Segmentació d'imatges de raigs X als punts de control dels aeroports

Alumne: Alejandro Brines Gimeno

Consultor: Samir Kanaan Izquierdo

1.- Problemàtica

La inspecció d'objectes als punts de control dels aeroports la sol realitzar un operari de forma manual visualitzant la imatge generada. Amb aquest sistema no es pot detectar al 100% tots els objectes que es troben a dins d'una bossa, be per que els objectes estan superposats uns amb altres, be per errors dels operaris.

Es desitja desenvolupar un sistema que separi els objectes per a facilitar visualització i rebaixar el temps d'espera dels passatgers.

2.- Que es la segmentació?

Es un procés per el qual es dividix la imatge en les parts u objectes que la formen. Es el primer pas en qualsevol procés d'anàlisis d'imatge.

Els algorismes de segmentació es basen en dos propietats bàsiques del nivell de gris: discontinuïtat i similitud.

2.1.- Segmentació basat per discontinuïtat

Es dividix la imatge basant-se en canvis bruscs del nivell de gris.

El mètode mes comú de buscar discontinuïtats es la correlació d'una imatge amb una mascara.

$$R=\sum_{i=1}^{9} w_i z_i$$

La resposta de la mascara ve referida a la seua posició central.

Hi ha tres tipus de discontinuïtats: punts, línies i bordes.

2.1.1.- Detecció de punts

Un punt aïllat de una imatge te un to de gris que difereix significativament dels tons de gris dels seus píxels veïns.

Aquesta es una mascara per detectar un punt aïllat.

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Direm que un píxel es un punt aïllat si el resultat d'aplicar la mascara sobre el píxel (en valor absolut) es major o igual que un cert valor llindar T.

2.1.2.- Detecció de línies

Una línia es una seqüencia de píxels en la que dos píxels consecutius estan connectats. En un entorn de 3x3 cada píxel es pot connectar amb algú dels seus 8 píxels veïns, i per tant, anem a tindre sols 4 direccions possibles.

Aquestes son les mascares.

$$\begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$$

Horitzontal Vertical 45º -45º

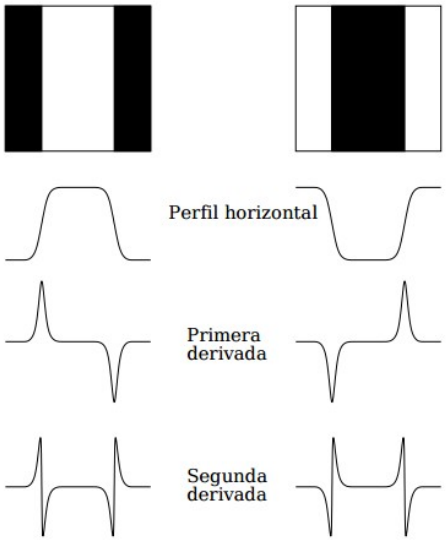
Al aplicar aquestes mascares sobre la imatge, la mascara que traga el valor mes alt voldrà dir que la línia segueix la direcció.

2.1.3.- Detecció de bordes (1)

Un borde es defineix com la frontera entre dos regions de nivell de gris relativament diferent.

La idea bàsica darrere de qualsevol detector de bordes es el calcul de un operador local de derivació.

El valor de la magnitud de la primera derivada ens serveis per a detectar la presencia de bordes, mentre que el signe de la segon derivada en indica si el píxel pertany a la zona clara o a la zona fosca. Ames la segon derivada presenta sempre un encreuament per cero en el punt mig de la transició. Aixo ens pot ser molt útil per a localitzar bordes en una imatge.



2.1.3.- Detecció de bordes (2)

Operadors per determinar les derivades parcials.

Els operadors de Sobel i de Frei-Chen tenen l'avantatge de proporcionar un suavitzat a mes del efecte de derivació. Ja que la derivació accentua el soroll, el efecte de suavitzat es particularment interessant, ja que elimina part del soroll. El requisit bàsic de un operador de derivació es que la suma dels coeficients de la mascara siga nul·la, per que la derivada de una zona uniforme de la imatge siga cero.

$$\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Roberts

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Prewitt

$$\begin{bmatrix} -1 & 0 & -1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Sobel

$$\begin{bmatrix} -1 & 0 & 1 \\ -\sqrt{2} & 0 & \sqrt{2} \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & -\sqrt{2} & -1 \\ 0 & 0 & 0 \\ 1 & \sqrt{2} & 1 \end{bmatrix}$$

Frei-Chen

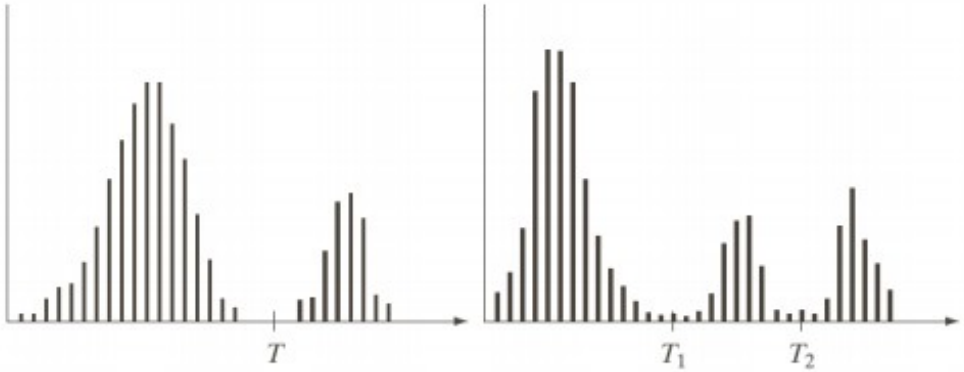
2.2.- Segmentació basat per similitud

Es dividix la imatge basant-se en la cerca de zones que tinguen valors similars en el nivell de gris.

Hi ha tres tècniques bàsiques en la detecció de similitud, aquestes son: tècniques llindars, creixement de regions i divisió i fusió de regions.

2.2.1.- Tècniques llindars

Es realitza una binarització per a diferenciar un objecte del fons de la imatge. A través del histograma obtindrem una gràfica on es mostra el numero de píxels per cada nivell de gris que apareix en la imatge. Per a binaritzar la imatge, es deu elegir un valor adequat (llindar) dins dels nivells de grisos, de tal forma que el histograma forme una vall amb eixe nivell. Tots els nivells de grisos menors al llindar es convertirà en negre i tots els majors en blanc.



2.2.2.- Creixement de regions

Es un procediment mitjançant el qual s'agrupen píxels o subregions en regions majors. Comença a partir de un conjunt de píxels llavor, de forma que a partir de cada llavor van creixen regions afegint píxels a la llavor de entre aquells píxels veïns que tenen propietats similars. El resultat de la segmentació dona lloc com a molt a tantes regions com llavors hi hagin.

Un exemple de similitud es la diferencia el nivell de gris absolut. Es calcula la diferencia en el valor absolut del nivell amb el de la llavor i si no supera un valor llindar s'afegix a la regió.

Hi ha dos problemes fonamentals, la elecció de les llavors i la elecció de les propietats a examinar per afegir els píxels.

2.2.3.- Divisió i fusió de regions

Bàsicament la tècnica es subdividir inicialment la imatge en un conjunt arbitrari de regions disjuntes i posteriorment fusionar o dividir.

Siga *R* la regió corresponent a la imatge completa i siga *P* el predicat de similitud elegit. Un procediment per segmentar *R* consisteix en subdividir recursivament cada regió en quadrants mes i mes xicotets fins que complisca para cada regió que P(Ri)=CERT. Es a dir si P(R)=FALS, es dividix la imatge en quatre regions (quadrants). Si per algun del quadrants es FALS, es torna a dividir dit quadrant en quatre subquadrants i així successivament.

3.- Eines de visió per computador

Per a la realització d'aquest treball s'ha utilitzat dos eines de visió per computador. Aquestes eines ha proporcionat mètodes i utilitats ja implementades per facilitar el desenvolupament del treball. Les eines son OpenCV i ITK.

- OpenCV es una eina de propòsit general desenvolupada per Intel. Es multi plataforma, esta implementat en C i C++, conte mes de 500 funcions que abasten una gran gama d'ares en el procés de la visió.
- ITK es una eina especifica per a la segmentació i classificació de imatges mèdiques finançat per National Library of Medicine (U.S.). Està implementat en C++ del estil de un programa genèric, fent el codi molt eficient.

4.- Entorn de desenvolupament

Degut que les eines de visió per computador utilitzades son lliures, s'ha decidit que el entorn de desenvolupament també siga utilitzant programari lliure.

- Com a sistema operatiu s'ha utilitzat GNU/Linux, una distribució DEBIAN 8.2 amb una arquitectura de 64 bits.
- Com IDE s'ha utilitzat Monodevelop versió 5.5 i llenguatge C#. La selecció del llenguatge ha sigut per l'experiència prèvia que hi ha per evitar corbes d'aprenentatge i possibles endarreriments al TFM.
- Per a les eines, s'han utilitzat wrappers existents per a OpenCV (EmguCV) i ITK (SimpleITK).

5.- Algorismes de segmentació

Els algorismes de segmentació estudiats en aquest treball han sigut un total de sis. Quatre d'aquests algorismes estan desenvolupats amb EmguCV (OpenCV) i els restants amb SimpleITK (ITK).

Algorismes implementats amb EmguCV:

- K-means
- Canny
- Otsu
- Threshold

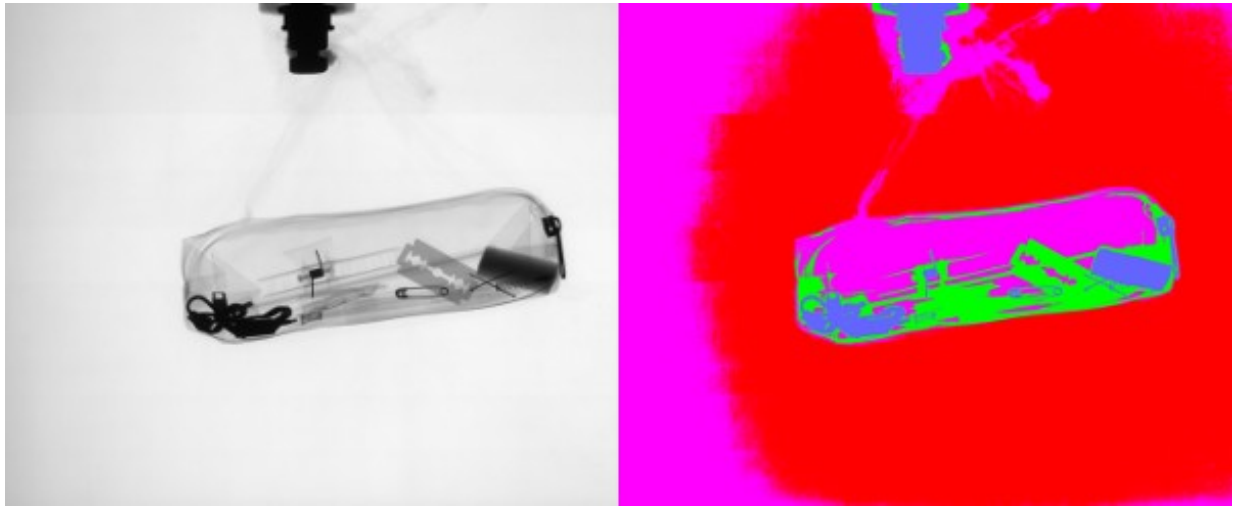
Algorismes implementats amb SimpleITK:

- Otsu
- Threshold

5.1.- Algorisme k-means (EmguCV)

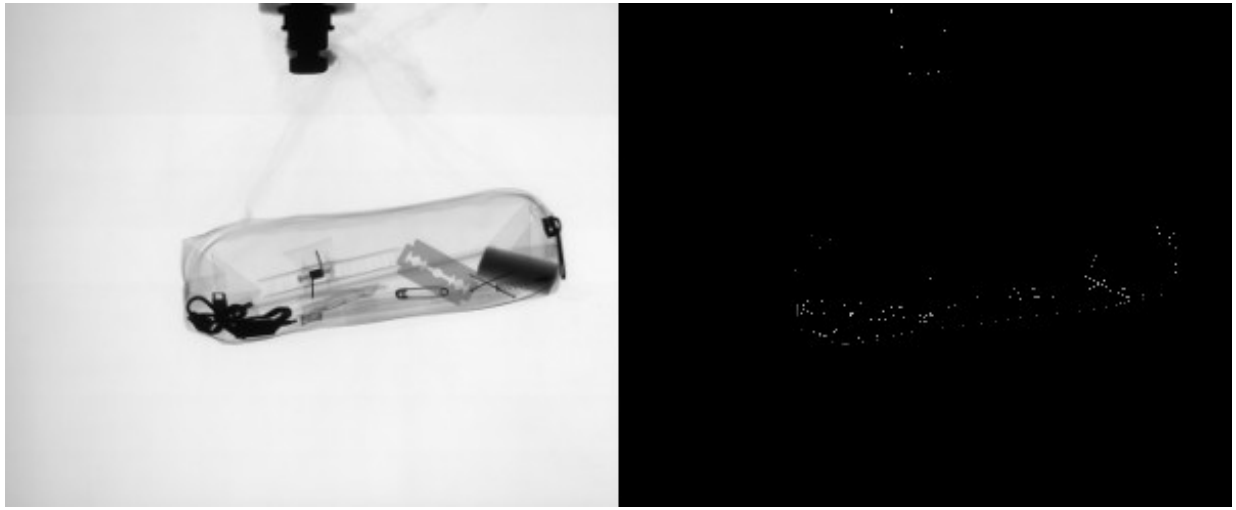
Aquest algorisme es un mètode d'agrupament que te com a objectiu la partició de n observacions en k grups en el que cada observació pertany al grup mes aprop a la mitja.

Al no ser un algorisme de segmentació pròpiament dit, per a la implementació es construïx la imatge a partir de la estructura retornada d'aplicar el algorisme píxel a píxel.



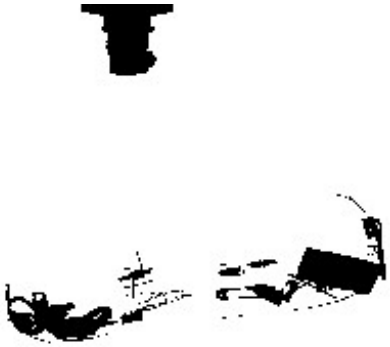
5.2.- Algorisme canny (EmguCV)

Aquest mètode utilitza el algorisme del mateix nom per a reconèixer els bordes. Una part important del algorisme es el de aplicar un filtre gaussià per a reduir el soroll. Se li indica el llindar per trobar els segments inicials, que en el nostre cas serà de 127 i el llindar utilitzat per a la vinculació de bordes, nosaltres utilitzarem 255.



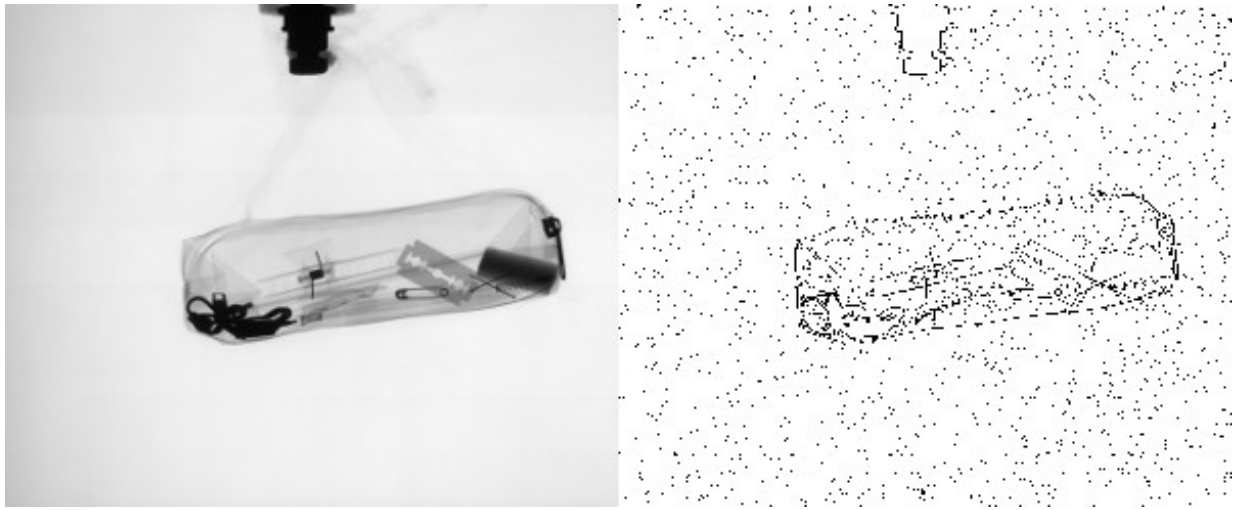
5.3.- Algorisme otsu (EmguCV)

Aquest algorisme realitza una binarització de la imatge. El valor llindar de la binarització es calcula automàticament a partir del histograma de la imatge. S'aplica un filtre gaussià per a reduir el soroll que puga hi haure. El valors llindar per veure si el píxel pertany o no es de 127.



5.4.- Algorisme threshold (EmguCV)

Aquest algorisme realitza una binarització adaptativa, es a dir, el llindar es calcula a partir de petites regions de la imatge. El valor llindar es calcula a partir de la suma ponderada dels valors dels veïns on els pesos son una finestra gaussiana. En el nostre cas si el píxel complix la condició agafa el valor 255 i la mida de la regió es de 11.

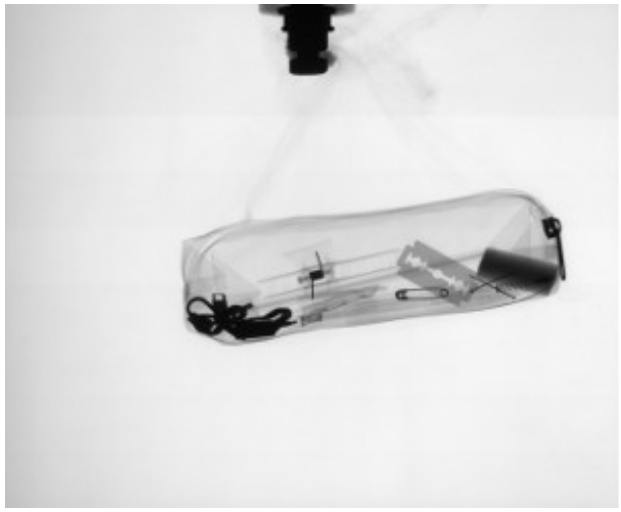


5.5.- Algorisme otsu (SimpleITK)

Aquest realitza una binarització de la imatge calculant automàticament el valor llindar a partir del histograma de la imatge. Com en la versió de *EmguCV* apliquem un filtre gaussià per a reduir el soroll que puga hi haure. El valors llindar per veure si el píxel pertany o no es de 127 i el valor que agafarà si pertany es de 255 també com en la versió de *EmguCV*.

5.6.- Algorisme threshold (SimpleITK)

Aquest algorisme realitza una binarització de la imatge. A diferencia del seu homòleg en *EmguCV* aquesta no es adaptativa i per tant dependrà molt del valor que se li done com a mínim i màxim. En el nostre cas i per homogeneïtzar amb la resta dels algorismes, aquest valors son 127 i 255.



6.- Avaluació dels algorismes

Per a la avaluació dels algorismes s'ha tingut dos aspectes, el temps d'execució i la qualitat de la imatge segmentada.

Per a quantificar la qualitat de la imatge, s'ha optat per utilitzar un mètode objectiu, el index Rand ajustat. Hi ha estudis previs sobre l'avaluació d'imatges segmentades indicant que el index Rand ajustat es bon indicatiu sobre la qualitat de la imatge segmentada amb la original.

6.1.- Index Rand ajustat (1)

Es una modificació del index Rand. Aquest index trau el grau de similitud de la imatge segmentada amb la imatge original.

Donat un conjunt S de n elements, i dos agrupacions d'aquests grups, $x=\{x_1, x_2, \dots x_r\}$ i $y=\{y_1, y_2, \dots y_s\}$ superposició entre x i y es pot resumir en una taula de contingència $[n_{ij}]$ en el que cada entrada n_{ij} denota el numero d'objectes $x_i \, i \, y_j : n_{ij} |x_i \cap y_j|$

x/y	y_1	y_2	$\dots$	y_s	$Sums$
x_1	n_{11}	n_{12}	$\dots$	n_{1s}	a_1
x_2	n_{21}	n_{22}	$\dots$	n_{2s}	a_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_r	n_{r1}	n_{r2}	$\dots$	n_{rs}	a_r
$Sums$	b_1	b_2	$\dots$	b_s	

6.1.- Index Rand ajustat (2)

La formula ve donada per

$$Index\ Ajustat = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2}\sum_i \binom{b_j}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2}\sum_i \binom{b_j}{2}\right] / \binom{n}{2}}$$

6.1.- Index Rand ajustat (3)

A efectes pràctics es realitza una comparació píxel a píxel de la següent forma, es compara dos píxels de la imatge original (A) i els dos píxels de la mateixa posició de la imatge segmentada (B). Aquesta comparació pot donar els següents resultats que els agruparem:

- = i = → el parell assignat es igual en A i en B. (a)
- ≠ i ≠ → el parell assignat es diferent en A i en B. (b)
- = i ≠ → el parell assignat es igual en A i diferent en B. (c)
- ≠ i = → el parell assignat es diferent en A i igual en B. (d)

6.1.- Index Rand ajustat (4)

Una volta s'ha tret els diferents grups (a), (b), (c) i (d) s'aplica la formula anterior:

$$Index\ Ajustat = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_{i}\binom{a_i}{2}\sum_{i}\binom{b_j}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_{i}\binom{a_i}{2}+\sum_{j}\binom{b_j}{2}\right]-\left[\sum_{i}\binom{a_i}{2}\sum_{i}\binom{b_j}{2}\right]/\binom{n}{2}}$$

Que traduïda als grups de les comparacions es

$$Index\ Ajustat = \frac{a-((a+c)*(a+d)/(a+b+c+d))}{(((a+c)+(a+d))/2.0)-(((a+c)*(a+d))/(a+b+c+d))}$$

6.1.- Index Rand ajustat (5)

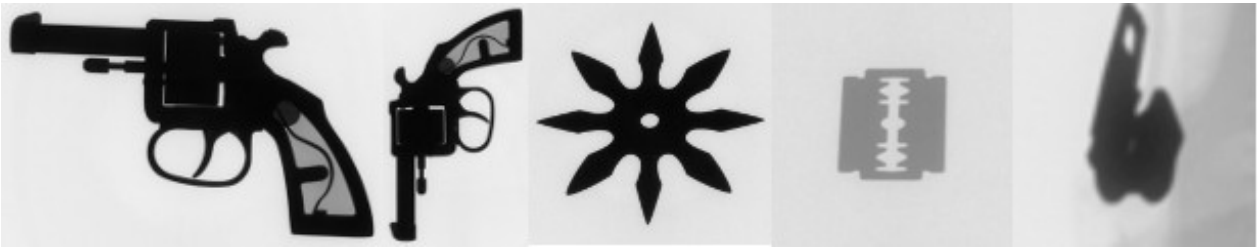
Degut a que comparar tots els píxels entre si d'una imatge es una tasca molt costosa en termes computacionals, s'ha realitzat una aproximació que es detalla a continuació.

- Per cada fila de la imatge sols es compara les columnes.
- La comparació de columnes es realitza en blocs de 100, es a dir es compara tots els primers 100 píxels. Després es compararà els 100 píxels del segon bloc (del 101 fins al 200).

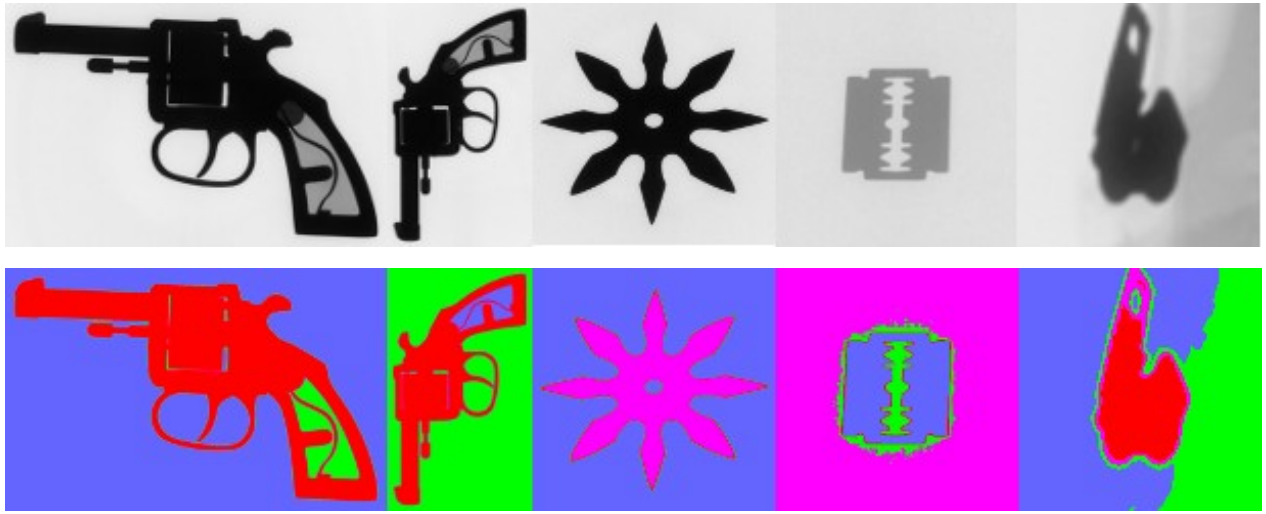
Amb aquesta aproximació es pot veure que el index Rand ajustat calculat i la comparació visual de la imatge segmentada amb la original son afins.

7.- Test Inicial

Abans de realitzar les probes s'ha realitzat un test inicial amb cinc imatges d'un sol objecte i s'ha quantificat el temps d'execució i el index Rand ajustat.

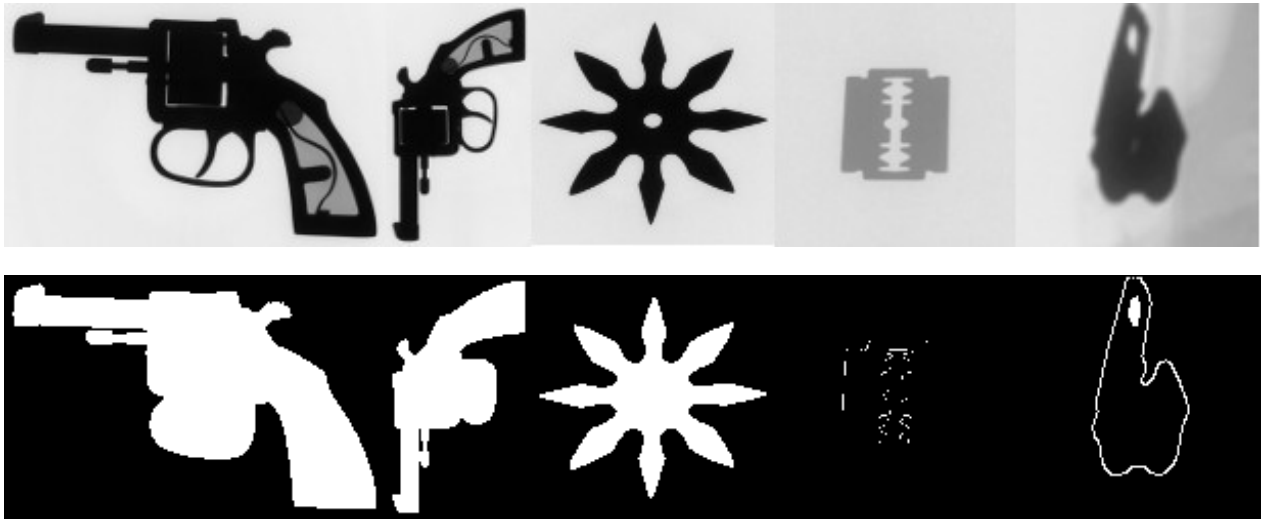


7.1.- Test Inicial k-means (EmguCV)



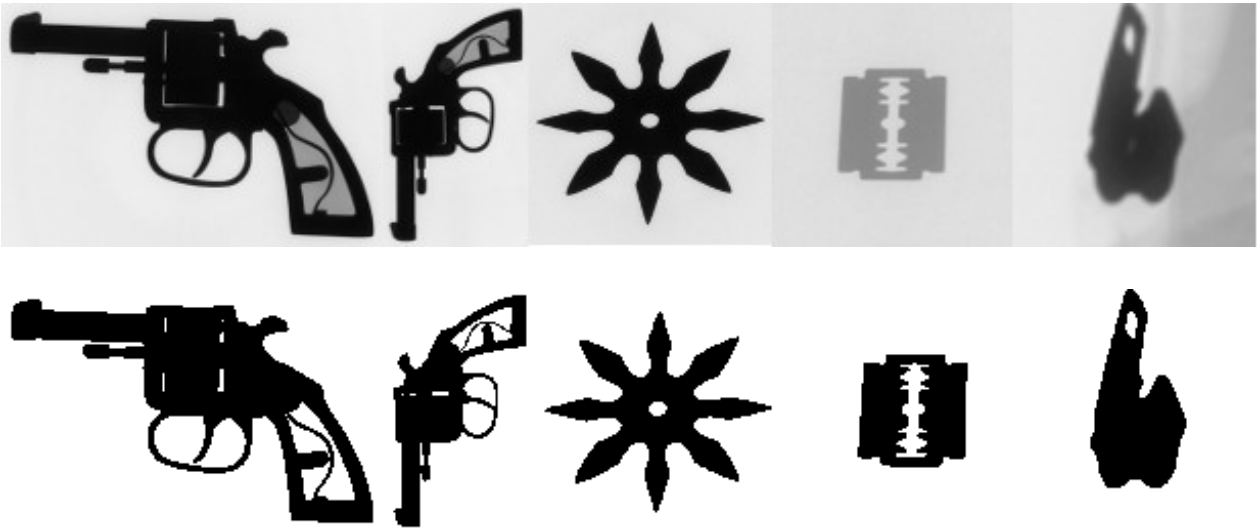
Imatge	Mida	Temps	Index Rand Ajustat
1.- B0049_0088	1285x816	13445	0,0803478857387481
2.- B0049_0113	770x1299	12856	0,0843110573778264
3.- B0050_0008	861x861	10763	0,0534051988767529
4.- B0051_0028	407x407	2537	0,0837441565004505
5.- B0057_0006	119x119	266	0,1100514780937980

7.2.- Test Inicial canny (EmguCV)



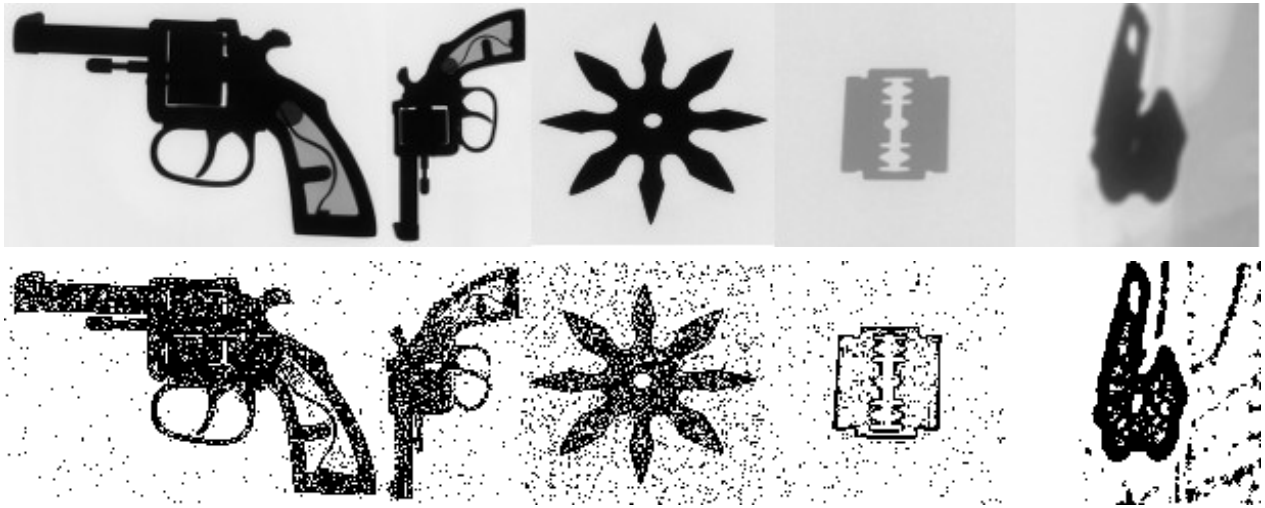
Imatge	Mida	Temps	Index Rand Ajustat
1.- B0049_0088	1285x816	122	0,0421130089527789
2.- B0049_0113	770x1299	53	0,0495830720917237
3.- B0050_0008	861x861	19	0,0421057159932107
4.- B0051_0028	407x407	3	0,0067889043750807
5.- B0057_0006	119x119	0	0,0063796063720293

7.3.- Test Inicial otsu (EmguCV)



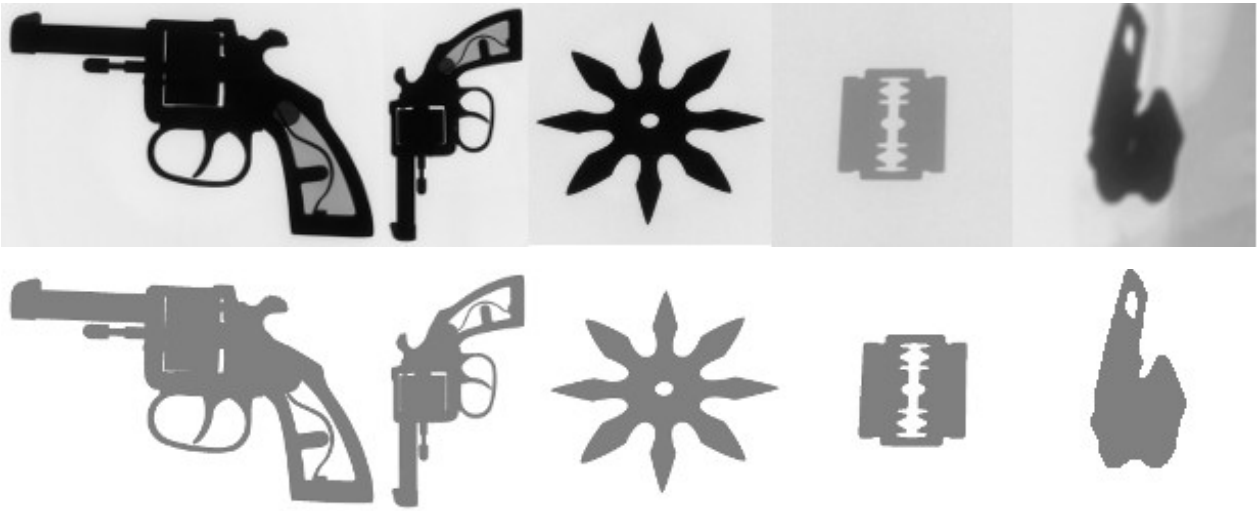
Imatge	Mida	Temps	Index Rand Ajustat
1.- B0049_0088	1285x816	160	0,0654467955967671
2.- B0049_0113	770x1299	13	0,0718352833267570
3.- B0050_0008	861x861	9	0,0431868127602163
4.- B0051_0028	407x407	1	0,0580938505531419
5.- B0057_0006	119x119	2	0,0425339631598744

7.4.- Test Inicial threshold (EmguCV)



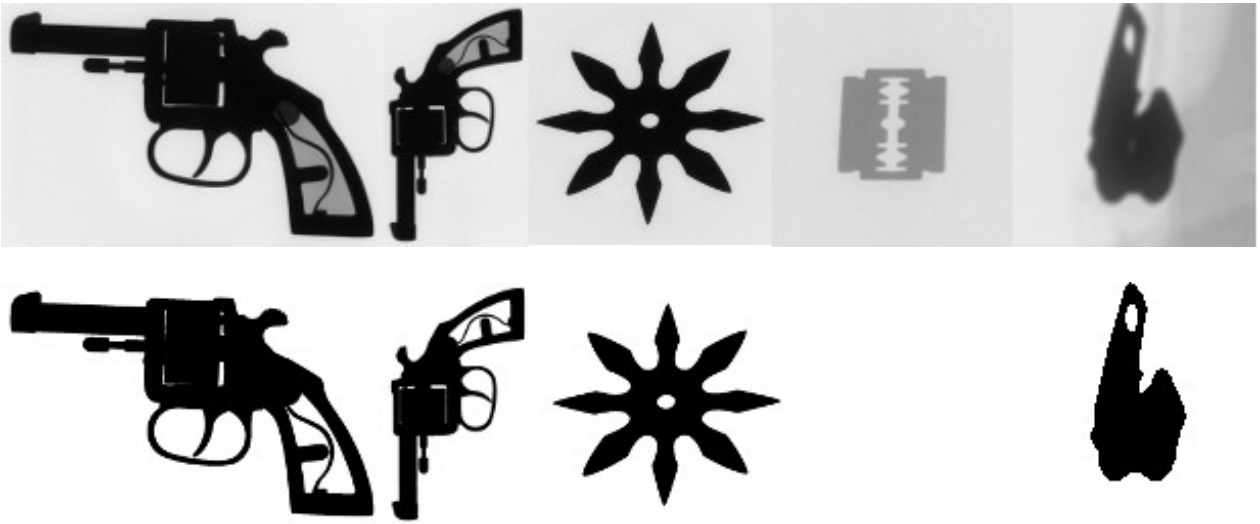
Imatge	Mida	Temps	Index Rand Ajustat
1.- B0049_0088	1285x816	86	0,0846030259902033
2.- B0049_0113	770x1299	12	0,0888268896823302
3.- B0050_0008	861x861	10	0,0760994647502181
4.- B0051_0028	407x407	2	0,0576853653122593
5.- B0057_0006	119x119	0	0,0464857936109476

7.5.- Test Inicial otsu (SimpleITK)



Imatge	Mida	Temps	Index Rand Ajustat
1.- B0049_0088	1285x816	276	0,0654429483542487
2.- B0049_0113	770x1299	36	0,0718539002583513
3.- B0050_0008	861x861	82	0,0431876570465264
4.- B0051_0028	407x407	17	0,0582946579078726
5.- B0057_0006	119x119	12	0,0423912520521632

7.6.- Test Inicial threshold (SimpleITK)



Imatge	Mida	Temps	Index Rand Ajustat
1.- B0049_0088	1285x816	74	0,0654358364585298
2.- B0049_0113	770x1299	11	0,0717936966010959
3.- B0050_0008	861x861	16	0,0432096829440277
4.- B0051_0028	407x407	2	0
5.- B0057_0006	119x119	2	0,0414356571506785

8.- Proba dels algorismes

Per realitzar la proba del algorismes, s'ha executat 150 imatges de diferent mida, 2688x2208, 900x1430, 850x850 i 612x452. Aquestes imatges contenen diferents objectes dins de les bosses. S'ha quantificat el temps d'execució i el index Rand ajustat.

Els resultats s'han normalitzats per la mitjana. S'ha agrupat els resultats per mida de les imatges, per veure la relació amb la imatge.

8.1.- Resultats 150 imatges

Algorisme	Temps	Index Rand Ajustat
k-means - EMGU	12677,0	0,081537102
canny - EMGU	36,0	0,008578288
otsu - EMGU	25,5	0,034990004
threshold - EMGU	21,0	0,102622174
otsu - SITK	87,5	0,034873905
threshold - SITK	36,5	0,036169408

8.2.- Resultats imatges mida 2688x2208

Algorisme	Temps	Index Rand Ajustat
k-means - EMGU	63625	0,088443875
canny - EMGU	155	0,007630716
otsu - EMGU	95	0,044671572
threshold - EMGU	95	0,119007105
otsu - SITK	307	0,044174209
threshold - SITK	70	0,042095562

8.3.- Resultats imatges mida 900x1430

Algorisme	Temps	Index Rand Ajustat
k-means - EMGU	12641,0	0,077906165
canny - EMGU	36,0	0,015048903
otsu - EMGU	19	0,029558597
threshold - EMGU	21,0	0,090360766
otsu - SITK	85,5	0,029402134
threshold - SITK	16,5	0,036606267

8.4.- Resultats imatges mida 850x850

Algorisme	Temps	Index Rand Ajustat
k-means - EMGU	6993	0,095373834
canny - EMGU	17	0,013799502
otsu - EMGU	10	0,044626861
threshold - EMGU	9	0,123092417
otsu - SITK	60	0,044261977
threshold - SITK	8	0,038752168

8.5.- Resultats imatges mida 612x452

Algorisme	Temps	Index Rand Ajustat
k-means - EMGU	2833,5	0,053360971
canny - EMGU	8,0	0,009625877
otsu - EMGU	6,0	0,021235154
threshold - EMGU	4,0	0,057294840
otsu - SITK	16,0	0,021055571
threshold - SITK	4,0	0,023054466

8.6.- Conclusions (1)

Després d'analitzar els resultats dels diferents algorismes es pot afirmar el següent:

- Threshold (EmguCV): millor index Rand de tots i mes ràpid en la execució.
- K-means (EmguCV): el segon millor en quant al index Rand, molt lent en la execució, descartat.
- Canny (EmguCV): pitjor index Rand de tots, descartat.
- Otsu (EmguCV): index Rand mig, velocitat d'execució acceptable, segona opció.
- Threshold (SimpleITK): index Rand paregut a otsu (EmguCV) molt mes lent, descartat.
- Otsu (SimpleITK): com threshold (SimpleITK), descartat.

8.6.- Conclusions (2)

En quant a les llibreries s pot dir que EmguCV està mes optimitzada que SimpleITK comparant els temps dels algorismes threshold i otsu. També s'ha detectat fuga de memòria en la execució de les probes en la llibreria SimpleITK, ha començat la execució amb un ús del 20% i a finalitzat amb un ús de memòria del 90%.