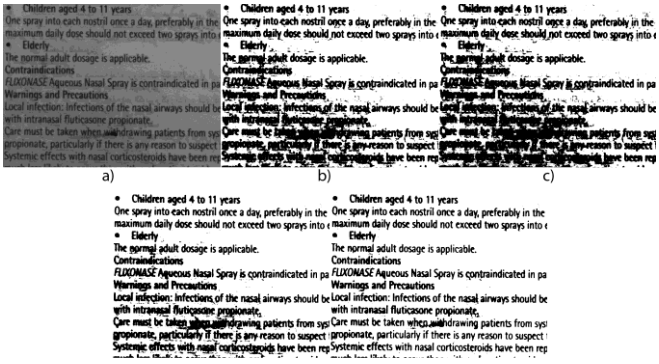


Figura 18 Diferents tipus de binarització junt amb l'original; a) captura original; b) binarització simple; c) binarització OTSU; d) binarització triangle; e) binarització OTSU per sectors.

A la Figura 18 tenim una mostra d'una àrea d'un prospecte que resultava complicada de binaritzar degut a la lluminositat i el poc contrast, aleshores vaig provar amb diferents tipus de binarizacions i cap d'ells oferia resultats satisfactoris; aleshores se'm va acudir fer una prova, aquesta era agafar una captura i anar binaritzant-la per sectors, agafar la captura i dividir-la en n parts i anar binaritzant cada part (amb el tipus OTSU) per a després posar totes les parts juntes.

Mentre que podem veure que aquesta idea de binaritzar per parts va donar bons resultats (com es pot veure a la Figura 18), sobretot en àrees complicades, en altres llocs on hi havia grans àrees blanques generava molt de «soroll» (deixava molts puntets negres) i vaig desistir de la idea ja que necessitava un mètode més genèric.

6.2 Blob detector i find contours

Ara amb les imatges binaritzades «només» havíem de restar una imatge a l'altra i de la imatge resultant agafar i fer servir el Simple Blob Detector[11] de OpenCV.
Comencem dient que un Blob és un conjunt de píxels connexos (es toquen entre ells) i que comparteixen alguna característica comuna com el color o la brillantor.

Tot seguit el Simple Blob Detector fa el següent:

1- Converteix la imatge en binària aplicant diferents llindars, entre el llindar mínim i el màxim aplicant una distància entre llindar i llindar.
2- De cada binarització n'extreu els grups de píxels connexos fent servir findContours i calcula el centre de cada conjunt.
3- A partir de les seves coordinades agrupa els centres de les diferents binaritzacions del pas 1. Aleshores els centres que es troben propers formen un grup anomenat Blob.
4- A partir d'aquests Blobs en calcula el seu centre i el seu radi, i acaba retornant les posicions i les mides dels Blobs.

Però resulta que no podem aplicar directament aquesta funció per a detectar els blobs, ja que després de realitzar la resta entre les dues imatges binaritzades ens queda una imatge amb molts contorns (com es pot observar a la Figura 19).

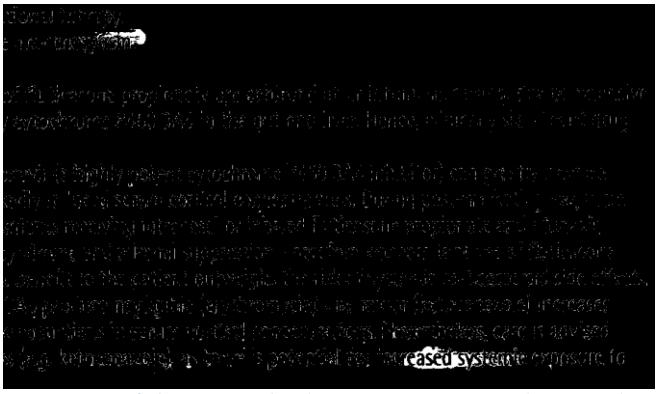


Figura 19 Diferència entre les dues imatges un cop ja binaritzades

Això és degut a que tot i que tenim captures que pertanyen a una mateixa zona d'un prospecte, s'han pres amb el mateix temps d'exposició i se les ha binaritzat amb el mateix llindar, no són perfectament iguals i per poca diferència que hi hagi entre elles, en el gruix de les lletres, quan fem la resta de les dues captures queda aquest perfil.

Doncs ara és aquest perfil que ens «molesta» per a trobar els blobs que l'hem d'eliminar; i per a esborrar-lo he fet servir la erosió de OpenCV. L'erosió serveix per a erosionar o «menjar-se» part dels límits dels objectes que no pertanyen al fons, així, un cop aplicada l'erosió ens queda una imatge on només queden els defectes d'impressió (tot i que aquests processos de binarització i erosió també eliminen algun dels defectes més petits).

Un cop ja hem erosionat la imatge fem servir el simple blob detector, i tot seguit fem servir el findcontours, que agafarà i trobarà els contorns dels blobs, a més de dir-nos quants contorns ha trobat.

6.3 Proves amb el dataset

Ara ja només queda executar el software que he programat per a comprovar com de bé o malament ho fa amb el conjunt de dades que vaig realitzar al CVC.

Com he comentat a l'apartat 6.1, el data set està compost de 648 imatges, 468 amb defectes i 180 sense.

Els procés sencer, després d'executar-ho 5 vegades m'ha donat una mitja de 2233.6 segons, el que em dóna un temps de processat per captura de 3.45 segons.

Procés sencer amb les 648 captures					
	1	2	3	4	5
segons	2230	2341	2434	2145	2018
mitja	2233.6				

Com he comentat anteriorment, les captures les vaig realitzar amb 6 temps d'exposició diferents, de manera que per a cada captura vaig fer servir un valor de llindar diferent (segons el temps d'exposició); els llindars (threshold) que vaig fer servir per a les proves són els següents, la fila **t. exp.** són els valors de temps d'exposició entre 1000 (és un valor del programa de captura de la càmera industrial i no deia quin tipus de valor era).

Llindar utilitzat segons temps d'exposició				
t. exp.	100	115	130	145-200
1a prova	100	120	140	150
2a prova	150	150	150	150

Tot seguit hi ha la matriu de confusió per a la primera prova (amb els llindars de la taula de llindars).

Procés amb les imatges a la seva mida original.		**Valors reals**	
		Defecte	No defecte
Valors predits	Defecte	460	7
	No defecte	8	173
En percentatges		Defecte	No defecte
Valors predits	Defecte	70.98%	1.08%
	No defecte	1.23%	26.69%

Amb els valors que he obtingut he calculat els paràmetres accuracy, precision i recall, que mostro a continuació.

$Accuracy = \frac{True\ Positive + True\ Negative}{Total} = \frac{460+173}{648} = 0.977 \approx$ 97.7%.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} = \frac{460}{460+7} = 0.9850 \approx 98.50\%$$

$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} = \frac{460}{460+8} = 0.983 \approx$ 98.3%.

Per a la segona prova amb un llindar més baix tenim els següents valors que presento a la matriu de confusió.

Procés amb les imatges a la seva mida original.		**Valors reals**	
		Defecte	No defecte
Valors predits	Defecte	458	29
	No defecte	4	157
En percentatges		Defecte	No defecte
Valors predits	Defecte	70.67%	4.47%
	No defecte	0.61%	24.23%

Amb els valors de la segona prova presento els valors-paràmetres accuracy, precision i recall, que mostro a continuació.

$Accuracy = \frac{True\ Positive + True\ Negative}{Total} = \frac{458+157}{648} = 0.949 \approx$ 94.9%.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} = \frac{458}{458+29} = 0.940 \approx 94.0\%$$

$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} = \frac{458}{458+4} = 0.991 \approx$ 99.1%.

7 Conclusió

L'objectiu principal d'aquest treball de fi de grau era el comprovar si es podia realitzar un sistema automatitzat del control de la qualitat d'impressió a una impremta on els controls eren esporàdics (agafant els operaris de les màquines alguna mostra de les bobines ja impreses per a comprovar si hi havia defectes), fent que aquests controls fossin més senzills i «sobre la marxa».

Tot i que no sigui imprescindible, idealment hauríem de poder analitzar cada prospecte imprès, això sí, controlant una mostra cada 3 o 4 segons ja és molt més que no els controls que fan ara mateix.

Les proves finals les he realitzat canviant de tenir un llindar específic per a cada temps d'exposició en la 1a prova a tenir-ne un de fix per a totes les imatges. El fer servir un llindar per a totes les captures ens servirà per si en qualsevol moment es canvia la impressió, per a que el programa continuï donant bons resultats, però d'altra banda, fent servir un únic llindar també ha fet que empitjorés l'exactitud (accuracy) i la precisió (precision).

Amb aquest treball hem pogut demostrar que amb l'ús d'una càmera industrial, un PC, Python i OpenCV, és ben factible que el control de qualitat de les impressions el porti una màquina o que aquesta ajudi a un operari que sigui amb les indicacions del programari qui decideixi si parar la impressió.

8 Treball futur

Degut a la velocitat d'impressió i a la mida de les imatges que hem capturat he pogut veure que no és possible controlar totes i cadascuna de les impressions, de fet necessitem més de 3 segons per a processar una captura, quan segurament la velocitat d'impressió sigui de més de 5 impressions per segon; aleshores una idea que tinc que podria servir per a millorar el programari i que es poguessin controlar més impressions seria fer el programari en C++ de manera que part del procés es pogués paral·lelitzar i així processar més captures de la càmera. Per a la paral·lelització ho podríem fer ja fos amb un processador multinucli o millor encara amb una targeta gràfica.

Una altra feina que es podria fer seria realitzar el mateix treball, que he fet amb la visió per computador clàssica, però amb la intel·ligència artificial.

AGRAÏMENTS

Ja que se'm dona l'oportunitat de fer-ho, m'agradaria finalitzar aquest informe donant les gràcies de tot cor a certes persones que han fet que aquest TFG es dugués a terme. Primer a la direcció de l'Escola d'Enginyeria i la rectoria de la UAB, que em van donar l'oportunitat de reincorporar-me i acabar els estudis passats uns anys.

Seguidament al meu tutor Coen Antens, a qui agraeixo molt especialment tot el seu temps i la seva dedicació en guiar-me per quins passos havia d'anar seguint i els consells que m'ha anat donant.

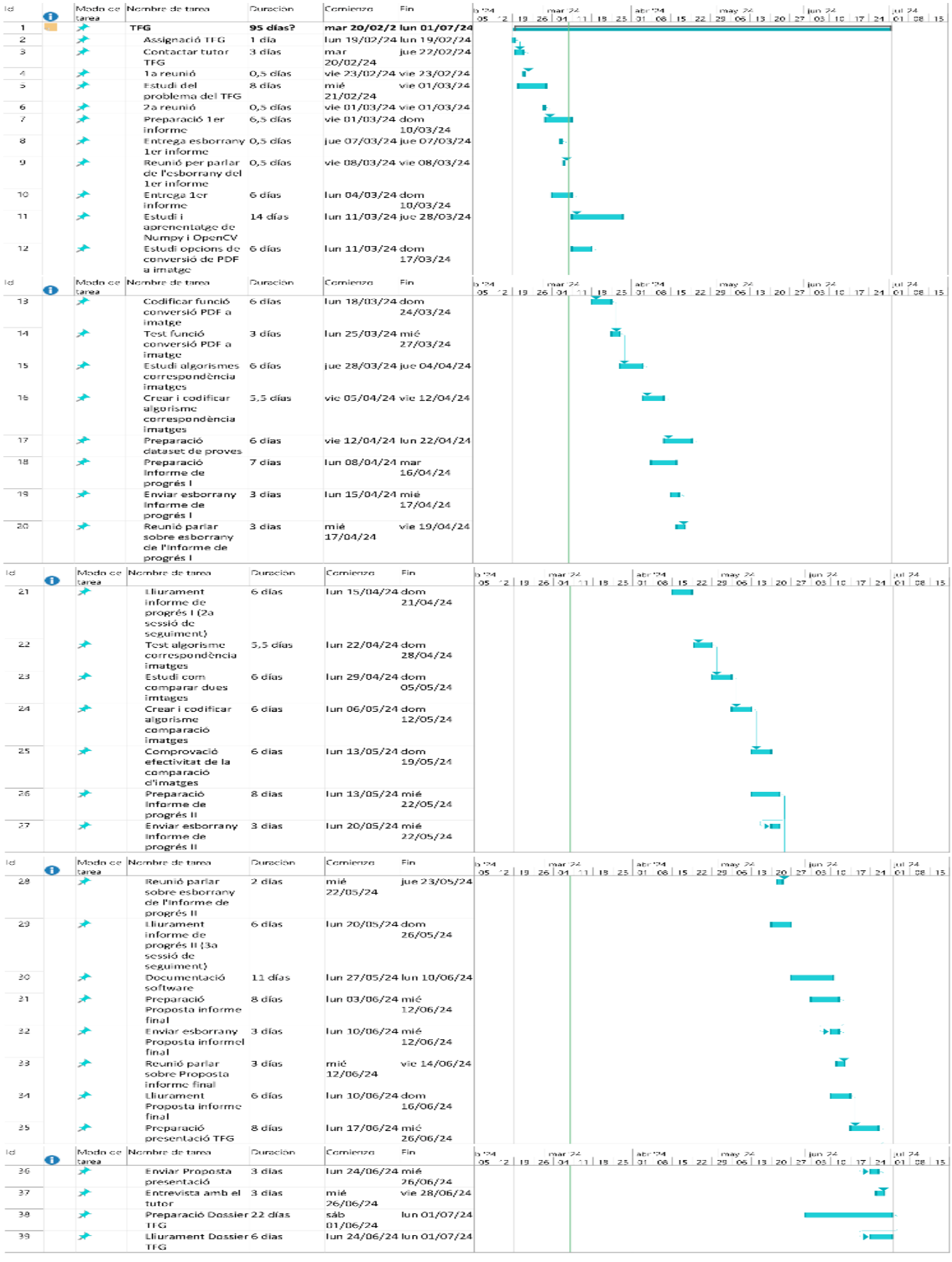
Després al meu estimat germà David, qui m'ha estat donant suport moral durant tots els anys de la carrera universitària (i més).

I finalment als meus pares Josep i Montserrat (els quals ens van deixar fa 2 i 3 anys respectivament), els quals em van animar, recolzar i aconsellar durant tota la meva vida. Aquest TFG va dedicat a ells especialment.

BIBLIOGRAFIA

[1] **José Martínez de Sousa.** Manual de edición y autoedición (2ª edición). Ediciones Pirámide, 2005.
[2] Numpy tutorial: https://numpy.org/doc/stable/user/quickstart.html
[3] OpenCV tutorial: https://docs.opencv.org/4.9.0/d2/d96/tutorial_py_table_of_contents_imgproc.html
[4] Homography: https://www.youtube.com/watch?v=l_qjO4cM74o
[5] Projecció de perspectiva: https://www.youtube.com/watch?v=2BIzmFD_pRQ
[6] SIFT: https://www.youtube.com/watch?v=IBcsS8_gPzE&list=PLlCkKK04bmVlvCs-S-2DnGf08MY2Hdd0n&index=5
[7] **Richard Szeliski.** Computer Vision Algorithms and Applications (2nd Edition). Springer, 2022. (Chapter 7: Feature detection and matching).
[8] **David Lowe**. Object recognition from Local Scale-Invariant Features. https://www.cs.ubc.ca/~lowe/papers/iccv99.pdf
[9] SIFT wikipedia: https://en.wikipedia.org/wiki/Scale-invariant_feature_transform
[10] SIFT method: https://patents.google.com/patent/US6711293
[11] Simple Blob Detector: https://docs.opencv.org/3.4/d0/d7a/classcv_1_1SimpleBlobDetector.html

ANNEX



Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin
1		TFG	95 días?	mar 20/02/24	lun 01/07/24
2		Assignació TFG	1 día	lun 19/02/24	lun 19/02/24
3		Contactar tutor TFG	3 días	mar 20/02/24	jue 22/02/24
4		1a reunió	0,5 días	vie 23/02/24	vie 23/02/24
5		Estudi del problema del TFG	8 días	mié 21/02/24	vie 01/03/24
6		2a reunió	0,5 días	vie 01/03/24	vie 01/03/24
7		Preparació 1er informe	6,5 días	vie 01/03/24	dom 10/03/24
8		Entrega esborrany 1er informe	0,5 días	jue 07/03/24	jue 07/03/24
9		Reunió per parlar de l'esborrany del 1er informe	0,5 días	vie 08/03/24	vie 08/03/24
10		Entrega 1er informe	6 días	lun 04/03/24	dom 10/03/24
11		Estudi i aprenentatge de Numpy i OpenCV	14 días	lun 11/03/24	jue 28/03/24
12		Estudi opcions de conversió de PDF a imatge	6 días	lun 11/03/24	dom 17/03/24
13		Codificar funció conversió PDF a imatge	6 días	lun 18/03/24	dom 24/03/24
14		Test funció conversió PDF a imatge	3 días	lun 25/03/24	mié 27/03/24
15		Estudi algorismes correspondència imatges	6 días	jue 28/03/24	jue 04/04/24
16		Crear i codificar algorisme correspondència imatges	5,5 días	vie 05/04/24	vie 12/04/24
17		Preparació dataset de proves	6 días	vie 12/04/24	lun 22/04/24
18		Preparació Informe de progrés I	7 días	lun 08/04/24	mar 16/04/24
19		Enviar esborrany Informe de progrés I	3 días	lun 15/04/24	mié 17/04/24
20		Reunió parlar sobre esborrany de l'informe de progrés I	3 días	mié 17/04/24	vie 19/04/24
21		Lliurament informe de progrés I (2a sessió de seguiment)	6 días	lun 15/04/24	dom 21/04/24
22		Test algorisme correspondència imatges	5,5 días	lun 22/04/24	dom 28/04/24
23		Estudi com comparar dues imtages	6 días	lun 29/04/24	dom 05/05/24
24		Crear i codificar algorisme comparació imatges	6 días	lun 06/05/24	dom 12/05/24
25		Comprovació efectivitat de la comparació d'imatges	6 días	lun 13/05/24	dom 19/05/24
26		Preparació Informe de progrés II	8 días	lun 13/05/24	mié 22/05/24
27		Enviar esborrany Informe de progrés II	3 días	lun 20/05/24	mié 22/05/24
28		Reunió parlar sobre esborrany de l'informe de progrés II	2 días	mié 22/05/24	jue 23/05/24
29		Lliurament informe de progrés II (3a sessió de seguiment)	6 días	lun 20/05/24	dom 26/05/24
30		Documentació software	11 días	lun 27/05/24	lun 10/06/24
31		Preparació Proposta informe final	8 días	lun 03/06/24	mié 12/06/24
32		Enviar esborrany Proposta informel final	3 días	lun 10/06/24	mié 12/06/24
33		Reunió parlar sobre Proposta informe final	3 días	mié 12/06/24	vie 14/06/24
34		Lliurament Proposta informe final	6 días	lun 10/06/24	dom 16/06/24
35		Preparació presentació TFG	8 días	lun 17/06/24	mié 26/06/24
36		Enviar Proposta presentació	3 días	lun 24/06/24	mié 26/06/24
37		Entrevista amb el tutor	3 días	mié 26/06/24	vie 28/06/24
38		Preparació Dossier TFG	22 días	sáb 01/06/24	lun 01/07/24
39		Lliurament Dossier TFG	6 días	lun 24/06/24	lun 01/07/24