Treball fi de carrera

ENGINYERIA TÈCNICA EN INFORMÀTICA DE SISTEMES

Facultat de Matemàtiques
Universitat de Barcelona

DETECCIÓ I SEGMENTACIÓ DE MODELS DE AERONAUS PER VISIÓ PER COMPUTADOR

Julià Ríos Paniagua

Director: Sergio Escalera Guerrero
Realitzat a: Departament de Matemàtica Aplicada i Anàlisi. UB
Barcelona, 6 de juliol de 2010

Agraïments

Els primers agraïments són pel meu tutor del projecte, Sergio Escalera Guerrero, que cada setmana m'ha anat indicant els passos que havia de seguir per aconseguir els objectius que ens havíem marcat.

En segon lloc, m'agradaria agrair també la paciència i solidaritat de'n Toni Hernández, que m'ha donat molt bons consells i m'ha ajudat a solucionar diversos problemes que han sorgit al llarg del treball, tant a l'hora de programar amb les llibreries de OpenCV en C++, com a l'hora de comprendre i utilitzar l'algoritme de GrabCut.

També a Marc Garcia Ramis que m'ha ajudat a utilitzar les llibreries de blobs i el mètode adaptive.

Moltes gràcies.

Resum

Dins la nova terminal de l'aeroport de Barcelona, es duen a terme una mitja de 400 vols diaris Una vegada a terra, tots aquests avions han de ser dirigits cap a diferents llocs dins l'aeroport.

Aquest projecte forma part de la iniciativa per crear un sistema automàtic per facilitar el treball que tenen els operadors de l'aeroport de Barcelona quan han d'identificar un avió i dirigir-lo a una pista disponible per aquest.

Utilitzarem una serie de vídeos on es veuen diferents avions apropant-se a la guia d'aterratge i ens centrarem principalment en la extracció de fons i segmentació de la imatge, per quedar-nos només amb l'avió i descartar el fons.

Després de diverses proves amb mètodes d'extracció de fons i segmentació d'imatges, s'ha fet possible trobar una segmentació prou acurada de l'avió, per tal de reconèixer l'avió.

Resumen

En la nueva terminal del aeropuerto de Barcelona, se llevan a cabo una media de 400 vuelos diarios. Una vez en tierra, todos estos aviones deben ser dirigidos hacia diferentes lugares del aeropuerto.

Este proyecto forma parte de la iniciativa de crear un sistema automático para facilitar la labor que tienen los operadores del aeropuerto de Barcelona cuando tienen que identificar un avión y dirigirlo a una pista disponible para éste.

Usaremos una serie de videos donde se ven diferentes aviones acercándose a la guía de aterrizaje y nos centraremos principalmente en la extracción de fondo y segmentación de la imagen, para quedarnos solamente con el avión y descartar el fondo

Después de diversas pruebas con métodos de extracción de fondo y segmentación de imágenes, se ha hecho posible encontrar una segmentación suficientemente cuidadosa, para reconocer el avión.

Abstract

In the new terminal in Barcelona's airport, take place an average of 400 flights per day. Once on land, all these planes must be lead to diferent places in the airport.

This work is part of the initiative to create an automatic system to enease the job of the workers of Barcelona's airport when they have to identify a plane and lead it to an available track for it.
We will use some videos where we can see different planes approaching to the landing guide and we will focus mainly on the background extraction and the image segmentation to keep just the plane and erase the background.

After a few tries with background extraction methods and image segmentation, it was possible to achieve a segmentation approximate enough to recognize the plane.

Index

Capítol 1

Introducció i motivació

Actualment, l'aeroport de Barcelona, així com tots els aeroports de la xarxa nacional, tenen un sistema guiat autònom d'aeronaus que tenen com a destinació un estacionament de passarel·la mitjançant el qual els passatgers accedeixen a l'edifici de la terminal.
El procediment que es porta a terme en un estacionament a finger és el següent:
Des del centre de coordinació d'operacions en plataforma (CECOPS) s'avisa a l'operari del finger de l'arribada d'un tràfic indicant-li el model exacte d'aeronau per que ell, manualment, activi la guia d'atracament especialment configurada per a aquest tipus d'avió.
Aquesta configuració, especial per a cada model d'avió, és necessària degut a la morfologia d'avions amb diferents mesures i tamanys.
La realitat és que a vegades les companyies amb vols regulars canvien de model d'avió per necessitats de manteniment sense avisar prèviament a CECOPS de l'aeroport en destinació, produint una incongruència entre el model que ve reflectit al pla de vol i el que finalment aterra.
A més, moltes vegades la gran afluència de tràfic simultani i la poca quantitat d'assistents disponibles per torn impossibilita la connexió de tots el finger requerits.
D'aquí la possibilitat de millorar el servei que l'aeroport ofereix a les companyies que operen en territori nacional, desenvolupant un sistema de detecció automàtica del model de l'avió per, d'aquesta manera agilitzar el procés d'atracament.

A la figura 1.1 podem veure la guia d'atracament:

Figura 1.1: guia d'atracament.

A continuació (fig 1.2) (fig 1.3)es mostra una guia de l'atracament de les que s'utilitzen avui en dia per l'estacionament d'aeronaus a l'aeroport de Barcelona, més concretament de la Terminal 1 (T1).
Com es pot veure la guia té una càmera just a la part inferior dreta, sota la pantalla de leds.

Figura 1.2: guia d'atracament

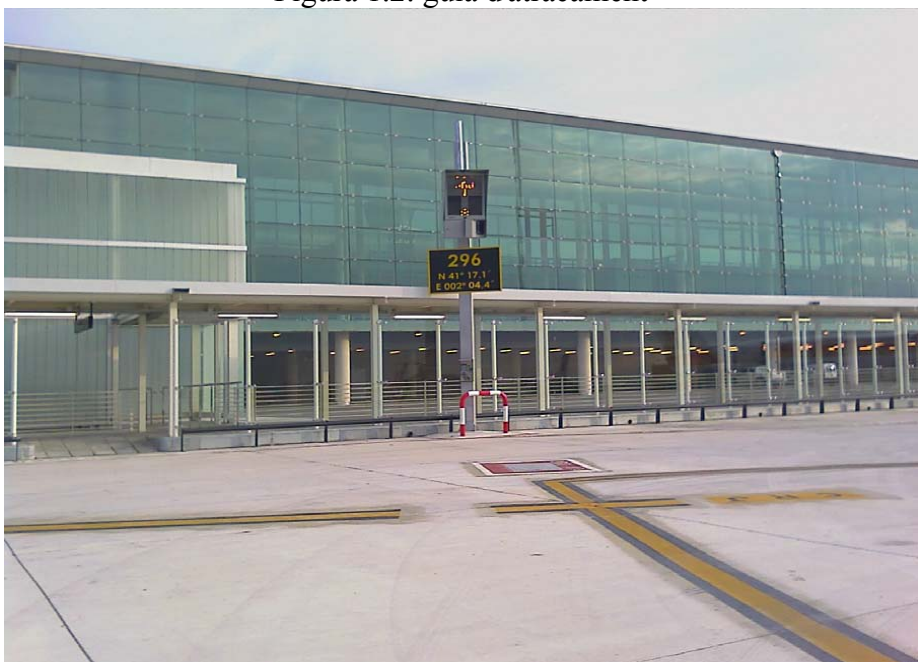


Figura 1.3: guia d'atracament

Les guies estan a una distancia del terra d'uns 5 metres aproximadament, però aquesta ubicació no és sempre la mateixa, poden arribar a estar més altes. A la figura 1.4 es pot veure a quina alçada es trobarà la càmera.

Figura 1.4: guia d'atracament

Depenent del pàrquing, la distància fins la càmera guia, en el moment en el que l'avio inicia el gir per procedir a l'estacionament és variable. En alguns stands com aquest (fig 1.5), són aproximadament uns 50 metres. En altres més, ja que les capacitats i els tamanys dels avions són diferents.

Figura 1.5: stand d'estacionament

A les figures (fig 1.6) i (fig 1.7) es pot veure com un avio s'apropa al punt d'estacionament seguint la guia

Figura 1.6: un avio s'apropa al punt d'estacionament

Amb la finalitat d'agilitzar l'aparcament de l'aeronau, evitar errors dels operaris, o confusions a l'hora de reconèixer el model, que, per excés de feina l'avió s'hagi d'esperar fins a que hi hagi un operari lliure, o altres tipus d'imprevistos, es proposa fer aquest procediment de manera automàtica fent que el programa separi del fons de la imatge l'avió quan s'apropa, pugui reconèixer quin tipus d'avió és, i indicar on hauria d'anar aparcat. Així es faria de manera més ràpida i sense necessitat que un operari de l'aeroport hagués d'estar pendent de fer-ho ell mateix.

Figura 1.7: un avio s'apropa al punt d'estacionament

Punts crítics a considerar serien la il·luminació, ja que el sistema hauria de funcionar de dia i de nit. La similitud dels models d'avions serà la feina més complicada, ja que si ens fixem una mica, són força semblants una vegada encarats en la direcció final de l'estacionament.
A favor hauríem de dir que, a part dels problemes que apareguin per falta de llum, ja sigui de nit o estigui plovent, les oclusions serien pràcticament inexistents, ja que la zona en la que l'avio ha d'estacionar o rodar fins arribar a aquest estarà sempre lliure de tràfic per normativa de seguretat a la plataforma. Es avions estaran sempre orientats igual, la qual cosa ens permetrà evitar la detecció per perspectiva.

Per tal d'aconseguir realitzar el procediment d'extracció de l'avió de la imatge , ens proposem segmentar la imatge per diferenciar entre el fons o background i la part que de la imatge que ens interessa, en aquest cas l'avió, anomenat foreground. Per realitzar aquesta segmentació en un vídeo on podem veure l'aeronau aterrant, seleccionarem els píxels en moviment i així obtindrem l'avió. Això serà possible, fent ús d'una càmera fixa. El mètode d'extracció de fons Adaptive seleccionarà els píxels que canvien d'un frame al següent, i així podrà extreure l'avió d'una seqüència d'imatge.

Tot i això, és molt possible que apareguin taques a la imatge segmentada, que poden ser vehicles movent-se al fons de la pantalla, ocells, operaris de l'aeroport, que el vent faci moure's la càmera, o qualsevol imprevist, per tant també haurem d'utilitzar uns canvis de morfologia, erosions i dilatacions, per tal de netejar la imatge final.

Fent servir aquesta morfologia diferenciarem tres parts de la imatge:

- Delimitarem les seccions de background, on només hi ha fons.
- Trobarem una regió on apareix l'avió, però on podem trobar tant background com foreground.
- Després de tots els canvis morfològics trobarem llavors que pertanyen exclusivament a l'avió, per tant seran foreground i els farem servir per recuperarar tota l'àrea de l'avió mitjançant el mètode d'optimització grabcut.

Amb el mètode grabcut el que farem serà diferenciar quatre parts de la màscara que haurem trobat amb el mètode Adaptive comparant-la amb el frame real del vídeo:

- Background (GC_BGD): imatge de fons
- Foreground (GC_FGD): avió
- Possible Background (GC_PR_BGD): part de la regió on podria aparèixer l'avió, que és Background
- Possible Foreground (GC_PR_FGD): part de la regió on podria aparèixer l'avió, que és Foreground.

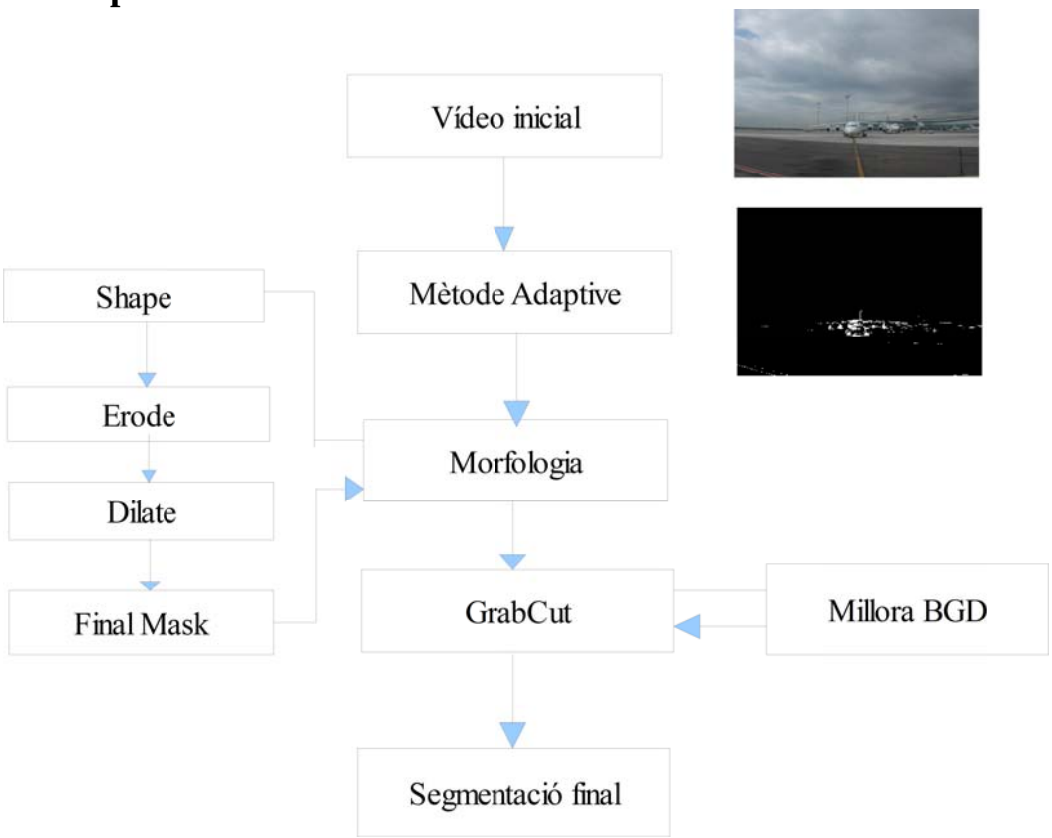
A més, per millorar la segmentació final de la màscara, utilitzarem un nombre reduït de frames consecutius a l'hora de fer servir grabcut. Així els píxels que apareguin com a Foreground a tots els frames anteriors seran considerats conjuntament per obtenir un resultat més acurat al frame actual.

Capítol 2

Metodologia

En aquest capítol explicarem amb detall la manera de funcionar i els principis que fan servir els mètodes que hem utilitzat per realitzar el treball. Començarem explicant el mètode d'extracció de fons Adaptive, en què es basa i quines fórmules utilitza. Després explicarem les operacions morfològiques matemàtiques que hem fet servir per tractar d'aconseguir una segmentació vàlida modificant els resultats que s'obtenen amb Adaptive i per utilitzar-ho amb grabcut. I, finalment, ens queda explicar el mètode de segmentació d'imatges grabcut, que, en un principi requereix de la intervenció d'un usuari, però aquí el tractem de forma totalment automàtica.

Esquema del procés

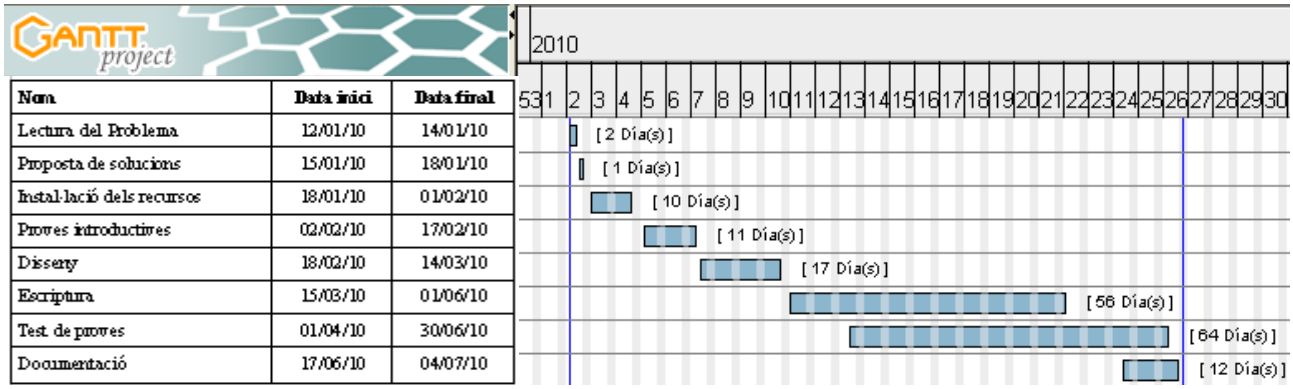


2.1)Planificació i despeses:

En aquest apartat detallarem els terminis dedicats i el pressupost necessari per a cada etapa del projecte.

Per realitzar la planificació els les despeses s'ha definit un equip d'un analista i un programadar amb un termini de 6 mesos per a realitzar el projecte.

A la taula 1 es mostra el diagrama de Gantt amb els terminis previstos per a cada tasca.



Nom	Data inici	Data final
Lectura del Problema	12/01/10	14/01/10
Proposta de solucions	15/01/10	18/01/10
Instal·lació dels recursos	18/01/10	01/02/10
Proves introductives	02/02/10	17/02/10
Disseny	18/02/10	14/03/10
Escriptura	15/03/10	01/06/10
Test de proves	01/04/10	30/06/10
Documentació	17/06/10	04/07/10

Taula 1: Diagrama de Gantt

Donem com a data d'inici el dia 12 de Gener del 2010, quan rebem l'enunciat del problema que cal resoldre.

El primer pas és realitzar la lectura del problema, aclarint bé l'enunciat, els passos a seguir, resolent els possibles dubtes que puguin sorgir i tenir clar els requeriments encomanats.

El segon pas és proposar les solucions possibles. Es decidirà quins mètodes s'utilitzaran per resoldre els problemes, i quin llenguatge s'emprarà. En un dia, s'entén que aquesta tasca pot quedar acabada.

El tercer pas és la instal·lació dels recursos necessaris al maquinari: programes requerits, llibreries que s'empraran, etc. Donarem un termini de deu dies, ja que poden aparèixer problemes inesperats, ja sigui a l'hora de trobar el programari necessari, com de compatibilitat entre ell.

El quart pas és provar tot aquest programari amb problemes de menor envergadura. Si el treball es basa en la segmentació de seqüència d'imatges, provar-ho amb una sola imatge i comprovar les modificacions que es poden fer i com millorar. Donarem un termini d'onze dies.

Al cinqué pas es realitzarà l'anàlisi i el disseny. En aquesta etapa es resumirà d'una manera més formal la informació estreta dels passos anteriors. Aquesta part és molt important, ja que la resta del treball es fonamenta en això. Per realitzar aquesta tasca es concediran 17 dies.

El sisé pas és l'escriptura del codi. Donarem un termini de cinquanta sis dies, ja que s'haurà de modificar sovint.

El seté pas consisteix en les proves. Es farà gairebé simultàniament al procés d'escriptura, ja que els resultats depenen directament dels canvis que es vagin fent al codi.

Finalment realitzarem la documentació on s'inclourà tota la feina realitzada fins el moment. Per aquesta etapa es proposen 12 dies.

Per a calcular les despeses derivades caldrà tenir en compte, a part del treball dels analistes, les llicències del programari i el hardware utilitzat.

Concepte	**Quantitat**	**Preu**	**Preu final**
Microsoft Visual Studio 2008	1	827.25 €	827.25 €
Equip informàtic	1	600,00 €	600,00 €
Hores de feina	246	25€/h	6.150,00 €
Total			7577.25€

Taula 2: Relació de despeses del projecte

2.2) Extracció de fons

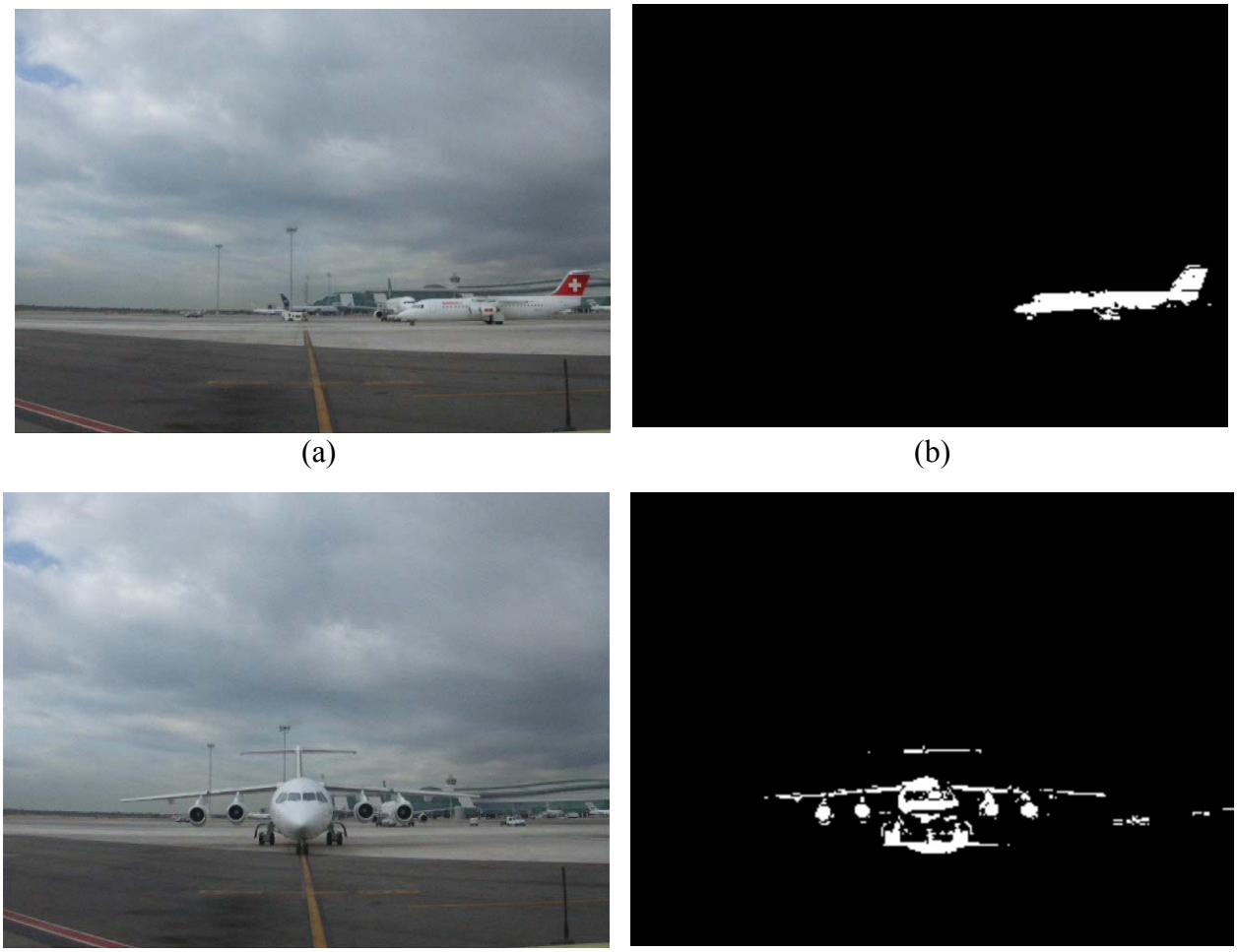
Per realitzar l'extracció de fons en seqüències d'imatges, hem fet servir el mètode Adaptive[1]. Aquest mètode realitza l'aprenentatge de fons a nivell de píxel, guardant informació de l'estat ideal del píxel com a model de fons i la variança permesa sobre aquest valor ideal. Aquest procés es descriu a continuació.
Donat un conjunt de *N* frames corresponents a imatges de fons ideals, construïm un model de densitat de probabilitats per a cada píxel. La representació del fons la realitzem mitjançant la representació a l'espai *(r,g)* on r= R/(R+G+B) i g=G/(R+G+b), on *R*, *G* i *B* corresponen a la intensitat dels canals de vermell, verd i blau per a cada píxel. L'espai de color (r,g) reflexa la informació de color present a les imatges i té l'avantatge de que és invariant a canvis de lluentor. En aquest mètode, cada canal de l'espai de color es tracta de forma independent per reduir el temps computacional. Mitjançant l'aprenentatge de les imatges corresponents a fons, podem obtenir el paràmetre del model per a cada píxel *i* de la imatge a partir d'una mitjana μ_i i una matriu de covariança σ_i calculats sobre l'espai de color (r,g).

Si un nou valor per a un píxel determinat $I_{i,t}$ pertany a la distribució gausiana d'aquest punt, els paràmetres són reajustats mitjançant la resposta d'un filtre d'impulsos infinits, tal i com es mostra a continuació:

$$\mu_{i,t} = (1-\alpha)\,\mu_{i,t-1} + \alpha(I_{i,t})$$
$$\sigma^2_{i,t} = (1-\alpha)\,\sigma^2_{i,t-1} + \alpha(I_{i,t}-\mu_{i,t})^T(I_{i,t}-\mu_{i,t})$$

On α correspon a la tasa d'aprenentatge (pes entre els valors actuals i el nou de l'actualització), el qual determina la velocitat dels canvis a patir pels paràmetres de la distribució.

Aquest mètode ens ofereix una avantatja ,i és la diferenciació entre ombres que puguin desaparèixer a les imatges de fons o altres objectes. En aquests casos l'espai de color (r,g) és útil per a detectar aquestes situacions anòmales. Això és degut a que s'espera que els canvis en la intensitat produïts per ombres es mantinguin dins un rang determinat. És a dir, el rati entre el valor d'intensitat i el valor esperat varien en un determinat rang. Sota aquesta situació, el color (r,g) varia lleugerament. Sota aquesta característica i modelant el rang de variança permès als píxels serem capaços de detectar aquestes situacions anòmales i corregir-les.



(a) (b)

(c) (d)

Figura 2.1. Imatges de l'avió entrant a la pista d'aterratge (a) i (c) i les seves respectives estraccions de fons (b) i (d).

2.3) Morfologia matemàtica. [2]

L'objectiu de les transformacions morfològiques és la extracció d'estructures geomètriques en els conjunts sobre els que s'opera, mitjançant la utilització d'un altre conjunt de forma coneguda denominat element estructurant. La mida i forma d'aquest element es tria, a priori, d'acord a la morfologia del conjunt sobre el qual s'ha d'interaccionar i a l'extracció de formes que es desitja obtenir. Exemples bàsics d'elements estructurants utilitzats en la pràctica es mostren a la figura 2.2.

Figura 2.2: Exemple de formes bàsiques d'elements estructurants plans.

2.3.1) Erosió

En “teoria de reticles” , una erosió és una operació que commuta amb l'ínfim. Donat un reticle complert *X,* una erosió és una funció $\varepsilon: X \rightarrow X$ en la que:

$$\varepsilon\left(\bigwedge_{i\in I} x_i\right) = \bigwedge_{i\in I} \varepsilon(x_i)$$

on I és qualsevol conjunt d'índexs i $\{x_i\}$ és una col·lecció arbitrària de valors x_i pertanyents a X.

La transformació d'erosió és el resultat de comprovar si l'element estructurant Y està totalment inclòs dins el conjunt X. Quan això no passa, el resultat de la erosió és el conjunt buit.

La erosió d'un conjunt X per un element estructurant Y es defineix com el conjunt de punts o elements x, pertanyents a X, de manera que quan un element estructurant Y es trasllada a aquest punt, l'element queda inclòs a X:

$$\varepsilon_Y(X) = \{x \mid Y_x \subseteq X\}$$

La equació anterior pot reformular-se en termes d'una intersecció de conjunts traslladats. Les translacions vénen determinades per l'element estructurant Y:

$$\varepsilon_Y(X) = \bigcap_{s\in Y} X_{-s}$$

L'efecte d'una operació d'erosió pot observar-se a la figura 2.3, en la que un element estructurant *Y*, en forma de disc circular, fa desaparèixer les estructures de menor mida a l'element.

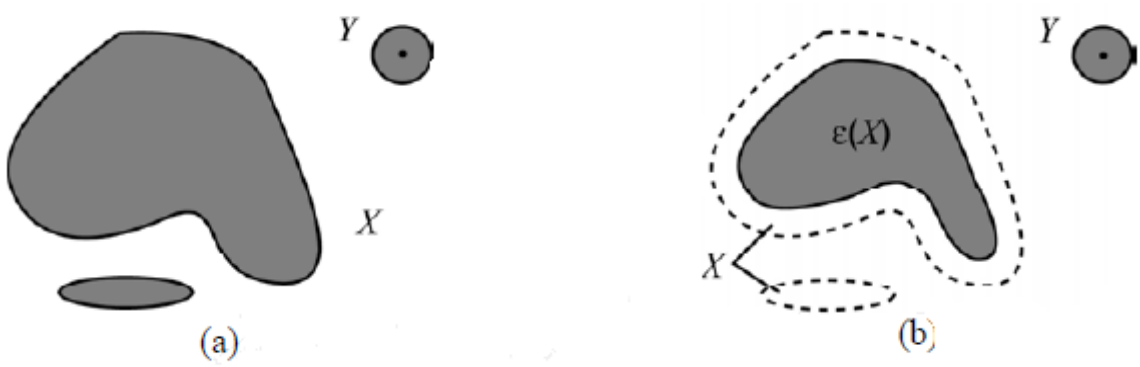


Figura 2.3: Erosió de *X* per un element estructurant *Y*. Els elements connectats del conjunt *X* més petits que *Y* són eliminats.

La última definició d'erosió pot extendre's directament al cas d'imatges binàries i d'escala de grisos. La erosió d'una imatge *f* per un element estructurant Y es denota per $\varepsilon_Y(f)$ i es defineis com el mínim (^) de les translacions de *f* pels elements *s* de *Y*:

$$\varepsilon_Y(f) = \bigwedge_{s \in Y} (f_{-s})$$

A la figura 2.4 es mostra la definició d'un element estructurant quadrat de mida 3x3 pla, amb origen al centre de l'element. L'origen de l'element estructurant és un concepte important, ja que defineix la orientació de translacions. La erosió de la imatge binària de la figura 2.5.a per l'element estructurant anterior es presenta a la figura 2.5.b. És possible observar com els objectes de menor mida al de l'element estructurant desapareixen.

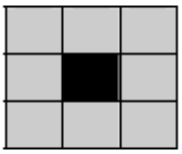


Figura 2.4: Element estructurant pla de mida 3x3. L'origen de l'element se situa al centre

Figura 2.5. Erosió d'una imatge binària mitjançant un element estructurant quadrat 3x3.

Per a imatges, la erosió pot ser definida com:

$$\varepsilon_Y(f)(x,y) = \min_{(s,t)\in Y} f(x+s, y+t)$$

El resultat de la erosió en senyals bidimensionals d'escala de grisos (imatges) és una senyal de menor valor, és a dir, una imatge més fosca, ja que la erosió pretén minimitzar el valor de la senyal que, en el cas dels grisos té una definició [0,255].

Es presenta un cas típic de minimització espaial per una zona d'una imatge la representació de la qual en nivells de grisos es troba esquematitzada a la taula de píxels de la figura 2.6. Per a les operacions morfològiques, l'element estructurant recorre la finestra seleccionada de la imatge. En el cas dels contorns i les cantonades s'ha triat l'aproximació que assegura que l'origen de l'element estructurant recorre tots els píxels de la finestra. Si l'element estructurant és més gran a un píxel i l'origen se situa al centre, als contorns i les cantonades de la finestra l'element estructurant actua en menor definició, al comparar-se el valor d'un menor nombre de píxels.

34	56	34	33	123	124	124	56
78	64	200	201	128	120	230	232
1	23	11	124	123	67	78	87
87	98	201	223	32	17	34	198
45	197	167	158	9	1	1	3
34	178	165	45	10	3	2	11
67	54	55	54	23	56	89	23
56	98	99	149	102	244	203	27

Figura 2.6. Representació d'un conjunt de píxels d'una imatge en escala de grisos

Utilitzant l'element estructurant pla de mida 3x3 definit a la figura 2.5 s'obtendria, com a imatge erosionada, el conjunt de píxels de la figura 2.7.

34	34	33	33	33	120	56	56
1	1	11	11	33	67	56	56
1	1	11	11	17	17	17	34
1	1	11	9	1	1	1	1
34	34	45	9	1	1	1	1
34	34	45	9	1	1	1	1
34	34	54	10	3	2	2	2
54	54	54	23	23	23	23	23

Figura 2.7. Erosió d'una imatge d'escala de grisos per un element estructurant pla bidimensional de mida 3x3.

A la figura 2.8 s'observa com el mateix element estructurant de mida 3x3 utilitzat a l'exemple anterior atenua la lluminositat de la imatge "Lenna". L'efecte visual és que els objectes foscos augmenten la seva definició en front els clars.

(a) (b)

Figura 2.8. Erosió d'una senyal bidimensional (imatge) definida per la funció *f*. Element estructurant de mida 3x3. El resultat en cada punt de la imatge és el mínim de tots els valors presents sota la definició de l'element estructurant.

2.3.2) Dilatació

La dilatació és la operació dual de la erosió. En "teoria de reticles", un operador $\delta: X \dashrightarrow X$ es denomina dilatació en el cas que commuti amb el suprem d'una col·lecció de valors:

$$\delta\left(\bigvee_{i\in I} x_i\right) = \bigvee_{i\in I} \delta(x_i)$$

on I és qualsevol conjunt d'índexs i $\{x_i\}$ és una col·lecció arbitrària de valors x_i que pertanyen a X.

El resultat de la dilatació és el conjunt de punts origen de l'element estructurant Y tals que l'element estructurant conté algun element del conjunt X, quan l'element es desplaça per l'espai que conté ambdós conjunts:

$$\delta_Y(X) = \{x \mid Y_x \cap X \neq 0\}$$

Aquesta última equació pot reescriure's com una unió de conjunts traslladats. Les translacions venen definides pel domini de l'element estructurant:

$$\delta_Y(X) = \bigcup_{s\in Y} X_{-s}$$

L'efecte d'una operació de dilatació pot apreciar-se a la figura 2.9 on un element estructurant *Y* de forma de disc circular augmenta la definició de l'objecte *X*.

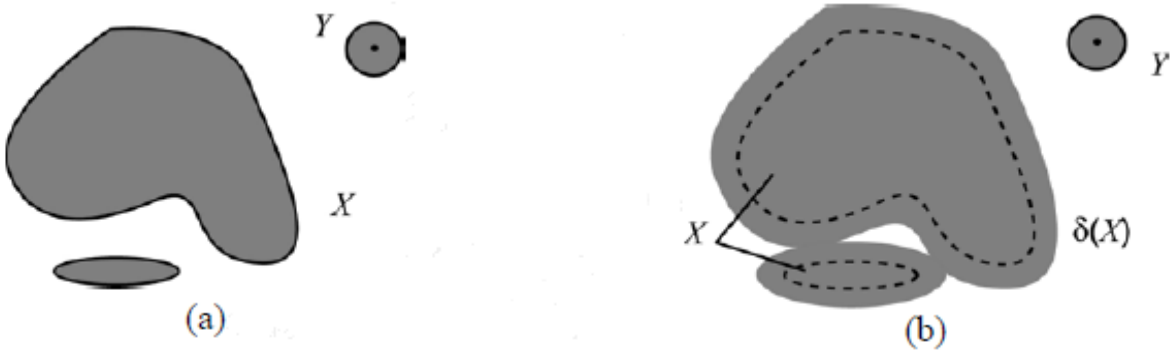


Figura 2.9. Dilatació de *X* per l'element estructurant *Y*. El conjunt *X* augmenta la seva definició.

Es pot extendre la última definició de dilatació a imatges binàries o d'escala de grisos *f*, interpretant la dilatació com el màxim valor de les translacions de *f* (definides per la forma de l'element estructurant) en cada punt de la imatge:

$$\delta_Y(f) = \bigvee_{s \in Y} f_{-s}$$

És a dir, el valor de la dilatació d'un píxel (x,y) és el màxim valor de la imatge en la finestra del veïnat definida per l'element estructurant quan el seu origen se situa en (x,y):

$$\delta_Y(f)(x,y) = \max_{(s,t) \in Y} f(x-s, y-t)$$

La dilatació de la imatge binària de la figura 2.10.a per un element estructruant de mida 3x3 s'il·lustra en la figura 2.10.b. A la imatge, els objectes augmenten la seva definició.

Figura 2.10. Dilatació d'una imatge binària mitjançant un element estructruant de mida 3x3.

El resultat de la dilatació en senyals bidimiensionals d'escala de grisos (imatges) és, generalment, un senyal de major valor, és a dir, una imatge més clara, donat que la dilatació maximitza el valor de la senyal. La dilatació del conjunt de píxels en escala de grisos presents a la figura 2.6 s'observa a la figura 2.11.

78	200	201	201	201	230	232	232
78	200	201	201	201	230	232	232
98	201	223	223	223	230	232	232
197	201	223	223	223	123	198	198
197	201	223	223	223	34	198	198
197	197	197	167	158	89	89	89
178	178	178	165	244	244	244	203
98	99	149	149	244	244	244	203

Figura 2.11. Dilatació d'una imatge d'escala de grisos per un element estructurant pla bidimensional de mida 3x3.

A la figura 2.12 es mostra com l'element estructurant de tamany 3x3 potencia la lluminositat de la imatge original de “Lenna”. L'efecte visual és tal que els objectes clars augmenten la seva definició en front als foscos.

(a) (b)

Figura 2.12. Dilatació d'una senyal bidimensional (imatge) definida per la funció *f*. Element estructurant de mida 3x3. El resultat en cada punt de la imatge és el màxim de tots els valors presents sota la definició de l'element estructurant.

2.4) Segmentació: GrabCut [3] [4] [5] [6] [7]

El mètode GrabCut és una tècnica iterativa de segmentació basada en l'algoritme Graph Cut. GrabCut millora Graph Cut afegint imatges en color i trimapes incomplets. La interacció de l'usuari és simplificada a seleccionar un rectangle al voltant de l'objecte desitjat, seguit per una petita edició correctiva.
Un trimapa és una imatge pre-segmentada consistent en tres regions de foreground, background i desconegut.

L'algoritme de GrabCut consisteix en:

1. L'usuari crea un trimapa inicial seleccionant un rectangle. Els píxels dins el rectangle són marcats com a desconeguts. Els píxels fora del rectangle són marcats com a background.
2. L'ordinador crea una segmentació de la imatge inicial, on tots els píxels desconeguts són inicialment posats a la classe de foreground i els del background conegut es posen a la classe de background.
3. Es creen unes Mescles de Models Gausians (Gaussian Mixture Models – GMMs) per les classes inicials de foreground i background.
4. Cada píxel de la classe foreground és assignat a la component gaussiana més similar de foreground GMM. De la mateixa manera, cada píxel de background és assigntat a la component més similar de background GMM.
5. Els GMMs són rebutjats i s'aprenen nous GMMs dels sets creats al set previ.
6. Es crea un gràfic i s'executa Graph Cut per trobar una nova tria de foreground i background dels píxels.
7. Es repeteixen els passos 4-6 fins que la classificació convergeix.

L'algoritme GrabCut requereix quatre camps d'informació per a cada píxel:

- Color – valor de RGB (z).
- Trimapa – ja sigui TrimapUnknown, TrimapBackground, o TrimapForeground.
- Matte – a la primera segmentació, ja sigui MatteBackground o MatteForeground (α).
- Índex de components – un número entre 1 i K, on K és el nombre de components Gaussianes a un GMM (k).

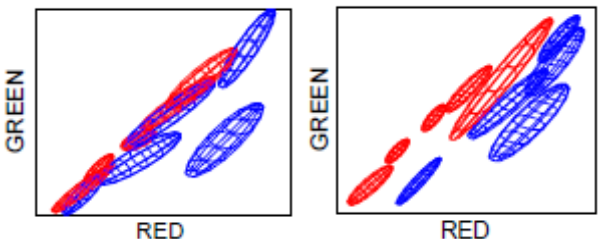
A més d'aquests, GrabCut necessita K components Gaussians per a cada un dels GMMs de foreground i background. Per a cada component emmagatzemem:

- μ – la mitjana (RGB).
- Σ' – l'invers de la matriu de covariància (una matriu 3x3).
- $\det\Sigma$ – el determinant de la matriu de covariància.
- π- component del pes.

2.4.1) Inicialització

El procés d'inicialització inclou els passos 1-3. Al pas 1, l'usuari inicialitza el trimapa seleccionant una regió rectangular al voltant de l'objecte d'interès. Els píxels dintre del rectangle es marquen com a TrimapUnknown. Els píxels fora són marcats com a TrimapBackground. Aquesta és la informació inicial que se li dona a l'algoritme. Al pas 2, la matriu MatteBackground és inicialitzada al set de TrimapBackground i MatteForeground al set de TrimapForeground. Aquesta distinció entre el trimapa i la matriu formalitza la distinció entre l'entrada d'usuari, que s'entén com a correcte, i la segmentació feta per l'algoritme GrabCut, que pot ser incorrecta.

Al pas 3, donada la matriu inicialitzada, creem els K components del GMM per a les regions de MatteForeground i MatteBackground. Això vol dir que hem de crear un total de 2K components. Primer dividim ambdues regions en K clústers de píxels com mostra la figura 2.13. Les components gaussianes són inicialitzades des dels colors a cada clúster. Per una bona distinció entre background i foreground és necessari generar components gaussianes de baixa variança.



(a) (b)

Figura 2.13. GMM a l'espai de colors RGB a la inicialització (a) i després de convergir (b)

2.4.2) Aprenentatge de les components GMM

A mesura que iterem per la part d'aprenentatge de l'algoritme (passos 4-6), la matriu canvia. Això mou alguns píxels de MatteForeground a MatteBackground, i viceversa. Quan això passa, ens interessa actualitzar els GMMs per reflectir les noves distribucions de colors del foreground i background. Una manera òbvia per aconseguir això és tornar a l'algoritme de clustering descrit a la secció d'"inicialització". Encara que això hauria de funcionar, la majoria d'algoritmes de clustering són massa lents per executar-los completament durant cada iteració. En canvi, s'utilitza un algoritme de mescla gaussiana de clusters amb actualització de la matriu per accelerar l'algoritme.

L'algoritme de clustering Gaussià consisteix en dos passos (4 i 5).

Primer, cada píxel al set de MatteForeground és assignat a la component de foreground de GMM que té la major probabilitat de produir el color del píxel. Això es troba simplement avaluant la equació gaussiana amb el color del píxel com a entrada.

De la mateixa manera, assignem píxels al set de MatteBackground amb amb la major semblança a la component de background de GMM. L'estructura de dades d'Índex de Components guarda les components a les quals cada píxel és assignat. Els píxels de foreground només s'assignen a les components de GMM de foreground i viceversa.

Segon, una vegada els píxels són clusteritzats, rebutgem les components Gaussianes actuals i creem unes de noves per a cada parella de Matriu/Component.

2.4.3) Duent a terme Graph Cut

El proper que es fa és construir un gràfic per usar a l'algortime Graph Cut. A Graph Cut hi ha dos tipus de links. N-links connecta els píxels al 8-neighborhood. Aquests links descriuen el preu per posar un límit de segmentació entre píxels veïns. Ens interessa que aquest preu sigui molt alt a regions amb gradient baix i que sigui baix en regions amb gradient alt (límits). Els pesos dels N-links són constants durant tota l'execució de l'algoritme. Encara que poden ser computats una vegada i re-utilitzats. Els T-links connecten cada píxel als nodes de foreground i background. Aquests, descriuen la probabilitat de que cada píxel pertanyi a foreground o background. A GrabCut, aquesta probabilitat està emmagatzemada als GMMs. A mesura que iterem a través dels passos 4-6, els GMMs s'actualitzen i les probabilitats canvien. Això vol dir que el pesos dels T-links han de ser actualitzats durant cada iteració.

Pels N-links el pes apropiat entre els píxels *m* i *n* és:

$$N(m,n) = \frac{50}{dist(m,n)} e^{-\beta \|z_m - z_n\|^2}$$

on Z_m és el color del píxel *m*.

Existeixen dos T-links per a cada píxel. El T-link de Background connecta el píxel al node de Background. El T-link de Foreground connecta el píxel al node de Foreground. Els pesos d'aquests links depenen de l'estat del trimapa. Si l'usuari ha indicat que un píxel en particular és foreground o background definitivament, es reflecteix posant el pes del link per tal que el píxel es forci al grup apropiat. Pels píxels desconeguts usem les probabilitats obtingudes dels GMMs per posar els pesos.

Entenem que O i B denoten els subconjunts de píxels marcats com a llavors de "objecte" i "background". Evidentment, la intersecció d'aquests conjunts és el conjunt buit.

El treball realitzat es mostra a la figura 2.13. Donada una imatge (figura 2.13(a)) podem crear un graf amb dues terminals (figura 2.13 (b)). El següent pas és calcular

el tall més òptim (figura 2.13(c)) separant les dues terminals. Aquest tall proporciona una segmentació (figura 2.13 (d)) de la imatge original. A l'exemple de la figura 2.13, la imatge és dividida entre una regió "objecte" i una regió "background". En general, els mètodes de segmentació generen segmentacions binàries.

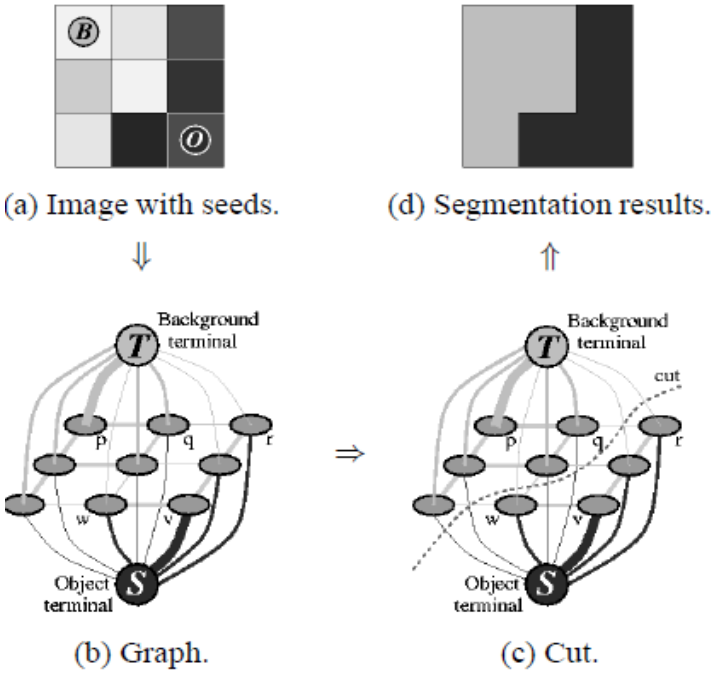


Figura 2.13. Un exemple de segmentació simple per una imatge de 3x3. Les llavors són O i B.

Els pesos de T-link pel píxel *m* són:

Tipus de Píxel	Back. T-link	Fore. T-link
m ε TrimapForeground	0	K
m ε TrimapBackground	K	0
m ε TrimapUnknown	$D_{Fore}(m)$	$D_{Back}(m)$

D_{fore} i D_{back} són les probabilitats de que el píxel pertanyi als GMM de foreground o background respectivament. Aquestes probabilitats són computades de la següent manera pel píxel *m:*

$$D(m) = -\log \sum_{i=1}^{K} \left[\pi(\alpha_m, i) \frac{1}{\sqrt{\det \Sigma(\alpha_m, i)}} \times \exp\left(\frac{1}{2} [z_m - \mu(\alpha_m, i)]^\top \Sigma(\alpha_m, i)^{-1} [z_m - \mu(\alpha_m, i)] \right) \right]$$

on el sumatori es fa sobre el set de components Gaussianes apropiat.

K és un valor calculat per assegurar-nos que és el major pes del gràfic:

$$K = \max_{m} \sum_{n:(m,n)\ \varepsilon\ E} N(m,n)$$

on *E* és el set de tots els contorns que uneixen píxels veïns.
Donats aquests pesos, el graf s'optimitza mitjançant l'algoritme max-flow/min-cut obtenint la segmentació.

En aquest treball, aportem la inicialització automàtica del Background, la regió desconeguda i de les llavors de Foreground, gracies als passos anteriors. Amb el mètode Adaptive, explicat anteriorment i les transformacions morfològiques aplicades, aconseguim la màscara necessària per aplicar l'algoritme GrabCut. D'aquesta manera, no necessitem la intervenció de l'usuari i es converteix en un procés completament automàtic.

A més, ampliem amb una connexió de N-grafs = N-frames per un resultat més acurat. És a dir, per la segmentació del frame *X* utilitzarem *N* frames propers a *X*. Per fer això es crea un graf a partir com faríem amb un sol frame, però se li afegeixen a cada píxel els pesos dels diferents frames com es veu a la figura 2.14.

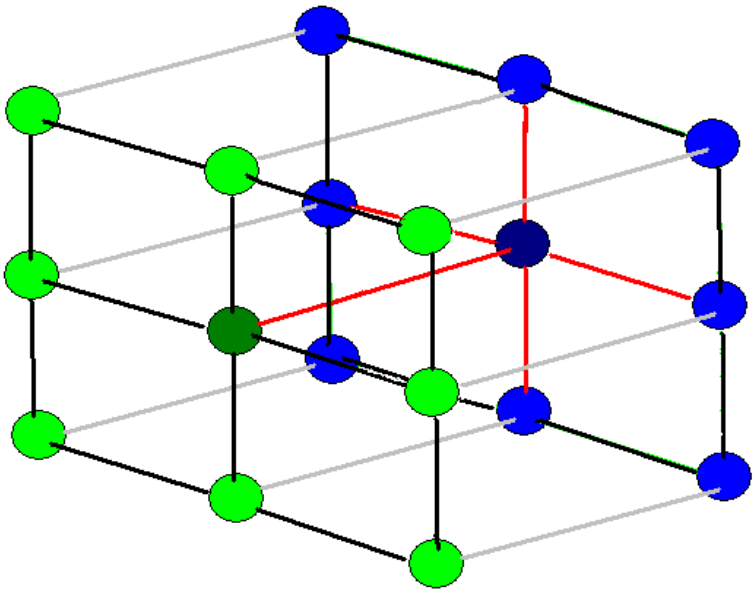


Figura 2.14. Tenim dues imatges de 3x3 (la verda i la blava). El píxel que estem comprovant és el del centre de la imatge blava. Com podem veure és influït pels pesos dels píxels de la mateixa imatge adjacents a ell, i també pel píxel en la mateixa posició del frame següent. Aquest fet es pot extrapolar amb més frames.

Capítol 3

Resultats

En aquest projecte hem fet proves amb 4 vídeos diferents a peu de pista, amb un framerate de 30 frames per segon.
De cada video analitzem 150 frames amb una resolució de 640x480.

Hem fet servir el llenguatge de programació C++ utilitzant el software Microsoft Visual Studio 2008 amb les llibreries de OpenCV 2.1[8] [9].

Per fer funcionar el mètode grabcut cal passar com a paràmetres el frame original del video, la màscara que hem obtingut després de fer servir l'Adaptive, els models Gaussians de foreground i background i, en el nostre cas, un flag per que faci servir la màscara que li enviem directament.

Després d'haver explicat els algoritmes que hem utilitzat, mirem els resultats obtinguts després d'aplicar tot el procés.

Per començar tenim un vídeo d'un avió dirigint-se cap a la càmera, el qual hem de segmentar i aplicar diferents operacions de morfologia.

Agafem, per exemple aquest frame de la figura3.1 on es veu l'avió model BA146 apropant-se a la guia d'aterratge:

Figura 3.1. Frame amb un avió model BA146 apropant-se a la càmera.

Quan segmentem aquest frame amb el mètode Adaptive obtenim la figura 2.2, podem apreciar l'avió, però veiem que també apareixen taques no desitjades corresponents a vehicles que circulen per darrera i petits tremolors de la càmera:

Figura 2.2. Segmentació de la imatge on podem veure els píxels que han canviat en aquest frame respecte el frame anterior.

Aquestes taques poden provocar problemes a l'hora de trobar l'avio en la imatge segmentada, per tant caldrà eliminar-les d'aquesta primera segmentació. Sinó, es

podrien agafar aquestes taques com a part de foreground, en comptes de background i l'aprenentatge seria erroni.

Per tal de fer això farem una primera operació de morfologia matemàtica, en aquest cas la erosió amb un element estructurant de 2x2 i obtindrem la figura 2.3:

Figura 2.3. Imatge bidimensional en escala de grisos erosionada amb un element estructurant 2x2.

Es pot apreciar com han desaparegut la majoria de les taques que podien provocar problemes, així com també han desaparegut parts de l'avió, però això no és problema, ja que, el que necessitarem més endavant és simplement, l'àrea d'un rectangle on estigui l'avio, i unes llavors que siguin segur part de l'avió, per tal que l'algoritme grabcut pugui fer una segmentació definitiva.

Després de la primera erosió, farem una operació convinada d'erosió i dilatació (figura 3.3)per intentar que l'avió no sigui un cúmul de parts diferenciades, sinó intentar que l'avió sigui una taca sencera:

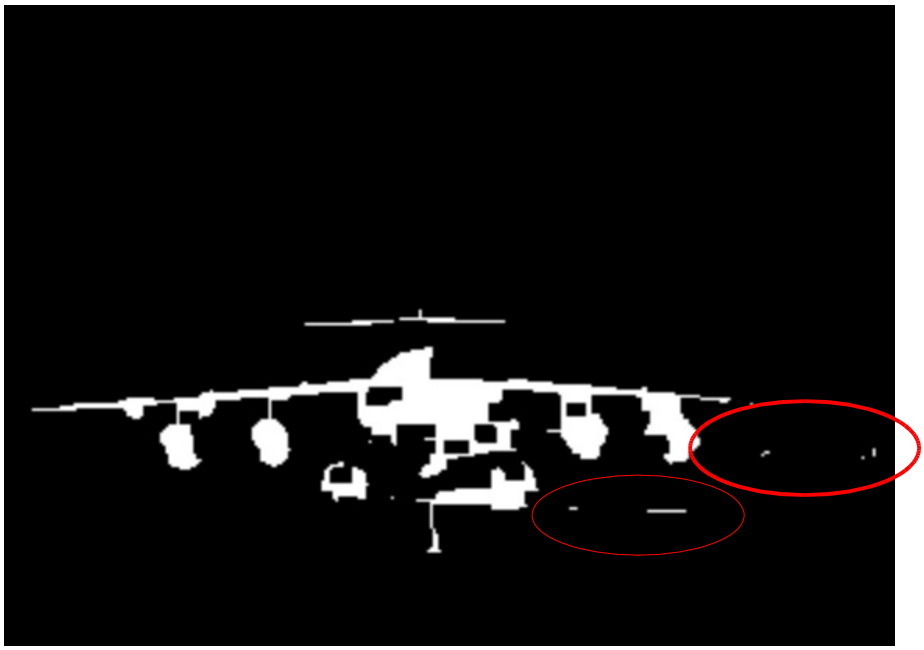


Figura 3.3. Imatge segmentada on apliquem una operació morfològica de tancament amb un element estructurant de mida 8x6

Després de la dilatació, es pot apreciar com les diferents taques que formaven l'avió es van unificar per crear una sola taca més gran.
Tot i això, és complicat que aquest procés sigui del tot acurat, per tant buscarem una modificació morfològica òptima per aconseguir el resultat que volem.
Ara que tenim una taca d'avió més gran, podem aplicar de nou una erosió per tal d'esborrar les taques que encara queden que no són de l'avió

Apliquem de nou una erosió (figura 3.4):

Figura 3.4. Imatge segmentada on apliquem una segona erosió amb un element estructurant de mida 2x2.

Veiem que les taques que no pertanyen a l'avió han desaparegut per complert i ja no molestaran a l'hora de fer la segmentació definitiva.

Per tant, agafarem aquesta imatge per utilitzar-la com a màscara a l'algoritme grabcut. Per tal de fer això cal aplicar dos canvis:

1. Cal trobar un rectangle que contingui l'àrea on estarà l'avió i posar-la com a unknown, és a dir, no se sap si és background o foreground.
2. Cal simplificar la imatge transformant-la d'una imatge amb 256 colors a una de només 4 colors.

3.1) Rectangle d'àrea desconeguda:

Per poder trobar un rectangle adient a les nostres necessitats el que farem serà buscar a la màscara la taca més gran i construir un rectangle al seu voltant.

Utilitzarem la llibreria de Blobs, per poder trobar aquesta taca. Quan ja tenim aquesta taca localitzada, estirarem l'àrea per tal que ocupi tota l'amplada de la imatge, ja que, amb les erosions, hem pogut provocar que les puntes de les ales quedessin escurçades (figura3.5).

Figura 3.5.Imatge segmentada on reconeixem automàticament les àrees de Background (negre), Unknown (blanc) i Foreground (gris).

Com es pot comprovar a la imatge tenim 3 regions diferenciades:

- El Background (GC_BGD): que és el fons de la imatge, i ho tenim representat pel color negre.
- El Foreground (GC_FGD): l'objecte en que estem interessats, en aquest cas és l'avió, i es representa pel color gris.
- La franja desconeguda (GC_PR_FGD o GC_PR_BGD): és la part on podria haver avió o fons i està representat pel color blanc.

3.2) Simplificació de la imatge:

Per tal d'utilitzar la imatge que hem obtingut després de fer la primera segmentació amb l'algoritme Adaptive i aplicades les diferents modificacions morfològiques com a màscara a l'algoritme grabcut caldrà donar-li el format adient.

La imatge obtinguda té una gamma de colors de 0 a 255 i l'algoritme grabcut només accepta màscares amb un total de 4 colors per diferenciar les zones distribuïts d'aquesta manera:

- Negre (0) pel Background
- Gris fosc (1) pel Foreground
- Gris clar (2) pel UnknownBackground
- Blanc (3) pel UnknownForeground

Per tant, el que farem serà el següent:

- Els píxels de la màscara que siguin negres (0) els posarem GC_BGD: mask.setTo(GC_BGD, mask==0)
- Els píxles de la màscara que siguin també negres (2) també els posarem a GC_BGD: mask.setTo(GC_BGD, mask==0)
- Els píxels blancs, que en la imatge serien els de major valor (agafarem a partir del 100), els posarem com a GC_PR_BGD (o GC_PR_FGD) és a dir, 2 : mask.setTo(GC_PR_BGD, mask>=100). En aquest cas és indiferent que posem GC_PR_BGD o GC_PR_FGD, ja que és la zona que modificarà l'algoritme grabcut.
- Per acabar, només ens queden els píxels de foreground, per tant, tots els píxels amb valor major que 3 els posarem com a GC_FGD: mask.setTo(GC_FGD, mask>3)

3.3) Segmentació amb màscara:

Quan ja tenim la màscara preparada per fer-la servir, arriba el moment d'utilitzar l'algoritme grabcut.

Aquest el trobem dins la llibreria d'OpenCV amb aquesta capçalera:

void cv::grabCut (const Mat& img, Mat& mask, Rect rect, Mat & bgdModel, Mat& fgdModel, int iterCount, int mode)

on *img* és la imatge original; *mask* és la màscara que utilitzarem, *rect* és un rectangle que conté la regió d'interés, *bgdMode* i *fgdModel* són els models de les Gaussianes de background i foreground respectivament, que omplirà l'algoritme; *iterCount* és el nombre d'iteracions que volem que faci l'algoritme; *mode* pot ser:

- GC_INIT_WITH_RECT inicialitza la màscara a partir del rectangle rect, que és l'àrea d'interés on es troba l'objecte que es vol segmentar. És el mètode bàsic i més fàcil.
- GC_INIT_WITH_MASK assumeix que ja proporciones una màscara vàlida a l'algoritme. És el que farem servir.

Aplicarem l'algoritme grabcut per aconseguir una primera segmentació (figura 3.6) utilitzant la màscara que hem obtingut amb la imatge corresponent.

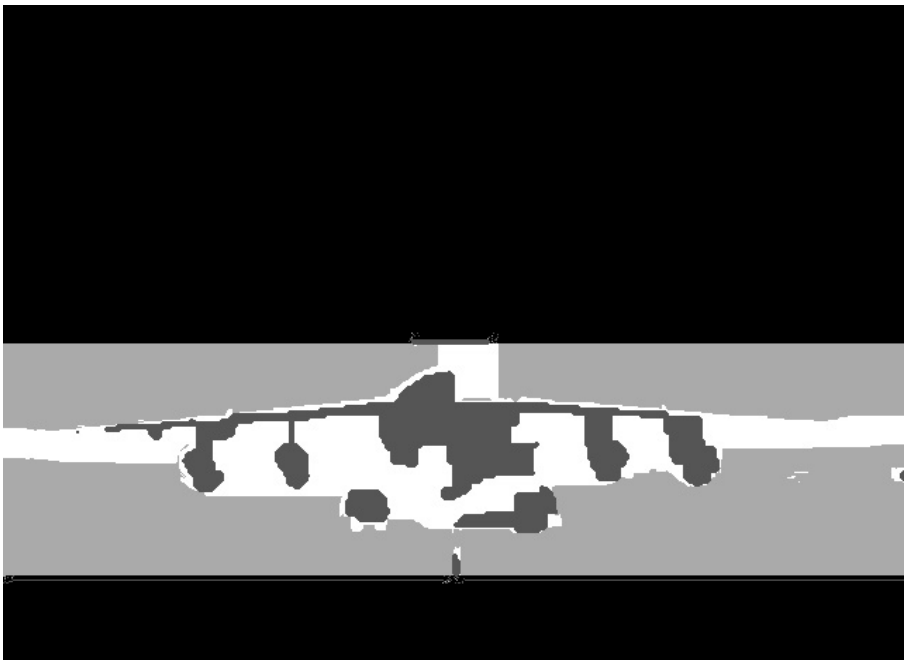


Figura 3.6. Segmentació amb grabcut on es distingeixen quatre àrees.

Com es pot comprovar, amb una primera aplicació del mètode grabcut no n'hi ha prou per aconseguir una segmentació suficientment acurada, ja que algunes zones marcades com a possible avió, es veu clarament que contenen parts del fons.

Per ajudar aconseguir un millor resultat caldrà tornar a aplicar el mètode, però aquest cop, havent fet unes petites modificacions.

El que farem serà comprovar el grau d'incertesa en els píxels de les regions de GC_PR_BGD i GC_PR_FGD mitjançant els models Gaussians. Si un píxel és més provable com a Background el posarem directament com a Background, i tots els demés píxels com a Foreground. El resultat que queda és el de la figura 3.7:

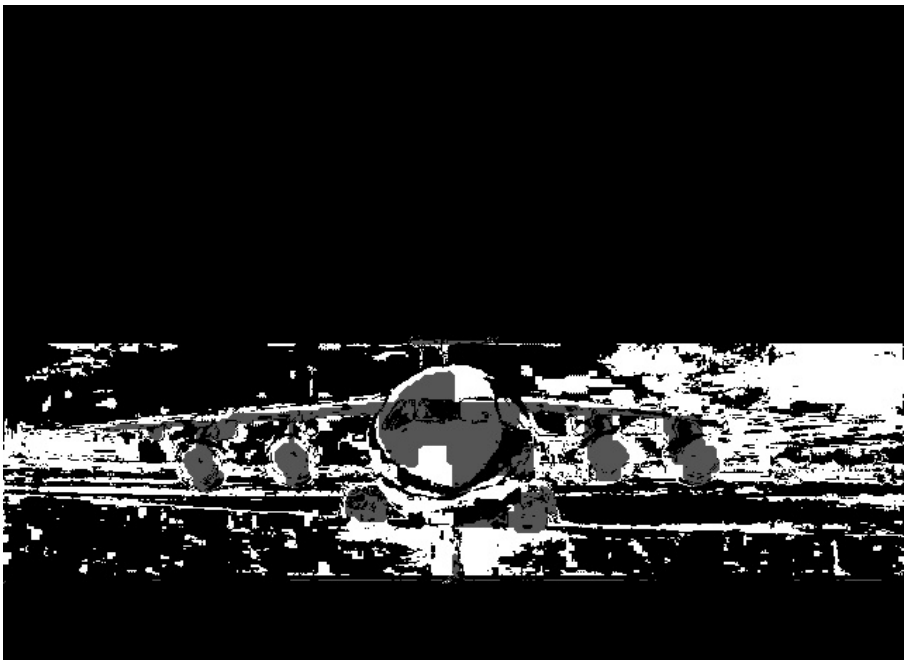


Figura 3.7. Imatge segmentada havent comprovat la possibilitat de que els píxels d'incertesa fossin de Background o de Foreground.

Una vegada aplicats aquests canvies es pot veure que l'àrea pertanyent a Background es fa més gran, cosa que ajudarà a segmentar més correctament més endavant. Això es fa, perquè en zones complicades, com podria ser l'àrea entre els motors de l'avió sempre ho reconeixia com a Foreground.

Una vegada fets aquests canvis tornem a aplicar el mètode grabcut sobre aquesta imatge aconseguint el resultat de la figura 3.8:

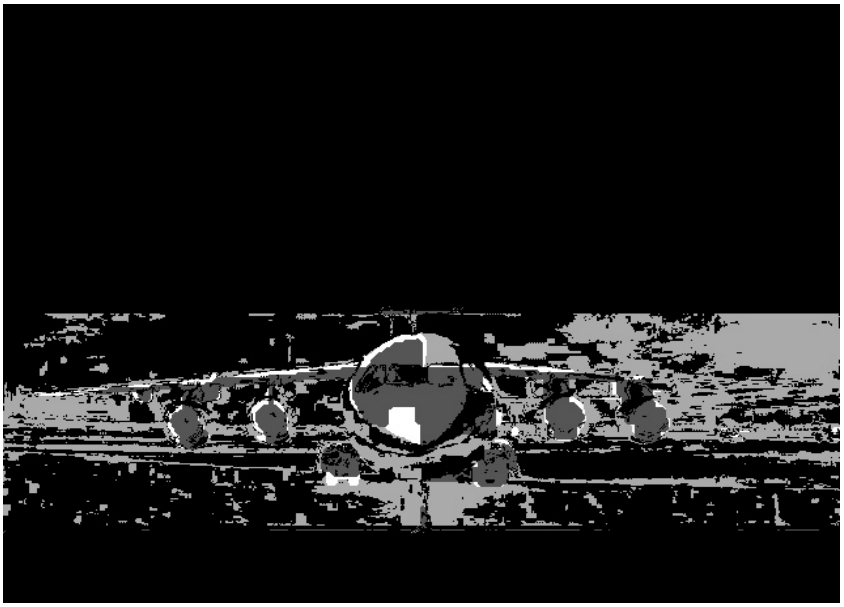


Figura 3.8. Imatge segmentada amb grabcut per segona vegada havent aplicat els canvis corresponents a la detecció de Background.

Com es pot veure en aquesta imatge, el canvi més significatiu és que, la major part del que abans era marcat com a incertesa de Foreground ara és marcat com a incertesa de Background. Només manté com a incertesa de Foreground algunes parts que és segur que pertanyen a l'avió.

El següent pas serà repetir la operació d'abans en que reconeixiem com a Background les parts d'incertesa de Background i el resultat és el de la figura 3.9:

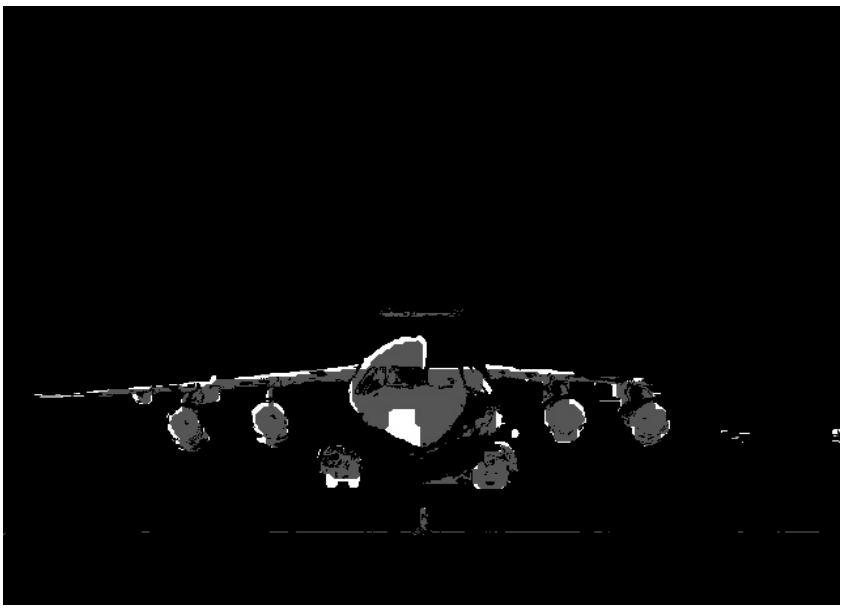


Figura 3.9. Segmentació de l'avió

3.4) Altres resultats

En aquest apartat mostrarem altres resultats obtinguts amb la segmentació aplicada com hem explicat en l'apartat anterior.

En la figura 3.10 podem veure el model d'avió BA146

Figura 3.10 Avió BA146

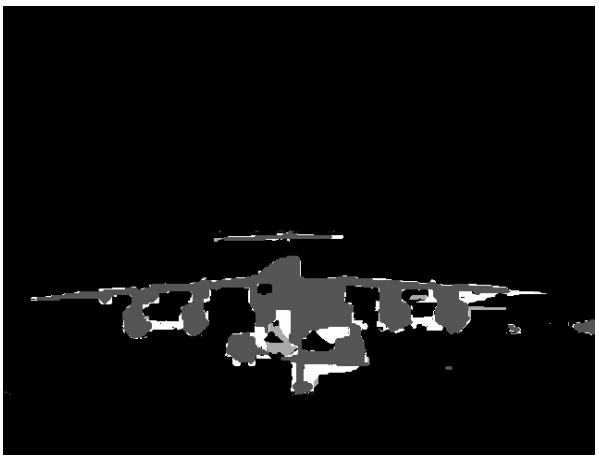


Figura 3.10 (a) segmentació amb un frame

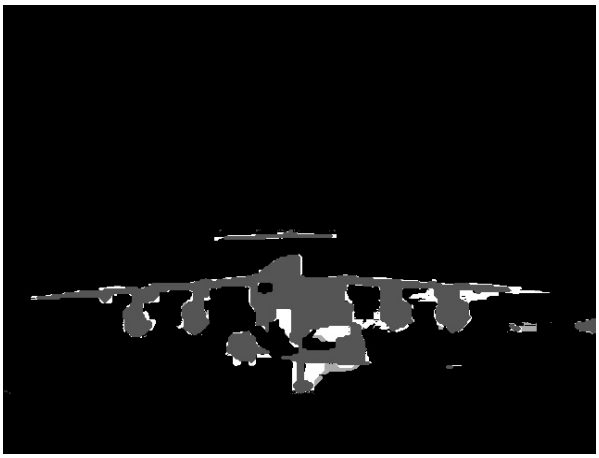


Figura 3.10 (b) segmentació amb 5 frames

Com es pot apreciar als resultats de les segmentacions, quan fem servir més d'un frame per segmentar, es concreta una mica més a l'hora d'extreure el fons de la imatge. No es pot considerar una millora substancial, ja que la segmentació final és molt semblant, però si es pot apreciar un canvi.

A la figura 3.11 podem veure el model d'avió B717:

Figura 3.11 Model d'avió B717

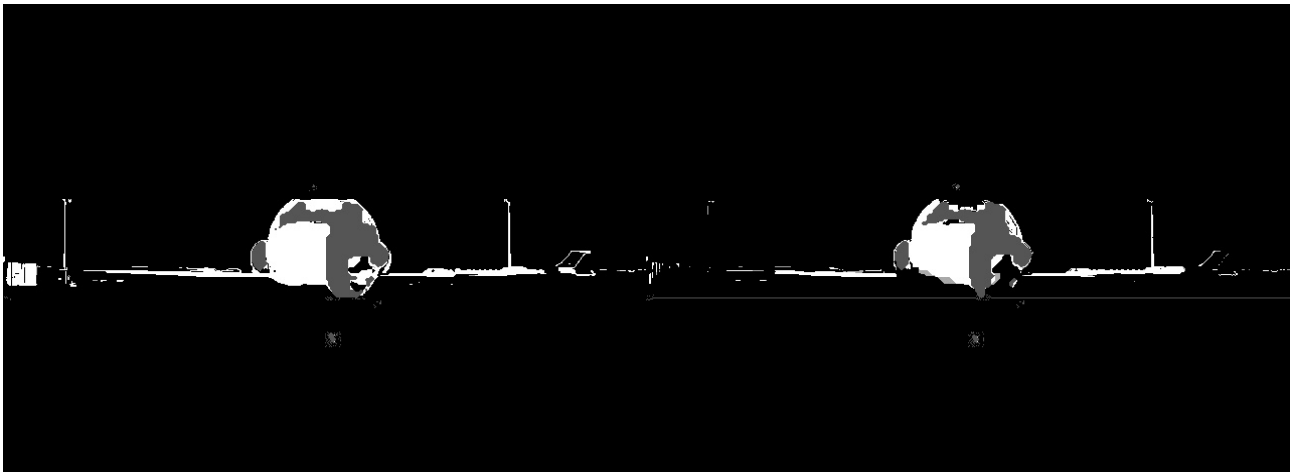


Figura 3.11 (a) segmentació amb un frame

Figura 3.11 (b) segmentació amb 5 frames

Com es pot veure, en aquest cas, les llavors són mínimes, no obstant la segmentació final és força acurada. El que aconseguim utilitzant més frames és eliminar background, tot i això encara es poden veure figures del fons com les faroles, o la cua d'un altre avió.

A la figura 3.12 podem veure el model d'avió A320:

Figura 3.12. Model d'avió A320

Figura 3.12 (a) segmentació amb un frame

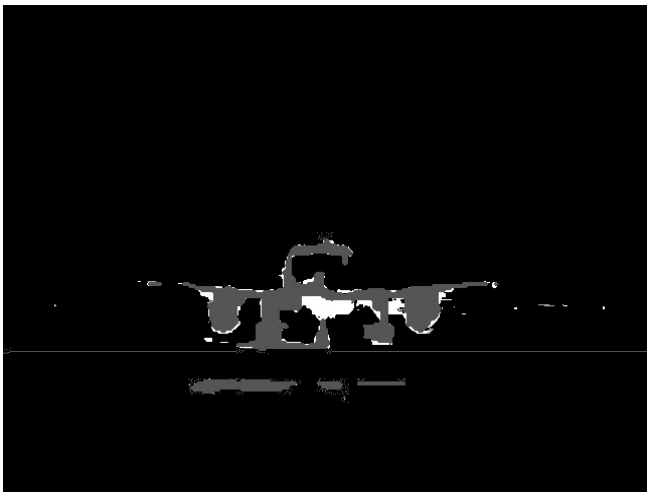


Figura 3.12 (b) segmentació amb 5 frames

En aquest vídeo ens trobem amb el problema del clima. Es pot veure que havia plogut, i la gamma de color del fons és molt semblant a la de l'avió, per això no es pot segmentar tan bé com es voldria. A més també es pot apreciar que el reflex a la bassa d'aigua de l'avió apareix com a foreground, i és evident, que això no és el desitjat. Tot i això, el tret més característic d'aquesta aeronau, que són els motors si que es poden diferenciar bé.

Capítol 4

Conclusions i treball futur

Aquest projecte ha estat realitzat amb la finalitat de ser incorporat en altres treballs com a part de reconeixement d'objectes, concretament, el reconeixement d'avions. Ha. s'ha desenvolupat un software en llenguatge visual C++ Ens hem centrat en la part de segmentació de la imatge a partir d'un vídeo d'entrada per poder crear una màscara prou acurada per que, després, es puguin comparar amb imatges d'avions i reconèixer el model.

Per realitzar la extracció de les imatges a partir d'un vídeo s'han utilitzat funcions pertanyents a les llibreries de openCV i blobs.

Els resultats obtinguts han estat més o menys acurats depenent del vídeo. En alguns casos s'han trobat problemes a l'hora de segmentar per culpa del moviment de la càmera, que es movia a causa del vent i complica la segmentació.
En altres casos, l'ombra de l'avió també ha estat un problema, ja que en la segmentació es confonia amb el mateix avió.

La complicació de fer tot aquest procés de manera automàtica és que, el que amb l'ull humà veuríem clarament que és part de l'avió i ho podríem indicar, quan ho fa la màquina, si els píxels no canvien d'un frame a l'altre, ho entén com a part del background, cosa que s'ha vist quan l'avió es col·loca de cara i es va apropant a la càmera. En aquest moment, els píxels que formen el morro de l'avió, la majoria de vegades, es mantenen iguals.

A més, les ales de l'avió són tan primes que, si al fons de la imatge hi ha edificis, o altres coses que no siguin regulars, les pot confondre amb aquest fons. Com hem vist als resultats, les ales de l'avió moltes vegades són difícils de segmentar completament.

Tot i això, en acabar de fer totes les segmentacions i experiments, aconseguim un resultat suficientment bo per reconèixer l'avió i no agafar com a foreground parts del background.

Per aconseguir una segmentació més acurada, vam utilitzar els frames propers al frame que estem segmentant procurant, així tenir en compte el moviment dels píxels

propers.
Amb aquesta solució vam aconseguir netejar la imatge segmentada i esborrar motes de pols que apareixen quan no s'utilitza la dimensió temporal per fer la segmentació, encara que, de vegades, no s'aconsegueix polir del tot.

A l'hora d'executar el programa, el temps d'execució per un sol frame era d'uns 20 minuts, mentre que quan fèiem servir 5 frames trigava més d'una hora i mitja. Per tant, les millores que es veuen són tan petites, i la diferència de temps és tan gran, que no es pot considerar una solució del tot eficient.

Bibliografia

[1] Metodologia i algoritmia
[2] Procesamiento Morfológico de Imágenes en Color. Aplicación a la Reconstrucción Geodésica. `

[3] Implementing GrabCut Justin F. Talbot Xiaoqian Xu
[4] “GrabCut” — Interactive Foreground Extraction using Iterated Graph Cuts
[5] ***Interactive Graph Cuts* for Optimal Boundary & Region Segmentation of Objects in N-D Images**
[6]http://www.emgu.com/wiki/files/2.1.0.0/html/cc88a069-dc27-e685-0229-e42aafeddd97.htm
[7]http://www.comp.leeds.ac.uk/vision/opencv/opencvref_cv.html
[8]http://opencv.willowgarage.com/documentation/cpp/reading_and_writing_images_and_video.html
[9] https://code.ros.org/svn/opencv/trunk/opencv/