



Escola Politècnica Superior

*Enginyeria Tècnica industrial,
Especialitat en Electrònica Industrial*

Projecte de final de carrera

“Avaluació experimental de tècniques d’Odometria visual basades en RANSAC”

2014-2015

Rafel Quetglas Coll

Dirigit per:

Dr. Antoni Burguera Burguera

Índex

Capítol 1. Introducció.....	6
1. Odometria i acumulació d'errors.....	6
2. Odometria visual.....	8
3. Motivació.....	9
4. Objectius.....	9
5. Metodologia.....	10
Capítol 2. Calibració de la càmera.....	12
1. Objectius.....	12
2. Model Pin-Hole.....	13
2.1 Sistema de coordenades.....	13
2.2 Paràmetres del sistema.....	15
2.3 Càlcul de projeccions.....	15
2.4 Efectes de distorsió.....	18
3. Calibració amb MATLAB.....	19
3.1 Resultats de calibració.....	21
Capítol 3. Recerca de característiques.....	23
1. Objectius.....	23
1.1 Features.....	23
1.1.1 Propietats dels features.....	24
1.2 Detectors.....	25
1.3 Descriptors.....	28
2. SIFT.....	28
2.1 Detector de features.....	29
2.1.1 Espai escalar Gaussià.....	29
2.1.2 Diferència de Gaussianes (DoG).....	31
2.1.3 Localització de característiques.....	33
2.2 Descriptor de features.....	33
2.2.1 Assignació d'orientació als keypoints.....	34
2.2.2 Creació de descriptors.....	35
3. Matching.....	36
3.1 Parametrització.....	37
4. Algorisme: detecció i matching.....	38
4.1 Apartat 1: Anàlisi de les imatges.....	40
4.1.1 Lectura de les imatges.....	41
4.1.2 Detecció i descriptor SIFT.....	42
4.1.3 Eliminació de la distorsió dels punts.....	42
4.2 Apartat 2: Matching i configuració.....	44
4.2.1 Parametrització.....	45
4.2.2 Origen de coordenades.....	45
4.2.3 Transformació píxel - metre.....	46
Capítol 4. Càlcul de moviment.....	48
1. Mínims Quadrats.....	48
1.1 Equacions.....	49
1.2 Aplicació al nostre programa.....	50

1.3	Problemes	52
2.	RANSAC.....	53
2.1	Algorisme.	54
2.2	Parametrització de RANSAC	56
2.3	Aplicació al programa.....	57
3.	Implementació de l'algorisme.....	58
4.	Visualització per pantalla.	61
5.	Paràmetres de sortida del programa.	62
5.1	Composició del moviment.....	62
5.2	Fitxer de sortida.	63
6.	Configuració del programa.	66
6.1	Arxiu d'entrada.....	66
7.	Interfície de sortida.....	67
	Capítol 5. Proves experimentals.....	69
1.	Experiments i vídeos.	69
2.	Mesures de l'error.	71
3.	Problemes amb el càlcul d'errors.	71
4.	Paràmetres avaluats.	74
5.	Resultat de les proves.	75
5.1	Resultats obtinguts del vídeo 1.....	76
5.2	Resultats obtinguts del vídeo 2.....	80
5.3	Resultats obtinguts del vídeo 3.....	84
5.4	Resultats obtinguts del vídeo 4.....	88
5.5	Resultats obtinguts del vídeo 5.....	92
5.6	Paràmetres finals.	96
	Capítol 6. Conclusions.....	97
1.	Conclusions de la calibració de la càmera.....	97
2.	Conclusions de la recerca de característiques.	97
3.	Conclusions del càlcul de moviment.....	97
4.	Conclusions de les proves finals.....	98
	Capítol 7. Propostes de millora	99
1.	Unificació del programa.....	99
2.	Millora de programació.	99
3.	Experimentació amb més vídeos.	100
4.	Utilització del Ground Truth.	100

Índex de Figures

Figura 1 Encoder introduït en una roda.....	6
Figura 2 Tipus d'errors.....	7
Figura 3 Exemple de visió per computador.....	8
Figura 4 Esquema general del projecte.....	10
Figura 5 Representació del model Pin-Hole.....	13
Figura 6 Sistema de coordenades emprats pel model Pin-Hole.	14
Figura 7 Perspectiva de coordenades càmera a pla imatge.....	16
Figura 8 Imatge presa amb un objectiu fish-eye.....	18
Figura 9 Menú principal del programa “toolbox_calib”.....	19
Figura 10 Reconeixement d'imatges de “toolbox_calib”.....	20
Figura 11 Selecció dels cantons del taulell d'escacs.	20
Figura 12 Cantons seleccionats del taulell d'escacs.....	21
Figura 13 Comparació entre imatge distorsionada.....	22
Figura 14 Exemple de feature.	24
Figura 15 Recerca de Features “interessants”	26
Figura 16 Finestra d'operació de Harris.	26
Figura 17 Feature tipus SIFT.....	29
Figura 18 Finestra per detecció de features.	30
Figura 19 Piràmide Gaussiana composta per 6 escales i 5 octaves.	31
Figura 20 Espai Gaussià, Diferència de Gaussianes.....	31
Figura 21 Piràmide de Diferència de Gaussianes (DoG).....	32
Figura 22 Recerca de màxima en l'espai de diferència de Gaussianes	33
Figura 23 Ponderacions SIFT	34
Figura 24 Subregions descriptors.....	35
Figura 25 Carpeta, localització de les imatges.	42
Figura 26 Transformació centre de coordenades.	46
Figura 27 Distància píxel i distància real.	47
Figura 28 Dataset de imatges consecutives.	48
Figura 29 Matching de les dues imatges de prova.....	51
Figura 30 Resultat de estimació de moviment amb mínims quadrats	51
Figura 31 Rotació de la imatge B.....	52
Figura 32 Superposició de imatges	52
Figura 33 Matching entre imatges.	53
Figura 34 Algorisme de RANSAC.	54
Figura 35 Exemple d'eliminació de matchings amb RANSAC.....	58
Figura 36 Visualització per pantalla de l'anàlisi.	61
Figura 37 Arxiu de sortida del programa.....	64
Figura 38 Noms dels fitxers de sortida.....	65
Figura 39 Arxiu d'entrada al programa.	66
Figura 40 Interfície amb l'usuari: Odometria.	68
Figura 41 Instruments emprats per dur a terme les proves.	70
Figura 42 Exemple de Ground Truth i odometria.	72

Figura 43 Exemple mala odometria amb baix error.....	73
Figura 44 Odometria amb vibració de la càmera.....	73
Figura 45 Resultat del vídeo 1.....	76
Figura 46 Gràfica de resultats de Mínims Quadrats del vídeo 1.....	77
Figura 47 Gràfica d'errors RANSAC del vídeo 1.....	78
Figura 48 Resultat del vídeo 2.....	80
Figura 49 Gràfica de resultats de Mínims Quadrats del vídeo 2.....	81
Figura 50 Gràfica de resultats de RANSAC del vídeo 2.....	82
Figura 51 Resultat del vídeo 3.....	84
Figura 52 Gràfica de resultats de Mínims Quadrats del vídeo 3.....	85
Figura 53 Gràfica de resultats de RANSAC del vídeo 3.....	86
Figura 54 Resultat del vídeo 4.....	88
Figura 55 Gràfica de resultats de Mínims Quadrats del vídeo 4.....	89
Figura 56 Gràfica de resultats de RANSAC del vídeo 4.....	90
Figura 57 Resultat del vídeo 5.....	92
Figura 58 Gràfica de resultats de Mínims Quadrats del vídeo 5.....	93
Figura 59 Gràfica de resultats de RANSAC del vídeo 5.....	94

Índex de Taules

Taula 1 Paràmetres de la càmera.....	21
Taula 2 Exemple de vector de sortida de l'algorisme FLANN.	37
Taula 3 Taula d'errors de Mínims Quadrats del vídeo 1.	76
Taula 4 Taula d'errors de RANSAC del vídeo 1.....	77
Taula 5 Error acumulat utilitzant Mínims Quadrats del vídeo 1.	78
Taula 6 Error acumulat utilitzant RANSAC (70%, SELEC=1) del vídeo 1.....	79
Taula 7 Taula de temps estimat amb Mínims Quadrats del vídeo 1.....	79
Taula 8 Taula de temps estimat amb RANSAC (70%, SELEC=1) del vídeo 1.....	80
Taula 9 Taula d'errors de Mínims Quadrats del vídeo 2.	81
Taula 10 Taula d'errors de RANSAC del vídeo 2.....	82
Taula 11 Error acumulat utilitzant Mínims Quadrats del vídeo 2.	82
Taula 12 Error acumulat utilitzant RANSAC (70%, SELEC=2) del vídeo 2.	83
Taula 13 Taula de temps estimat amb Mínims Quadrats del vídeo 2.....	83
Taula 14 Taula de temps estimat amb RANSAC (70%, SELEC=2) del vídeo 2.....	83
Taula 15 Taula d'errors de Mínims Quadrats del vídeo 3.	85
Taula 16 Taula d'errors de RANSAC del vídeo 3.....	85
Taula 17 Error acumulat utilitzant Mínims Quadrats del vídeo 3.	86
Taula 18 Error acumulat utilitzant RANSAC (70%, SELEC=1) del vídeo 3.	86
Taula 19 Taula de temps estimat amb Mínims Quadrats del vídeo 3.....	87
Taula 20 Taula de temps estimat amb RANSAC (70%, SELEC=1) del vídeo 3.....	87
Taula 21 Taula d'errors de Mínims Quadrats del vídeo 4.	88
Taula 22 Taula d'errors de RANSAC del vídeo 4.....	89
Taula 23 Error acumulat utilitzant Mínims Quadrats del vídeo 4.	90
Taula 24 Error acumulat utilitzant RANSAC (70%, SELEC=1) del vídeo 4.	90
Taula 25 Taula de temps estimat amb Mínims Quadrats del vídeo 4.....	91
Taula 26 Taula de temps estimat amb RANSAC (70%, SELEC=1) del vídeo 4.....	91
Taula 27 Taula d'errors de Mínims Quadrats del vídeo 5.	92
Taula 28 Taula d'errors de RANSAC del vídeo 5.	93
Taula 29 Error acumulat utilitzant Mínims Quadrats del vídeo 5.	94
Taula 30 Error acumulat utilitzant RANSAC (70%, SELEC=2) del vídeo 5.	94
Taula 31 Taula de temps estimat amb Mínims Quadrats del vídeo 5.....	95
Taula 32 Taula de temps estimat amb RANSAC (70%, SELEC=2) del vídeo 5.....	95
Taula 33 Taula amb les millors opcions per vídeo(Mínims Quadrats).	96
Taula 34 Taula amb les millors opcions per vídeo (RANSAC).....	96
Taula 35 Elecció final de paràmetres.	96

Capítol 1. Introducció

En el present projecte evaluarem experimentalment algunes tècniques d'odometria visual basades en Mínims Quadrats i Random Sample Consensus (RANSAC). La robustesa d'aquestes tècniques és fonamental per a moltes aplicacions en diversos camps com la robòtica, l'automoció o la medicina.

Cercarem aquesta robustesa implementant un odòmetre visual en C++ i amb diferents experiments podrem observar quins paràmetres del nostre programa seran els que major robustesa proporcionen.

Amb aquesta finalitat plantejarem distints tipus d'experiments i definirem criteris quantitatius per a avaluar-los.

Al llarg del document s'explicarà com hem anat desenvolupant el projecte. Per això primerament explicarem teòricament tots els apartats importants del projecte i a la finalització de cada apartat s'exposarà l'aplicació pràctica que haurem implementat al nostre programa.

1. Odometria i acumulació d'errors.

Les tècniques d'odometria fan servir sensors proprioceptius per tal d'estimar petits canvis en la posició de vehicles o robots. Acumulant aquestes estimacions al llarg del temps es pot calcular la posició del vehicle respecte del seu punt de partida.

Els sensors odomètrics solen estar localitzats en punts estratègics com les rodes o les cames d'un robot. Els *encoders* (*Figura 1*) a les rodes podrien ser un exemple clar i simple d'aquests sensors.

Els principals avantatges de l'odomètria són l'alta precisió a curtes distàncies i el seu baix cost d'implementació ja que el material utilitzat per a dur-la a terme cada vegada és més econòmic.



Figura 1 Encoder introduït en una roda.

Tanmateix també presenta greus inconvenients com la precisió de l'odometria a llargues distàncies, ja que aquesta tècnica està basada en prendre moltes mostres i l'acumulació d'errors és força important per a l'estimació del moviment.

Aquests errors es poden dividir en dos tipus: els errors sistemàtics i els errors no sistemàtics.

- Errors sistemàtics: són els errors que més influeixen en el resultat final ja que són els errors que anem acumulant constantment. Per contrapartida són els errors més fàcils de solucionar. Per exemple: el diàmetre de les rodnes mal alineades o resolució de l'*encoder*. També es poden

donar exemples amb odometria visual: una mala col·locació de la càmera que no estigui totalment paral·lela al sòl, un mal càlcul de l'altura o una mala configuració de la càmera la qual distorsioni la imatge original.

- Errors no sistemàtics: són aquells errors que en certa mesura no es poden evitar en la seva totalitat ja que són errors que apareixen inesperadament. Per exemple: desplaçament en sòls desnivellats, que les rodes patinin perquè el sòl és molt relliscós o per una elevada acceleració. Per tant el processament de les dades seria inconsistent amb el moviment real realitzat per el robot. Un exemple en odometria visual que ens servirà de cara al context del projecte és el tremolor ràpid de la imatge quan el vehicle és mou sobre un sòl rugós.

Tots aquests errors al final ens afectaran a la nostra estimació de moviment de dues maneres: error en distància (X, Y) (*Figura 2-a*) i error en orientació (σ) (*Figura 2-b*). En concret l'error en orientació és molt més perjudicial a llarg termini com es pot observar a la Figura 2.

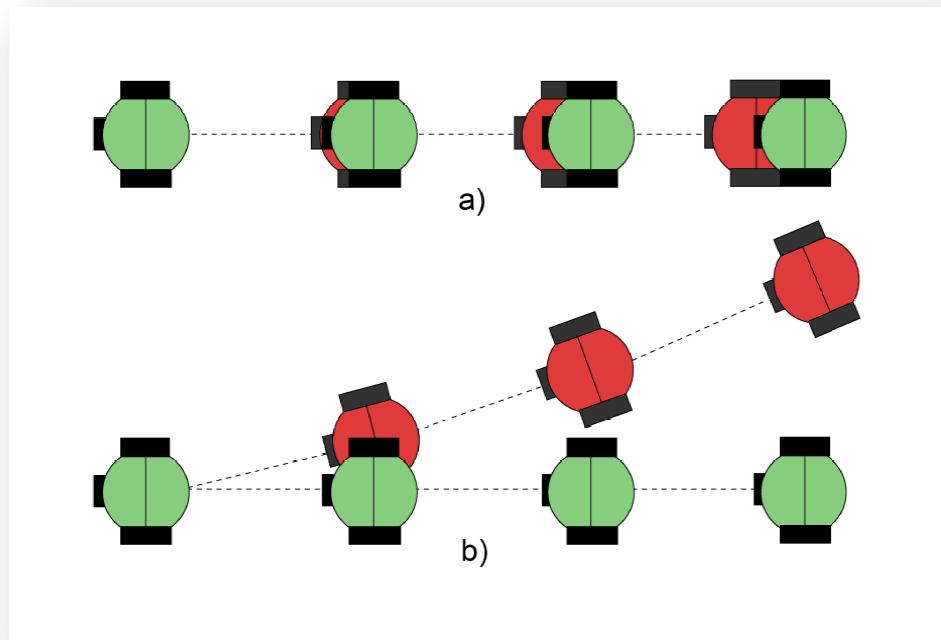


Figura 2 Tipus d'errors: (a) errors en distància i escala y (b) errors angulars en odometria. La posició real del robot està representada en verd mentre que la posició estimada es representa en vermell.

2. Odometria visual.

L'odometria Visual és una alternativa a l'odometria clàssica ja que podem evitar els errors com el relliscament de les rodes, o el no contacte de les rodes amb el sòl. Es basa en el processament d'imatges consecutives preses des del vehicle en moviment cercant punts en comú (Figura 3) entre aquestes. Aquests punts s'anomenen *features*, *keypoints* o característiques. Al llarg d'aquest document emprarem aquests termes indistintament. Gràcies a ells es pot determinar el moviment que ha patit la càmera i per tant el robot, entre dues imatges consecutives. La detecció d'aquests *features* es sol donar en característiques distintives que es situen a les imatges com per exemple cantonades.

Tanmateix també té els seus inconvenients, ja que amb càmeres convencionals es necessita de la il·luminació adequada, cosa que pot obligar a incorporar un sistema d'il·luminació al robot, augmentant el seu preu i consum energètic.

La il·luminació pròpia té més inconvenients. El focus que il·lumina va canviant de perspectiva a mesura que el robot va canviant de posició per tant lesombres projectades pels objectes canviaran. Per mor d'això és probable que sorgeixin "falsos *features*" o pèrdua d'informació i per tant errors.

L'ambient enfocat per la càmera ha d'esser un ambient predominantment estàtic de tal forma podrem obtenir referències estàtiques per calcular el desplaçament. El cas d'entorns no estàtics és objecte de recerca actualment. Es tracta d'una situació complexa i propensa a errors que cau fora de l'àmbit d'aquest projecte de final de carrera.

Els termes "robot" i "càmera" sovint els utilitzarem de manera indistinta al llarg del document per a referir-nos al mateix.

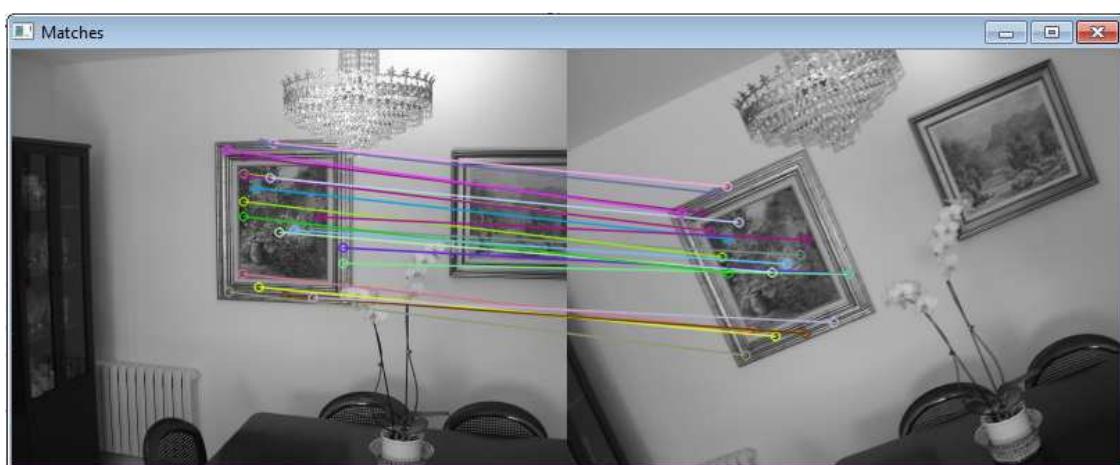


Figura 3 Exemple de visió per computador. Cada línia de color indica els punts en comú "features" que tenen les imatges.

3. Motivació.

La robòtica ha anat adquirint molta importància en el nostre desenvolupament tecnològic i, dins ella, un dels camps més importants és el de la navegació autònoma. La navegació autònoma és la capacitat d'un robot de poder desplaçar-se d'un punt de l'espai a un altre de manera autònoma.

En un robot és primordial que tingui aquesta característica implementada ja que amb la navegació autònoma poden realitzar tasques molt útils. Uns dels exemples més simples és la navegació autònoma d'un vehicle per les carreteres o l'estacionament automàtic.

La navegació autònoma permet a un robot desplaçar-se per un entorn evitant els obstacles que impedeixen arribar al seu destí. A aquest camp s'hi estan dedicant molts esforços actualment i ja comencen a sortir les primeres aplicacions comercials, com ara aspiradors autònoms, vehicles amb aparcament automàtic o el propi Google Car.

Hi ha vegades que l'odometria visual és l'única informació que tenim disponible per a dur a terme la navegació. Per exemple, quan no hi ha cap referència externa, quan un altre subsistema sensor no aporta cap informació útil o quan la informació que aporten els altres sensors és massa imprecisa per a utilitzar-la. Són exemples d'aquestes situacions les aspiradores autònomes que no utilitzen GPS, els sistemes d'aparcament automàtic on la utilització del GPS és massa imprecisa o els Google Cars dins túnels on no hi ha senyal de GPS.

Un altre punt a favor d'aquesta tecnologia és el seu baix cost, i també un baix cost d'implementació que fa més atractiva la seva utilització. Els principals causants del baix cost són els següents:

- Càmeres cada vegada més econòmiques i de major qualitat.
- CPUs econòmiques i de baix consum cada vegada més potents que permeten processar imatges.

4. Objectius.

L'objectiu del projecte és implementar un sistema de visió per computador (*Figura 4*) el qual sigui capaç d'estimar el moviment d'un mòbil a partir de les imatges capturades per la càmera acoblada al mateix. Aquestes imatges seran capturades tan paral·lelament al sòl com sigui possible, ja que voldrem saber com s'ha anat desplaçant el nostre objecte per aquest en dues dimensions (2D).

L'objectiu final consisteix en seleccionar un odòmetre visual i parametrizar-lo adequadament. Aquesta parametrització estarà enfocada a la minimització dels errors de l'estimació de moviment del odòmetre.

Per a assolir aquests objectius es duran a terme les següents tasques:

- Utilització de software d'adquisició i calibració de les imatges.

- Implementació de diversos odòmetres visuals que funcionaran *off-line* per a facilitar la seva avaluació.
- Implementació d'un software de visualització i avaluació de resultats.
- Experimentació i avaluació de les dades proporcionades per cada odòmetre.
- Selecció dels valors dels paràmetres per un odòmetre més robust.

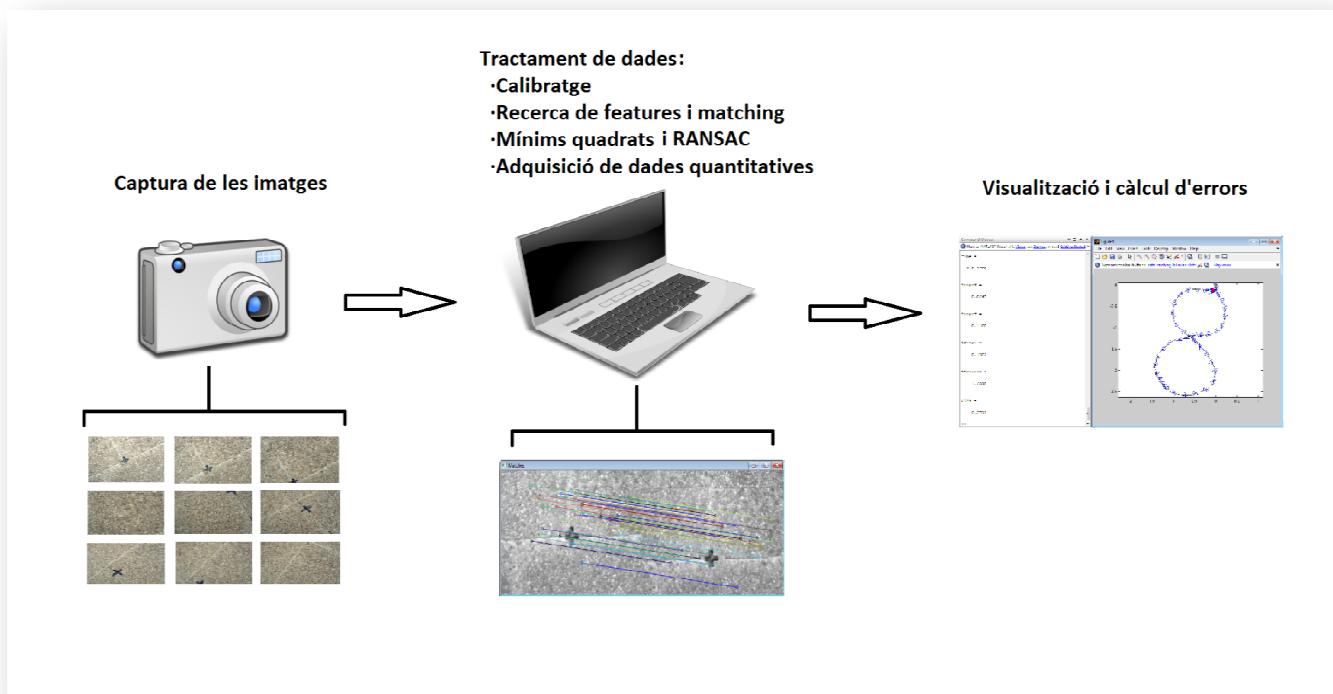


Figura 4 Esquema general del projecte.

5. Metodologia.

En el present projecte hem seguit una metodologia que ens ha permès realitzar-lo amb més eficàcia, adquirint una major comprensió de tot el procés que s'ha anat seguint:

- **Recerca Teòrica:** És un pas primordial per anar organitzant i aprenent tot el relacionat amb el que anirem utilitzant en aquest projecte.
- **Calibratge de la càmera:** Al nostre programa és bàsic obtenir la imatge mes fidel a la realitat, per tant necessitarem obtenir una imatge amb les mínimes distorsions possibles. També serà necessari per a l'obtenció de la relació píxel-metre, cosa que ens permetrà extreure informació quantitativa de les imatges.
- **Implementació de SIFT i FLANN:** Serà d'especial importància trobar els punts d'interès de les imatges anomenats *features*. Per aquesta tasca emprarem l'algorisme Scale-Invariant Feature Transform (SIFT). També haurem d'emparellar les correspondències que haurem trobat entre imatges consecutives i que

corresponen al mateix “*feature*”. Aquest procés s’anomena “*matching*” i emprarem l’algorisme anomenat Fast Library for Approximate Nearest Neighbors (FLANN).

- **Implementació de Mínims Quadrats:** Amb tot l’anterior implementat ja podrem saber quin ha estat el moviment entre dues imatges basant-nos en aquest algorisme.
- **Implementació de RANSAC:** Una vegada implementat Mínims Quadrats ja podem implementar RANSAC. És un procés molt interessant per eliminar les correspondències dolentes i poder intentar saber la millor estimació de moviment entre les dues imatges.
- **Visualització:** Necessitarem visualitzar els moviments estimats per el nostre sistema de visió i també necessitarem tractar les dades rebudes del programa. Per tant implementarem un software per visualitzar-lo.
- **Definir paràmetres a avaluar i experiments:** Ara que ja tenim tot implementat podem passar a seleccionar quins paràmetres fixarem, quins anirem variant a les proves empíriques i quins tipus d’experiments realitzarem.
- **Realització de les proves:** Haurem de realitzar les proves físicament amb les mesures necessàries i comprovant que tot estigui correctament calibrat.
- **Anàlisis dels resultats:** Quan ja tinguem les dades passades a l’ordinador i totes les proves realitzades analitzarem les respostes del programa en les diferents situacions i amb diferents configuracions. Determinarem quina configuració del nostre odòmetre és la més robusta.
- **Conclusions i propostes de millora:** Extraurem les conclusions generals del projecte i definirem una sèrie de propostes que podrien ser molt interessants a l’hora de continuar amb aquest projecte.

Capítol 2. Calibració de la càmera

La calibració de la càmera és un pas necessari per els sistemes de visió per computador i sistemes de robòtica. La calibració de la càmera s'ocupa del problema de la correspondència entre les coordenades tridimensionals (3D) de punts a l'escena i les coordenades bidimensionals (2D) dels corresponents punts en la imatge.

Les càmeres que emprarem no seran d'alta qualitat, per tant necessitarem tenir en compte els errors que constantment s'aniran acumulant.

- **Errors propis de la càmera:** Les càmeres en si produeixen un error. Quant més barata sigui la càmera que emprem més alt serà aquest error. Els errors típics de les càmeres CCD són la captació incorrecta de color i l'acoblament entre CCDs propers. A més a més, en càmeres de telèfons o similars hi ha un error addicional degut a l'escaneig seqüencial.
- **Lents barates:** La lent és un factor molt important per a minimitzar l'error, per tant una lent barata ens pot agreujar molt l'error comés per la càmera.

Normalment la calibració de la càmera es precisa per:

- **Interpretació 3D d'imatges.**
- **Reconstrucció de models del món.**
- **Interacció de robots amb el món.**

En el nostre cas, primordialment necessitarem reduir al màxim possible aquests efectes negatius de la càmera, ja que utilitzarem una gran quantitat de *frames* i aquest error es multiplica considerablement.

1. Objectius.

L'extracció d'aquesta informació té dos objectius clars:

- **Obtenció de dades quantitatives:** Aquest és un pas vital del projecte perquè extraurem la relació píxel-metre, fonamental per a la obtenció de dades quantitatives de les imatges.
- **Distorsió de la imatge:** Extraurem les dades de distorsió de la nostra càmera que després necessitarem per aplicar la correcció de la localització en píxels dels nostres *features*.

2. Model Pin-Hole.

El nostre model de calibració es basa en el model bàsic de Pin-Hole (Figura 5). En aquest model es defineix un punt principal o centre òptic a través del qual passen totes les projeccions. També es defineix un pla imatge al qual es projecten els punts de l'espai observats per la càmera. Les imatges, per tant, hi apareixen representades a la inversa. El centre òptic es troba a una distància f , o distància focal, del pla imatge. L'eix òptic Z es situa perpendicularment al pla imatge.

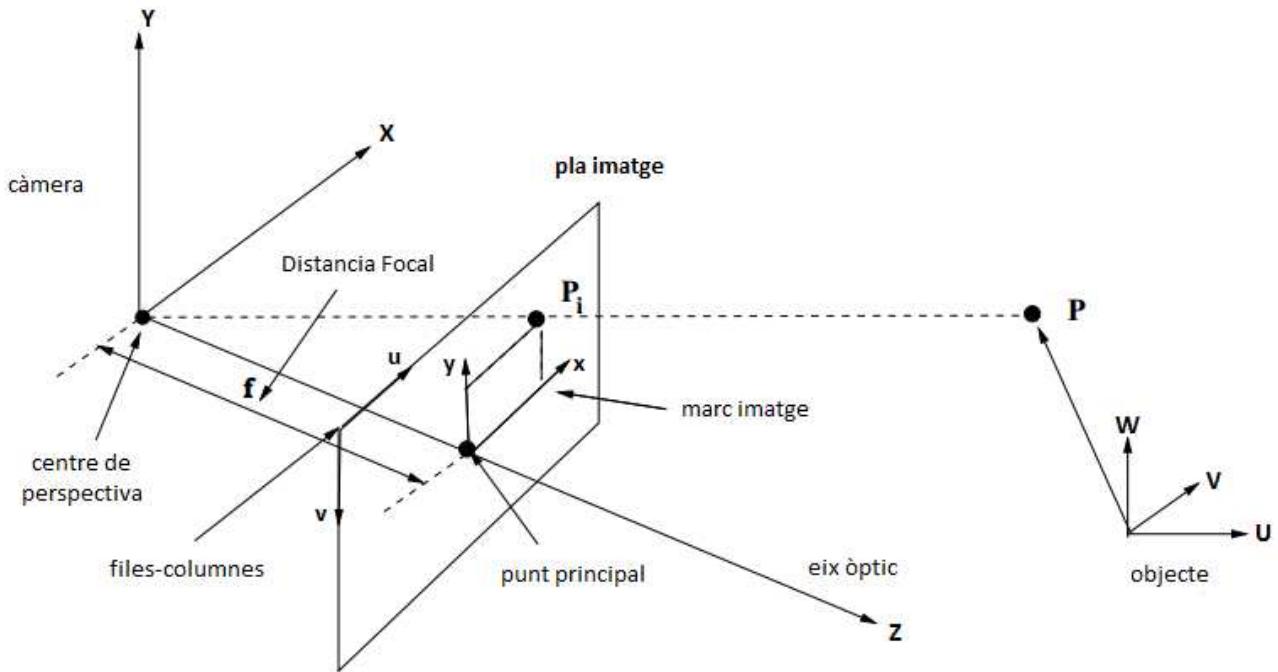


Figura 5 Representació del model Pin-Hole.

2.1 Sistema de coordenades.

El model Pin-Hole consta d'un sistema de coordenades (Figura 6) que ens permetrà obtenir les referències necessàries per a poder transformar els punts de l'espai en (3D) al pla imatge (2D). Posteriorment es podran adquirir les dades quantitatives necessàries.

Per tant necessitarem passar el sistema de coordenades de món, al sistema de coordenades de la imatge normalitzada. Entre aquests dos sistemes de coordenades afegirem un sistema intermedi anomenat sistema de coordenades de la càmera que servirà per establir una connexió entre ambdós sistemes de coordenades.

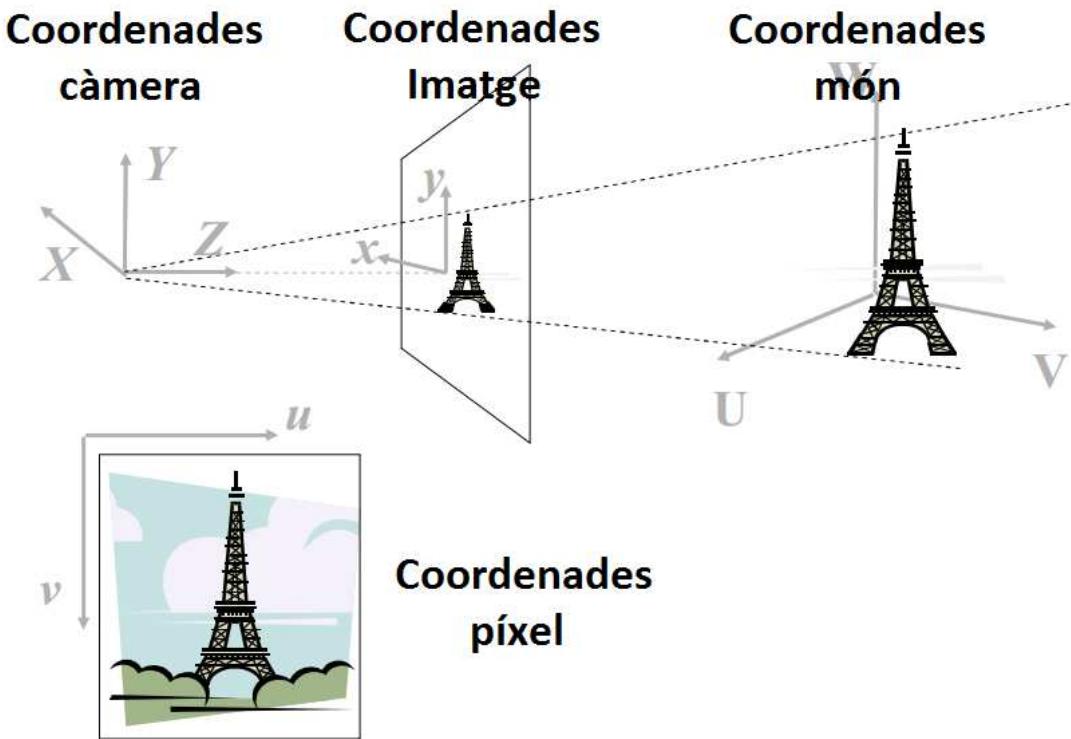


Figura 6 Sistema de coordenades emprats pel model Pin-Hole.

Per poder determinar on es localitza la projecció d'un punt (P) del sistema de coordenades mòn al sistema de coordenades de la imatge (P_i), entren en escena els sistemes de coordenada abans esmenats:

- **Sistema de coordenades mòn (W, U, V):** És el sistema de referència on es troben els punts en 3D. La col·locació d'aquest sistema és arbitraria.
- **Sistema de coordenades de la càmera (X, Y, Z):** És el sistema de coordenades que determina la posició del punt en 3D respecte de la càmera (C_c). L'origen de coordenades d'aquest sistema és el centre òptic o centre de perspectiva de la càmera.
- **Sistema de coordenades de la imatge (x, y):** És el sistema de coordenades que determina la posició del punt en 2D respecte al pla imatge (C_p). L'origen de coordenades és on talla l'eix òptic (Z_c) al pla imatge (C_p).
- **Sistema de coordenades de la imatge normalitzat (u, v):** És el sistema de coordenades que determina la posició del punt en 2D respecte al pla imatge (C_p). L'origen de coordenades en aquest cas és el cantó superior esquerra de la imatge.

2.2 Paràmetres del sistema.

El model que emprarem (model Pin-Hole) utilitza diversos paràmetres que té la càmera per projectar els punts del pla món al pla imatge. Aquests paràmetres es divideixen en dos grups, els intrínsecos i els extrínsecos. Tots aquests paràmetres són utilitzats dins matrius, matriu intrínseca i matriu extrínseca. L'equació del model de Pin-Hole es mostra a continuació.

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \quad (1)$$

Els paràmetres intrínsecos són els que fan referència a les característiques internes de la càmera, geometria interna i l'òptica de la càmera. Aquells paràmetres s'utilitzen per fer les referències entre el sistema de coordenades de la càmera i el sistema de coordenades de la imatge. Entre aquests paràmetres trobarem:

- **Distància focal (Fx, Fy):** és la distància que hi ha entre el centre òptic i el centre de la imatge del pla imatge en mil·límetres.
- **Centre de la imatge (Cx, Cy):** és el punt on l'eix òptic travessa el pla imatge, aquest punt està expressat en píxels.
- **Factor d'escala (1/Sx, 1/Sy):** indica la proporció existent entre l'objecte en el món real i l'objecte projectat en el pla imatge.

Els paràmetres extrínsecos són els que descriuen i relacionen els sistemes de coordenades món i els sistemes de coordenades de la càmera:

- **Rotació (r):** submatriu esquerra 3x3 de l'equació (1) que ens relaciona la direcció entre el sistema càmera i el sistema món, rotant així entre els seus tres eixos.
- **Translació (t):** submatriu dreta 3x1 de l'equació (1) que ens relaciona la distància que hi ha entre el sistema de coordenades de la càmera i el sistema de coordenades món.

2.3 Càlcul de projeccions.

Com abans hem esmenat, el model pin-hole explica com la càmera capture les imatges. Amb una sèrie de transforacions dels punts que la càmera capture del món real amb l'objectiu de passar-los al pla imatge, és a dir, del sistema de coordenades món al sistema de coordenades imatge. El model pin-hole ho explica en tres senzilles passes.

- Sistema de coordenades món – càmera.
- Sistema de coordenades càmera – imatge.
- Sistema de coordenades imatge – imatge normalitzada.

Sistema de coordenades mòn → sistema de coordenades càmera.

Aquest pas s'encarrega de passar el sistema de coordenades mòn al sistema de coordenades de la càmera. Per a realitzar-lo utilitza els paràmetres extrínsecos, de rotació i translació. Es realitzarà la rotació en els tres eixos del sistema de coordenades i la translació donada per la submatriu 3x1 de l'equació (1).

La fórmula simple es podria exposar d'aquesta manera:

$$P_c = r(P_w - t) \quad (2)$$

On P_c és el punt en les coordenades càmera, r és la matriu de rotació amb la qual rotarem els nostres punts al voltant dels tres eixos, P_w és el punt en les coordenades mòn i t és la matriu de translació.

També es pot expressar amb matrius, en coordenades homogènies. Utilitzant aquí els paràmetres extrínsecos de rotació r i translació t .

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 1 & -c_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} U \\ V \\ W \\ 1 \end{pmatrix} \quad (3)$$

Sistema de coordenades càmera → sistema de coordenades imatge.

Aquest pas ens permet passar les coordenades càmera 3D (X, Y, Z) a les coordenades imatge 2D (x, y) (Figura 7). En aquest apartat ja utilitzem els paràmetres intrínsecos de la càmera per a realitzar la projecció.

Farem ús de la trigonometria per a calcular la projecció a partir del sistema de coordenades de la càmera.

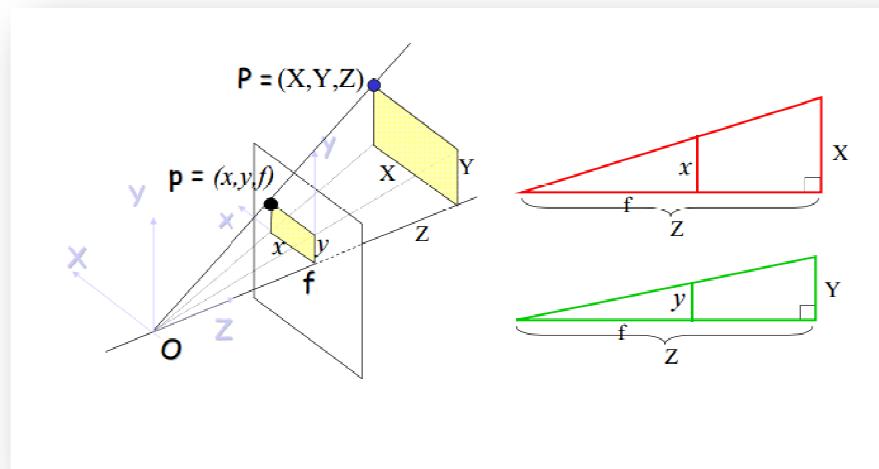


Figura 7 Perspectiva de coordenades càmera a pla imatge.

D'aquí podem extreure les dues equacions principals, si suposem que són ideals. Avon f és la distància focal.

$$x = f \frac{X}{Z} \quad (4)$$

$$y = f \frac{Y}{Z} \quad (5)$$

Ara podem representar amb matrius aquestes equacions, per això necessitarem introduir les coordinades homogènies. Representarem un punt en 2D (x, y) mitjançant un punt 3D (x', y', z') afegint una tercera coordenada "fictícia". Donat (x', y', z') podem recuperar el punt 2D (x, y) mitjançant aquesta conversió.

$$x = \frac{x'}{z'} \quad y = \frac{y'}{z'} \quad (6)$$

La representació amb matrius serà de la següent manera:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (7)$$

Sistema de coordenades imatge → sistema de coordenades imatge normalitzada.

Aquest pas descriu la transformació de coordenades entre les coordenades de la imatge projectada i la matriu de píxels. Aquí entren en joc tots els paràmetres intrínsecos de la càmera: f, O i S .

$$u = \frac{1}{S_x} f_x \frac{X}{Z} + c_x \quad v = \frac{1}{S_y} f_y \frac{Y}{Z} + c_y \quad (8)$$

On C_x, C_y són les coordenades del centre de la imatge respecte del cantó superior esquerra d'aquesta. També hem de tenir en compte que hi ha diferents factors d'escala en x i y que no necessàriament han d'esser quadrats.

Podem representar-ho amb matrius d'aquesta manera.

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} \frac{f_x}{S_x} & 0 & c_x & 0 \\ 0 & \frac{f_y}{S_y} & c_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (9)$$

Per tant podem deduir u i v , amb les antigues equacions, ja que hem utilitzat el mateix procediment d'afegir una coordenada "fictícia". D'aquesta manera demostrem d'on prové el model de Pin-Hole.

2.4 Efectes de distorsió.

Els càlculs anterior que hem explicat són suficients si suposem que la càmera és ideal. Per tant si volem analitzar una càmera real amb les seves distorsions haurem d'afegir-les. Hi ha dos tipus de distorsions:

- Distorsió radial.
- Distorsió tangencial.

Aquestes distorsions són creades per l'objectiu de la càmera (Figura 8) que distorsionen la realitat. Les imatges capturades tindran un error sistemàtic que s'haurà de minimitzar.



Figura 8 Imatge presa amb un objectiu fish-eye.

Partim de l'equació (6) quan volíem passar el sistema de coordenades càmera al sistema de coordenades imatge. A continuació afegirem els coeficients de distorsió que quedaran representades en unes noves variables anomenades x'' i y'' .

$$x'' = x'(1 + k_1 r^2 + k_2 r4) + 2p_1 x'y' + p_2(r^2 + 2x'^2) \quad (10)$$

$$y'' = y'(1 + k_1 r^2 + k_2 r4) + p_1(r^2 + 2y'^2) + 2p_2 x'y' \quad (11)$$

Els paràmetres $K1$ i $K2$ són els coeficients de distorsió radial i $k3$ i $k4$ són els coeficients de distorsió tangencials. On r es pot definir com.

$$r^2 = x'^2 + y'^2 \quad (12)$$

Per tant podem definir els punts de la imatge normalitzats u i v havent afegit els coeficients de distorsió de la imatge de la següent manera, on C_x i C_y són el centre de la imatge.

$$u = f_x \cdot x'' + C_x \quad (13)$$

$$v = f_y \cdot y'' + C_y \quad (14)$$

Per un anàlisi amb més profunditat de la distorsió amb el model Pin-Hole es pot revisar el contingut a la bibliografia [1].

3. Calibració amb MATLAB.

La calibració és el procés d'obtenir els paràmetres de la càmera (típicament intrínsecs) a partir d'imatges de patrons de calibració agafades amb la càmera a calibrar. La càmera que utilitzarem és una càmera SONY cyber shot DSC-P10 [2]. En el present projecte hem emprat un programa que ens permet amb senzills passos realitzar l'obtenció de les dades necessàries per a la calibració de la càmera utilitzada per a les posteriors gravacions. Aquest programa s'anomena “toolbox_calib” [3]. Una vegada hem encès el programa ens apareix un panell inicial (Figura 9) que ens permet elegir diferents opcions a seleccionar. A continuació anirem desglossant punt a punt.



Figura 9 Menú principal del programa “toolbox_calib”.

A nivell pràctic, el procés de calibració sempre necessita imatges o patrons de referència captats amb la càmera a calibrar. La majoria del software de calibració empra quadricules homogènies com a patró de referència.

El toolbox de Matlab empra un patró tipus taulell d'escacs. Aquest taulell contará amb diferents cantons que seran els punts de referència de cada imatge realitzada. Necessitarem una sèrie d'imatges per que el programa pugui resoldre el sistema d'equacions necessari. Aquestes imatges es realitzaran prenent fotografies del taulell d'escacs en diferents punts de vista.

En la realització de les imatges per fer-ho lo més similar possible a quan estiguem realitzant les gravacions de les proves, s'ha realitzat un vídeo del taulell d'escacs on es veu des d'un parell de perspectives i distàncies.

El programa ens demana que configurem MATLAB per a que el directori es trobi situat en l'arxiu on tinguem les imatges del taulell guardades. Haurem de guardar les imatges realitzades amb un sufix anomenat “xx” i seguit del numero d'imatge, per exemple “xx1, xx2, xx3, xx4...” en el directori seleccionat.

Una vegada realitzat aquesta tasca, aproximadament unes 8 imatges està bé, podrem seleccionar al menú l'opció “Images names” on ens demanarà el sufix dels noms de les imatges i el tipus d'imatge. El mateix programa reconeixerà una a una les imatges guardades al directori (Figura 10).

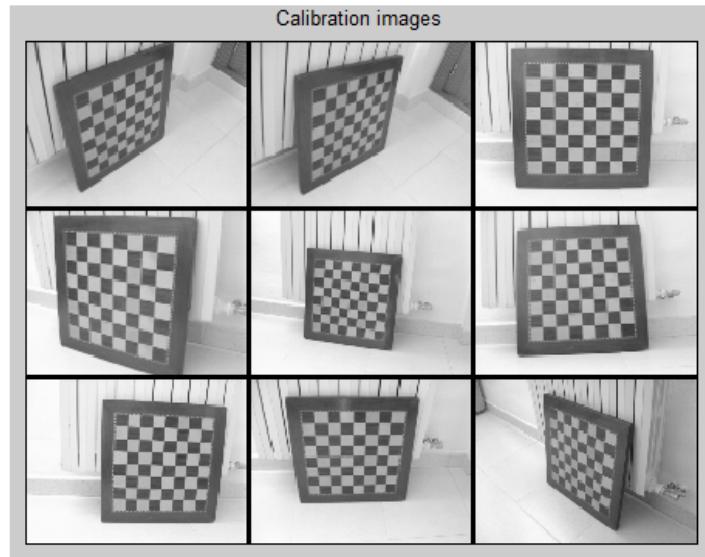


Figura 10 Reconeixement d'imatges de “toolbox_calib”.

Una vegada el programa ens ha reconegut les imatges haurem de passar a la selecció dels cantons de les imatges. Per això haurem de seleccionar el boto del menú principal del programa “Extract grid corners”. A continuació passarem a realitzar la cerca dels cantons del taulell d'escacs.

En primer cas ens demanarà quantes interseccions hi haurà en el nostre taulell tant en l'eix Y com en l'eix X. En el nostre cas tenim un taulell de 8x8 requadres, per tant tenim 7 interseccions a l'eix X i 7 interseccions a l'eix Y. A continuació ens demana l'amplada en “mm” dels requadres, en el nostre cas són de 400x400 mm.

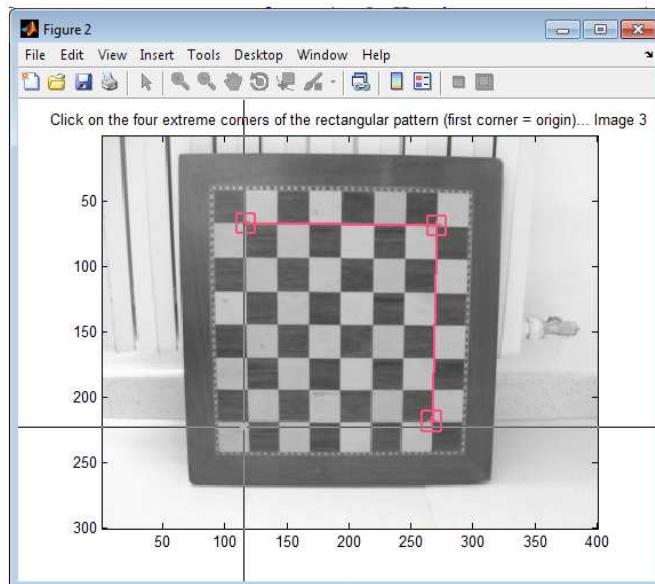


Figura 11 Selecció dels cantons del taulell d'escacs.

La detecció dels cantons d'aquest programa és semi manual (Figura 11). El programa requereix la selecció, per part de l'usuari, de la regió de la imatge a analitzar. Fet això, ja de

manera automàtica, es detecta la ubicació dels cantons. El resultat final per a cada una de les imatges analitzades és semblant al que es mostra a la Figura 12.

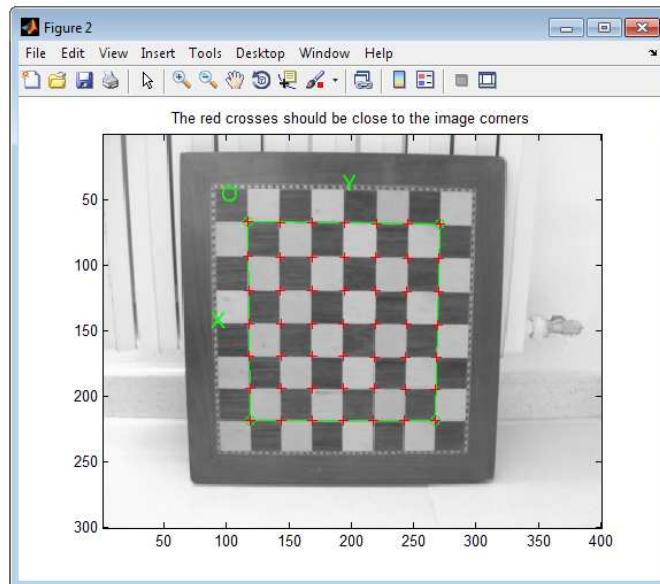


Figura 12 Cantons seleccionats del taulell d'escacs.

Una vegada tenim tots els punts d'interès seleccionats (Figura 12), ja podrem obtenir els nostres paràmetres de la càmera necessaris per a la calibració de la càmera. Seleccionarem al menú principal el botó “Calibration”. A continuació ens apareixerà per pantalla totes les variables necessàries.

3.1 Resultats de calibració.

A continuació es mostrerà el resultat numèric dels paràmetres que utilitzarem per a la calibració de la càmera en el present projecte facilitades pel programa “toolbox_calib”.

Distància Focal	$f = [449.38312 \ 448.48709]$ píxels
Punt principal	$c = [208.20762 \ 147.07710]$ píxels
Distorsió	$kc = [-0.19034 \ 0.16094 \ -0.00122 \ -0.00178 \ 0.00000]$

Taula 1 Paràmetres de la càmera.

On f equival a distància focal de la càmera en píxels, c és el punt principal en píxels pues les imatges que s'han emprat i s'empraran durant to el projecte són de 400x300 píxels. Kc correspon al coeficient de distorsió, que s'emprarà posteriorment per a eliminar la distorsió.

A continuació un exemple d'una imatge a la qual eliminem la distorsió utilitzant els paràmetres realitzat per MATLAB (Figura 13). Com es pot veure, a la imatge original s'observa una certa distorsió radial que provoca que les línies rectes del tauler d'escacs apareguin corbades. Després de l'eliminació de la distorsió, les línies es veuen rectes, tal i com és a la realitat.

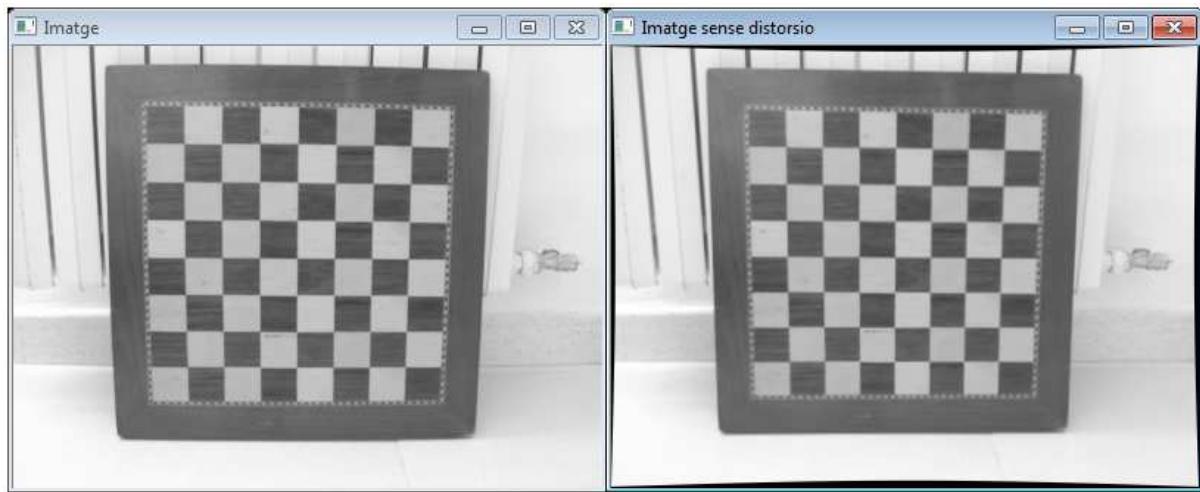


Figura 13 Comparació entre imatge distorsionada (esquerra) amb una imatge sense distorsió (dreta).

Capítol 3. Recerca de característiques

En aquest apartat explicarem una part fonamental del present projecte. La localització dels *features* de les imatges serà l'eix al voltant del qual girarà tot el projecte. Explicarem pas a pas com aconseguirem localitzar aquests *features* i com determinarem quins coincideixen entre dues imatges consecutives i quins haurem d'eliminar. Primerament s'explicarà breument la part teòrica relacionada amb els detectors - descriptors i *matching*. Posteriorment explicarem com hem aplicat aquests processos al nostre programa.

1. Objectius.

El procés de recerca de característiques o *features* consta d'un parell de processos. Aquests processos són necessaris per a realitzar les correctes assignacions de *features* iguals. Consta dels següents passos:

- **Localització de *features* (detectors):** Aquest apartat s'encarrega de localitzar els punts importants o "interessants" de les imatges els quals ens serviran per relacionar una part de la imatge anterior amb la següent.
- **Caracterització de *features* (descriptors):** Els descriptors ens permetran esbrinar les característiques que tenen els *features* per tal de poder diferenciar-los i relacionar-los directament amb *features* d'altres imatges.
- **Assignació de *features* (*matching*):** L'apartat de *matching* serà capaç de relacionar mitjançant les matrius que proporciona el descriptor els *features* que coincideixen entre dues imatges consecutives i descartar, en conseqüència, els *features* no desitjats.

1.1 Features.

En la visió artificial i processament de imatges el concepte de *features* o característica és una petita porció de la imatge que té unes propietats singulars respecte la resta de la imatge. Aquesta petita porció ens permetrà distingir-la i tenir punts de referència fàcilment detectables. Aquests *features* són el resultat d'operar dins la imatge a nivell de píxel mitjançant "veïnats".

Aquestes operacions són anomenades "*feature extractor*" o "*feature detector*". Detecten estructures específiques simples com per exemple punts, cantons o estructures més complexes com objectes. Dins el document els *features* els anomenarem també com "característiques" o "keypoints".

L'aplicació a la que voldrem aplicar la recerca de *features* (Figura 14) depèn molt del tipus de *features* que voldrem trobar. En el nostre cas voldrem trobar *features* que siguin molt

robusts en quant a la rotació i punt de vista de la càmera i també als possibles renous, també necessitarem una certa repetitivitat.

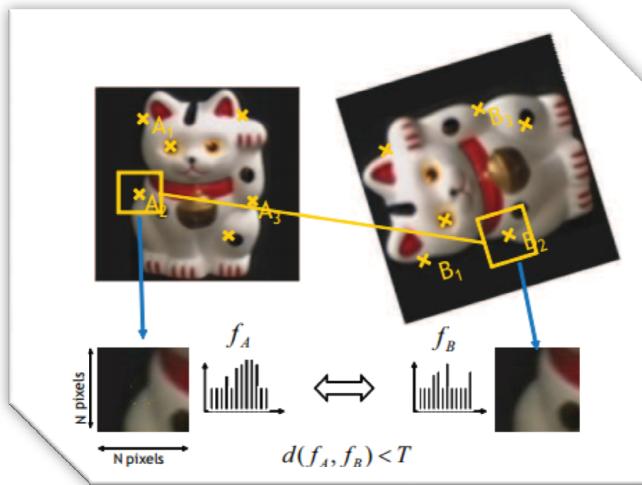


Figura 14 Exemple de feature.

1.1.1 Propietats dels features.

Com abans hem esmenat, els *features* necessiten una sèrie de propietats necessàries per a la localització i distinció de dits *features*, aquestes són les més generals:

- **Repetitivitat:** Són *features* que són capaços d'esser trobats de maneres totalment independents a cada imatge operada, es a dir hem d'esser capaços de detectar els mateixos punts encara que canviïn les condicions de visió.
- **Invariant a translacions, rotacions i escales:** Els *features* detectats han d'esser capaços d'esser trobats inclús si el punt de vista de l'observador canvia tant en translació, rotació i escala. El detector de *features* pot seleccionar una posició, orientació i escala canòniques i així poder emparellar-les posteriorment.
- **Invariant en presencia de renous o taques:** El detector serà capaç de detectar el mateix punt en dues imatges inclús si en alguna de les imatges aquell punt esta borrós o està afectat per renou.
- **Distintivitat:** Les regions trobades per el detector hauran de tenir estructures fàcilment distingibles entre elles.
- **Localitat:** Les regions petites són menys sensitives a les deformacions pels canvis de visió, on l'oclusió pot esser un problema.
- **Quantitat:** Hi ha d'haver suficients punts per representar la imatge, ja que en molts de casos es necessiten un nombre mínim determinat de punts com per realitzar

l'aplicació. Com per exemple la detecció d'objectes o en el nostre cas per a realitzar l'odometria.

- **Eficiència:** La detecció dels *features* ha d'esser eficient temporalment si pretenem per exemple realitzar un anàlisis en temps real o en temps de vídeo.

1.2 Detectors.

El primer pas per a l'extracció de característiques locals és trobar un conjunt de punts clau distintius que poden ser localitzats de forma fiable dins condicions variables de imatge, canvis de punts de vista, i en presencia de renou.

En particular, el procediment d'extracció ha de donar els mateixos punts de característiques si la imatge d'entrada sofreix una translació o una rotació. No tots els punts de la imatge podran complir aquestes condicions. Per exemple si considerem un punt situat en una regió uniforme no podrem determinar el seu moviment exacte, ja que no podem distingir aquest punt del seus veïnats. De la mateixa manera si tenim en compte un punt en una línia recta, només podrem mesurar el seu moviment, si el moviment es realitza en perpendicular a la línia.

Això motiva a centrar-nos en un determinat subconjunt de punts, és a dir, aquells que presenten canvis en diverses direccions.

Els detectors són els encarregats de cercar els punts “interessants” en les imatges. Aquest pas és molt important ja que posteriorment es treballarà damunt aquests punts seleccionats com a potencialment útils.

Detector Harris.

A continuació posarem un exemple de detector de cantonades com per exemple el detector Harris per a la comprensió de com funciona un detector de *features*.

Per tal de detectar punts clau de la imatge es necessiten utilitzar estratègies que marginin característiques de les imatges que normalment són poc útils i localitzar les que ens interessen. En el cas del detector Harris serien les cantonades.

Suposant que el detector opera en finestres petites dins la imatge, el propòsit és trobar punts de la imatge tal que el mínim canvi causat pel moviment de la finestra en qualsevol direcció sigui gran. Aquesta condició ens servirà per a distingir una cantonada de, per exemple, una simple línia (Figura 15).

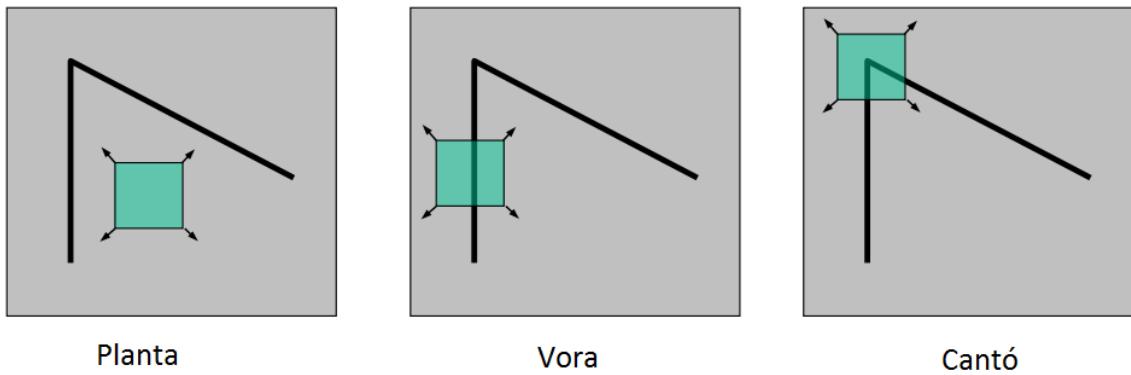


Figura 15 Recerca de Features “interessants”.

- Les regions tipus planta no experimenten cap canvi en cap moviment que realitzi la finestra seleccionada.
- La regió tipus vora obtindrà canvis en la finestra. Excepte si el moviment es realitza en la direcció de la vora, el qual no experimentarem cap canvi.
- La regió de tipus cantó en canvi en qualsevol moviment que realitzi la finestra s'obtindrà un canvi substancial.

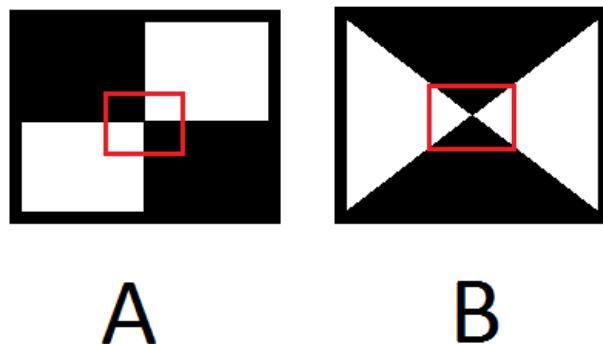


Figura 16 Finestra d'operació de Harris a un cantó ben orientat (A) i un mal orientat (B).

Si suposem que volem trobar una cantó com el de la Figura 16-A. El detector Harris opera en una petita finestra com es mostra a la Figura 16 amb un marc vermell, on el cantó està dins. Per detectar el cantó calcularem els gradients que hi ha al marc vermell tant en l'eix X com a l'eix Y anomenats $\sum(I_x)^2$ i $\sum(I_y)^2$ respectivament.

$$\begin{aligned}\sum(I_x)^2 &\rightarrow \text{GRAN} \\ \sum(I_y)^2 &\rightarrow \text{GRAN}\end{aligned}$$

Per a que es pugui considerar cantó, els gradients tant en l'eix X com a l'eix Y han d'esser grans. Si només ho és un dels dos eixos, es tracta d'una línia.

El detector Harris també troba cantons que no estan en la posició òptima com la de la Figura 16-B. Ja que el gradient horitzontal no serà tan gran com el gradient vertical. Per aquests casos el detector Harris rotarà el sistema de coordenades per a que l'orientació del cantó sigui la correcta i tornar al cas anterior. Aquesta rotació es realitzarà mitjançant la següent matriu.

$$\begin{pmatrix} \sum(I_x)^2 & \sum I_x I_y \\ \sum I_x I_y & \sum(I_y)^2 \end{pmatrix}$$

On $\sum I_x I_y$ són el producte dels components dels gradients. Aquesta matriu ens donarà dos valors, si aquests dos valors són grans es tractarà d'un cantó.

Per obtenir aquests gradients es pot realitzar utilitzant la següent relació:

$$E(u, v) = \sum_{(x,y) \in W} [I(x+u, y+v) - I(x, y)]^2 \quad (15)$$

Mitjançant el moviment de la finestra W en els seus eixos (x,y) compararem cada píxel abans i després del moviment utilitzant la suma de les diferencies al quadrat. Anomenat Sum of Squared Differences (SSD) "error" E(u,v). Per a que un punt sigui bo, l'error E(u,v) de la finestra ha d'esser gran.

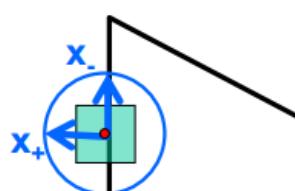


Figura 17 Moviment de finestra per a localitzar cantons.

Fitxant-nos en la Figura 17 es mou el centre de la finestra verda en qualsevol direcció del cercle blau cercant el gradient o error comés per la finestra al ser traslladada.

Com podem observar el detector Harris és un detector simple de característiques ja que el seu propòsit és cercar cantonades únicament. Nosaltres haurem d'utilitzar un detector un poc més sofisticat que tinguin en compte altres aspectes de la imatge els quals siguin invariables en rotació, il·luminació, punt de vista i sobretot la repetitivitat. Per tant no ens bastarà simplement en la localització de cantons ja que tenim especial interès per la minimització dels possibles errors comesos per qualsevol dels components.

1.3 Descriptors.

Una vegada ja hem localitzat els *features*, en el nostre cas de les dues imatges a comparar, haurem de saber identificar i diferenciar els *features*. D'aquesta tasca s'encarreguen els descriptors. Donant solució al problema de no saber quin *feature* es correspon amb el mateix de l'altre imatge o eliminar falsos positius. Els falsos positius són *features* que creiem que són el mateix però en realitat no ho són.

En el present projecte per realitzar aquesta tasca utilitzarem el descriptor SIFT. És un descriptor invariant als canvis de rotació i escala, i bastant robust en quant a l'efecte dels renous, zones borroses i canvis d'iluminació. Aquestes característiques ens seran de gran utilitat a l'hora de minimitzar els errors, sobretot els canvis de rotació i dels efectes de renou i zones borroses.

2. SIFT.

Scale Invariant Feature Transform (SIFT) [4] fou introduït originalment per Lowe com una combinació de detector de regions d'interès locals Difference of Gaussians (DoG) i un descriptor de *features* corresponent [Low99, Low04b]. No obstant, ambdós components han estat utilitzats per separat. En particular una sèrie d'estudis ha confirmat que el descriptor SIFT és adequat per a la combinació amb altres detector de regions com per exemple Harris, Harris-Laplacian, Hessian-Laplace, LoGm etc, que generalment assoleixen un bon rendiment.

El detector SIFT té com a objectiu assolir la robustesa a les variacions d'iluminació i petits canvis de posició mitjançant la codificació de la informació de la imatge en un conjunt localitzat d'histogrames d'orientació de gradient.

El detector-descriptor SIFT original i que nosaltres utilitzarem en el present projecte consta de diferents apartats. Els dos apartats principals com hem esmentat abans en la introducció són SIFT detector i SIFT descriptor.

- **Detector SIFT.**

- **Espai escalar Gaussià:** La creació d'un espai escalar és fonamental en SIFT, ja que necessitarem trobar *features* a diferents escales.
- **Diferència de Gaussianes (DoG):** La diferència de gaussianes ens permetrà localitzar els màxims i mínims entre els espais escalar Gaussians anteriorment creats.
- **Localització de les característiques:** Establiment de les característiques bones i eliminació de candidats dolents.

- **Descriptor SIFT.**

- **Assignació d'orientació als *keypoints*:** Assignació d'una orientació canònica als *keypoints* anteriorment localitzats.
- **Creació de descriptors:** Es crearan els descriptors necessaris per a la diferenciació dels *keypoints*.

2.1 Detector de features.

Aquest apartat s'encarrega de la selecció dels *features* o característiques, que també són anomenades “*keypoints*”. Els *features* de SIFT són una regió circular de la imatge amb escala i orientació. Es descriuen amb un marc geomètric de quatre paràmetres:

- El centre del *Keypoint* amb les coordenades X e Y.
- L'escala (el radi de la regió).
- L'orientació (angle expressat en radians).

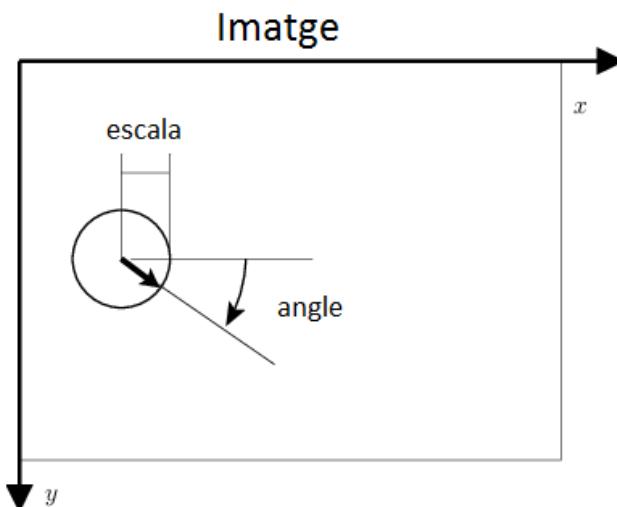


Figura 17 Feature tipus SIFT.

El detector SIFT utilitza com a punts clau estructures de la imatge que són similars a “taques”. Mitjançant la recerca d'aquestes taques en múltiples escales i posicions el detector SIFT és invariant, o més exactament covariant, de translacions, rotacions i escalat de imatges.

2.1.1 Espai escalar Gaussià.

En l'apartat anterior hem xerrat que el detector Harris és invariant a les rotacions però no a l'escala. Per exemple si una cantonada és molt més gran que la finestra que s'utilitza per detectar *features* (Figura 18) l'algorisme no serà capaç de detectar-la. En aquest cas Harris no és invariant a escales mentre que SIFT sí.

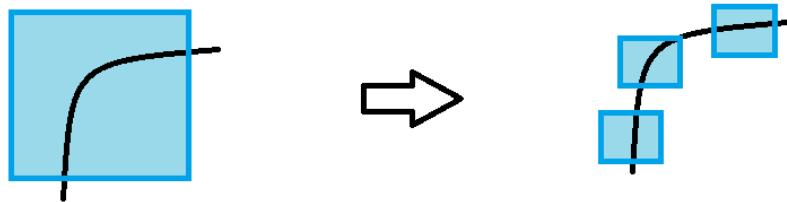


Figura 18 Finestra per detecció de features. Esquerra: la finestra és suficientment gran com per a detectar una cantonada. Dreta: les finestres no són lo suficient grans i detecten únicament línies.

Per a detectar *features* a diferents escales no es pot utilitzar la mateixa finestra com es demostra a la Figura 18. Per a poder cercar *features* a diferents escales SIFT crea un espai anomenat “espai d’escala Gaussià”. Aquest espai d’escala està compost per un grup de imatges obtingudes mitjançant els suavitzats progressius de les imatges d’entrada, que és molt similar a la reducció gradual de la resolució de la imatge. Convencionalment, el nivell de suavitat s’anomena escala de la imatge.

Aquests filtrats es realitzen mitjançant el filtre de Gauss, creant així una estructura piramidal de imatges on cada nivell les imatges són suavitzades i reduïdes en grandària. Per tant es podran cercar *features* a escales molt més grans amb la imatge més reduïda.

Per a calcular l’espai SIFT utilitza la funció continua coneguda com scale-space $L(x,y,\sigma)$. Per obtenir aquesta funció $L(x,y,\sigma)$ a partir de la imatge original $I(x,y)$ s'utilitza la funció Gaussiana següent.

$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y) \quad (16)$$

On l’operador $*$ indica la convolució entre la imatge original i la gaussiana G .

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}} \quad (17)$$

Per a calcular tot l’espai $L(x,y,\sigma)$ s’ha de construir una piràmide gaussiana (Figura 19) convolucionant amb diferents filtres $G(x,y,\sigma)$ variant el paràmetre σ . Hi ha dos termes que són importants definir per a comprendre la piràmide.

- **Octava:** Conjunt de imatges de l’espai gaussià amb el mateix tamany que es diferencia pel filtrat σ amb el qual han estat obtingudes.
- **Escala:** Conjunt de imatges de l’espai gaussià filtrades amb el mateix paràmetre σ però amb diferent tamany.

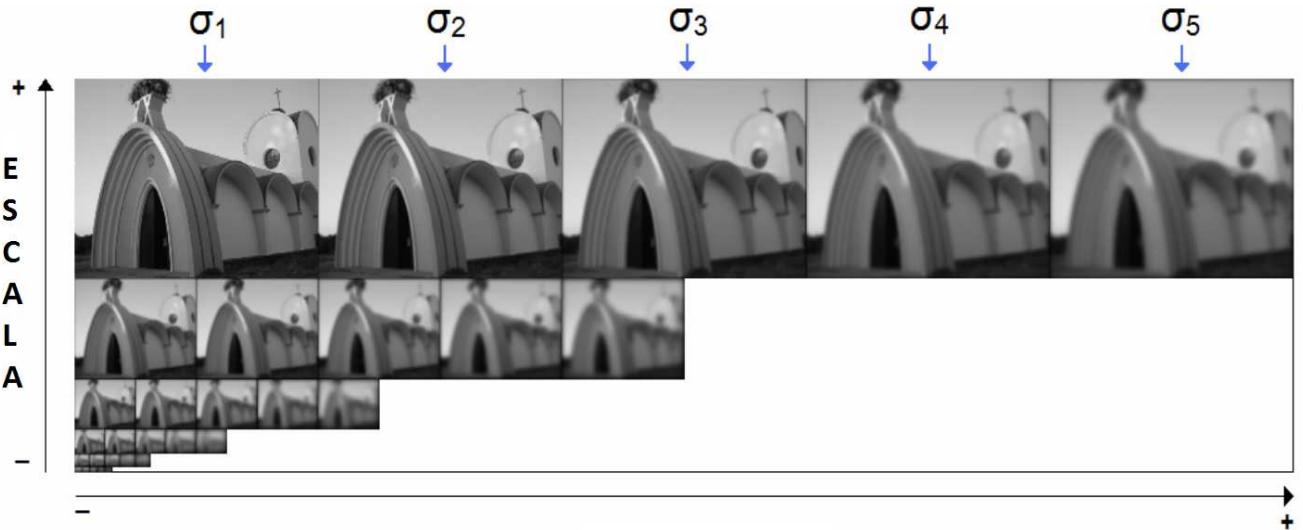


Figura 19 Piràmide Gaussiana $L(x, y, \sigma)$ composta per 6 escales i 5 octaves.

L'augment de l'escala mitjançant l'octava significa duplicar la mida de la mascara per a suavitzar, aquest efecte és mes o manco l'equivalent a la meitat de la resolució de la imatge.

2.1.2 Diferència de Gaussianes (DoG).

Una vegada l'espai escalar està creat SIFT realitza la diferencia de Gaussianes (Figura 20) que ens permetrà localitzar màxims i mínims entre les imatges operades.

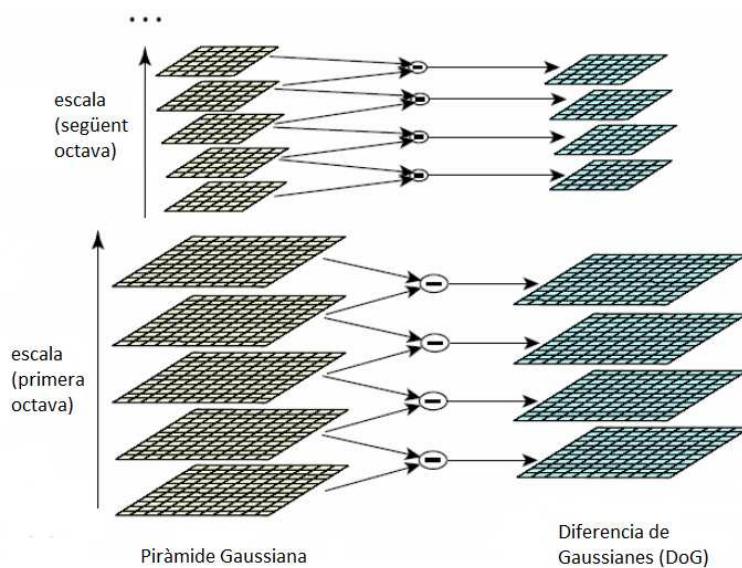


Figura 20 Espai Gaussià (esquerra), Diferencia de Gaussianes (dreta).

Per a detectar *features* estables en l'espai escalar SIFT no s'utilitza la funció L esmenada anteriorment. S'utilitza una funció que deriva d'ella i que s'anomena Diferència de Gaussianes $D(x, y, \sigma)$. Aquest càlcul és molt menys costós que el càlcul de la funció L ja que només es calcula restant les imatges veïnes de les mateixes octaves de la piràmide gaussiana.

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \quad (18)$$

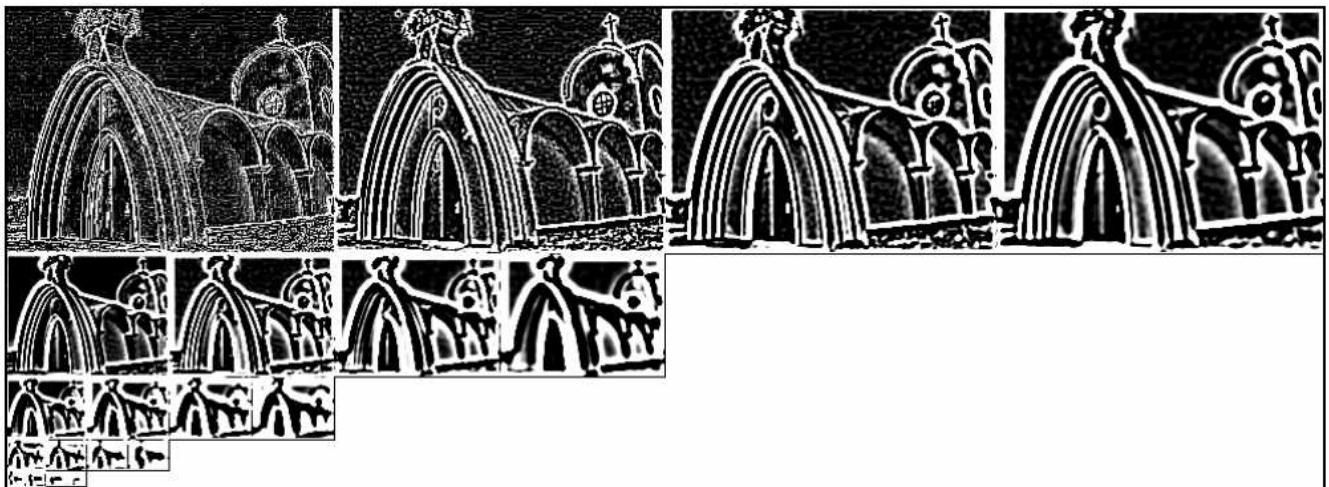


Figura 21 Piràmide de Diferència de Gaussianes (DoG).

Una vegada ja s'han realitzat les operacions necessàries i s'ha creat la piràmide de diferència de Gaussianes (Figura 21) hem de realitzar la recerca de màxims i mínims. Cada punt es comparat amb els seus 8 veïnats de la mateixa imatge i comparat també amb els nou veïnats de la imatge superior i inferior. Un punt només serà seleccionat com a possible *keypoint* si és major o menor que els seus 26 veïns (Figura 22). Per cada màxim o mínim trobat, l'output serà la localització i l'escala. És a dir, per a cada punt de interès trobat es guardarà a quina escala i a quina octava de la piràmide pertany i a quina posició [fila, columna] està dins la imatge corresponent.

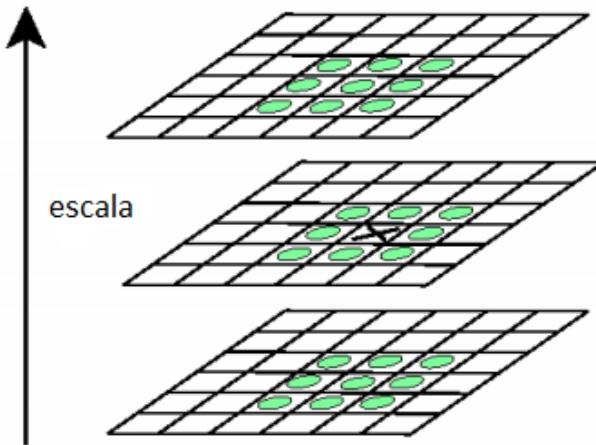


Figura 22 Recerca de màxima en l'espai de diferència de Gaussines. La creu es el píxel comparat amb els seus 26 veïns (verds).

2.1.3 Localització de característiques.

Una vegada realitzat màxims de DoG, tindrem un ampli llistat de punts o possibles *keypoints* trobats amb una precisió com a màxim de píxel. Com major sigui l'escala on el punt ha estat trobat correspondrà a un major nombre de píxels de la imatge principal. També s'ha de tenir en compte que molts d'aquests punts són falsos o erronis i en conseqüència han d'esser eliminats.

Totes les dades anteriorment registrades per a cada *keypoint* (escala, octava i posició) ens permetrà poder descartar *keypoints* potencialment dolents. Hi ha dos criteris per a descartar:

- Baix contrast.
- Localitzats al llarg de vores.

2.2 Descriptor de features.

Una vegada s'han obtingut i filtrat els *features*, SIFT procedeix a la part de descripció dels mateixos. Aquest apartat és fonamental per a la diferenciació entre *features* i sobretot per a poder assignar *features* parells mitjançant *matching* entre imatges contigües. Aquest apartat s'encarrega de dues tasques principals:

- **Orientació dels keypoints:** S'encarrega d'assignar una orientació als *features* per així evitar errors en la comparació posterior de *features* que varien en la orientació.
- **Creació de descriptors:** És la creació de la distintivitat necessària per al posterior *matching* que es realitzarà.

2.2.1 Assignació d'orientació als keypoints.

En aquest apartat es calcularan les orientacions de cada keypoint. Una vegada les tinguem, es podran construir els descriptors ja que aquests estaran referenciats a les seves respectives orientacions i, per tant, aconseguir la invariància en rotació.

SIFT defineix una regió de 16x16 píxels al voltant del punt on volem calcular l'orientació i a cada un dels punts es calcula el gradient. Aquest ve determinat pel seu mòdul $m(x,y)$ i inclinació $\Theta(x,y)$.

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \quad (19)$$

$$\Theta(x,y) = \tan^{-1} \left(\frac{(L(x,y+1) - L(x,y-1))}{(L(x+1,y) - L(x-1,y))} \right) \quad (20)$$

La imatge utilitzada per fer els càlculs anteriors serà la imatge de la piràmide gaussiana $L(x,y,\sigma)$ on es va detectar el *keypoint* que s'està analitzant. Després de calcular els mòduls i les inclinacions de cada píxel s'agruparà tota la informació en un histograma, un per a cada keypoint. De tal forma que cada histograma d'orientació estarà format de 36 bins per cobrir el rang de 360º (Figura 23).

A mesura que s'afegeix al histograma cada orientació $\Theta(x_1,Y_1)$ de la regió 16x16, l'esmenat valor es pondera amb el seu mòdul $m(x_1,Y_1)$ i per una finestra circular gaussiana. Aquestes dues ponderacions es realitzen pels següents motius:

- Donar major importància a les orientacions amb mòduls elevats, que per tant són més importants.
- Donar major pes als punts propers al keypoint, és a dir, els punts centrals a la finestra.

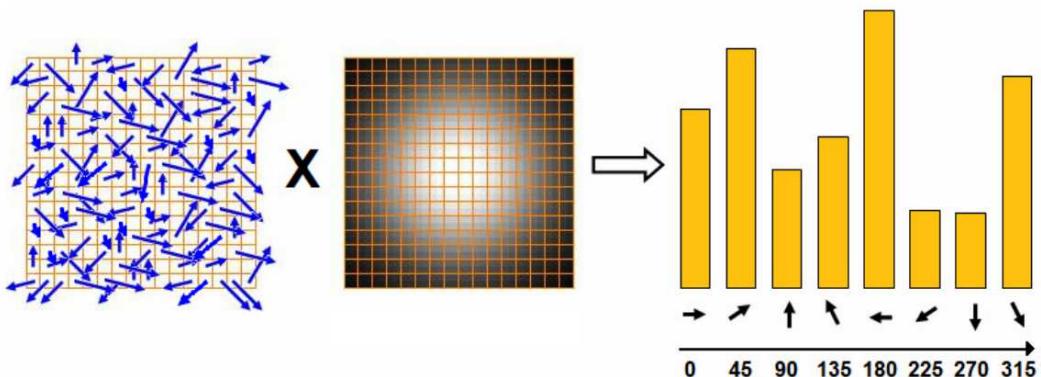


Figura 23 Esquerra: regió 16x16 amb els gradients. Centre: finestra circular gaussiana. Dreta: histograma de keypoints.

Els pics més alts de cada histograma són les direccions dominants dels gradients locals i per tant l'orientació final del keypoint. A vegades SIFT no es quedará només amb el pic més alt, també es cercaran pics secundaris que tinguin una altura major al 80% del pic principal. Si

no existeix cap pic secundari amb aquesta característica només es quedarà amb el pic major.

Fins aquest punt ja tenim guardats la següent informació dels punts més importants: localització, octava, escala i orientacions principals. Amb tota aquesta informació ja es poden crear els descriptors.

2.2.2 Creació de descriptors.

El següent pas és, utilitzant tota la informació que ja tenim, obtenir un descriptor per a cada zona de interès. Aquest apartat ens aportarà més robustesa a les possibles variacions de il·luminació i canvis de vista.

El procés s'inicia de la regió 16×16 abans calculada a l'apartat 2.2.1 ja multiplicades per la finestra circular gaussiana. Aquesta matriu es divideix en regions de 4×4 píxels amb l'objectiu de resumir tota la informació en petits histogrames de només 8 bins, és a dir , 8 orientacions. Abans de realitzar aquest procés cada gradient de la finestra 16×16 es rota tants graus com especifica l'orientació principal del *keypoint* calculada en l'apparat anterior i així serà independent a la inclinació de la imatge.

Per a cada *keypoint* abans teníem un histograma amb 36 possibles orientacions, ara passarem a tenir 16 histogrames amb 8 possibles orientacions. Per a evitar grans canvis entre regions, cada subregió serà filtrada un altre cop amb una finestra circular gaussiana de tamany 4×4 (Figura 24).

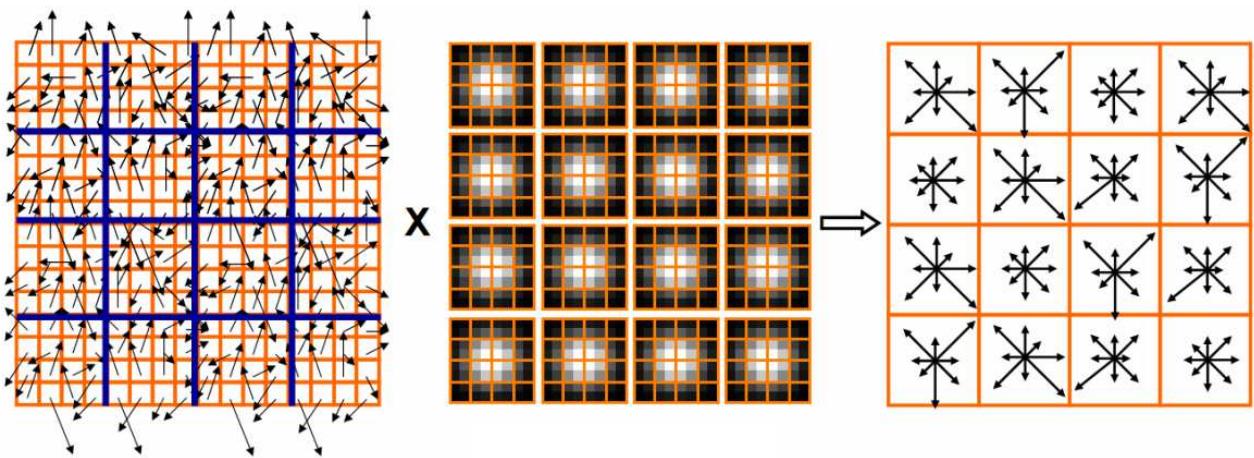


Figura 24 Esquerra: subregions definides. Centre: finestres gaussianes per a cada subregió. Dreta: histograma de les subregions.

El nostre descriptor és invariant a la lluminositat, això es degut a que els gradients estan calculats mitjançant diferencies entre veïns. Per tant el fet de sumar una constant de llum a tota la imatge no influirà en el resultat final.

Una vegada tenim tots aquests paràmetres calculats podem donar per acabat el procés de creació de descriptors.

3. Matching.

Es l'algorisme que s'encarrega de la recerca i establiment de correspondències entre les dues imatges que voldrem comparar. Aquest algorisme s'engega una vegada s'han realitzat tots els passos previs de detector i descriptor de les dues imatges, tenint així un llistat de *features* amb els corresponents descriptors. L'algorisme de *matching* farà ús d'aquest llistat per a poder realitzar les correspondències.

En el nostre cas hem de suposar que tenim dues imatges I1 i I2, que són les imatges on voldrem realitzar les correspondències. La imatge I2 serà una versió transformada en translació i rotació. En el present projecte la diferència d'iluminació i de canvi d'escala en principi no serà de gran importància. L'objectiu és trobar els mateixos *features* en les dues imatges independentment d'aquests canvis. Per això SIFT s'ha encarregat anteriorment amb la detecció i descripció dels *features* per a poder emparellar-los entre les dues imatges. El *matcher* tractarà, en la mesura del possible, realitzar l'òptima comparació d'aquests punts ja que normalment solen aparèixer falsos positius (molt perjudicials alhora de minimitzar errors). Aquests falsos positius són assignacions del *matching* errònies i per tant el programa tindrà informació falsa que pot ser perjudicial per a l'odometria final.

Per a realitzar aquesta tasca es mesura la diferencia de la distància entre els descriptors dels *features*. Ja que en el cas del descriptor SIFT es crea un espai N-dimensional on els descriptors es consideren punts i en calculem la distància.

$$d(u, v) = \left(\sum_i (u_i - v_i)^2 \right)^{1/2} \quad (21)$$

On $u = (u_1, u_2, \dots)$, $v = (v_1, v_2, \dots)$ són les coordenades dins l'espai N-dimensional dels *features*.

Hi ha moltes possibles estratègies a seguir amb la distància calculada entre aquests vectors.

- Es pot retornar tots els vectors dels *features* amb una distància menor a un valor llindar.
- Veïnat més proper: es retorna el vector amb la distància més petita de cada *feature* comparat.
- Relació de distància entre veïnats propers (Nearest neighbor distance ratio). On d_1 i d_2 són les distàncies entre el més proper i el segon més proper al feature. Si NNDR és petit, el veïnat més proper és un *matching* correcte ja que hi ha una gran distintivitat entre ambdós *features* (més proper i segon més proper).

$$NNDR = \frac{d_1}{d_2} \quad (22)$$

En el present projecte per a realitzar els *matchings* utilitzarem Fast Library for Approximate Nearest Neighbor (FLANN)[5]. Ja que és un algorisme bastant estable i amb un llistat molt llarg de *features* és més ràpid que altres algorismes com per exemple BruteForce [6], que compara un per un tots els *features* amb la resta de *features* per poder establir

correspondències. Tant FLANN com BruteForce són part de la funcionalitat estàndard d'OpenCV. S'ha pogut veure un exemple de *matching* combinant SIFT amb FLANN a la Figura 3.

3.1 Parametrització.

Com hem esmentat abans, els algorismes quan realitzen *matching* també produeixen un error significatiu. Aquest error sorgeix amb l'aparició de falsos positius, punts que el programa creu que són veraders però que realment no ho són i poden afectar greument a l'odometria. Per poder minimitzar aquests falsos positius aplicarem un llindar al llistat ja realitzat de *matchigns* per FLANN anomenat (*numMin*).

FLANN ens retorna un vector. Cada element del vector referencia els dos *features* relacionats emprant la posició que ocupen dins del vector de *features*. També retorna la relació de proximitat de descriptor entre *features* emparellats, que anomenarem "Distància".

Nº feature 1	184	34	505	94	492	333	430
Nº feature 2	395	210	470	15	118	584	221
Distance	132,346	408,587	363,671	384,409	55,353	80,777	339,025

Taula 2 Exemple de vector de sortida de l'algorisme FLANN.

Com més petita sigui aquesta relació de distància significa que hi ha una relació de similitud entre *features* més alta i a l'inversa, com més gran sigui aquesta relació més possibilitats tenim de que es tractin de falsos positius. Per tant haurem de fixar un llindar que ens faci de filtre i evitar el màxim possible aquest tipus d'error.

Per establir aquest llindar utilitzarem la relació (23). Prendrem la distància més petita de distància del vector de dades, és a dir, el *matching* entre *features* més consistent $Distance_{min}$. El llindar vendrà donat pel producte d'aquesta distància i del nostre paràmetre *numMin*.

$$Valor_{llindar} = Distance_{min} * numMin \quad (23)$$

Si es vol un llindar més exhaustiu, (*numMin*) haurà d'esser més petit. D'aquesta manera els únics *matchings* que seran escollits com a bons seran els que tenen la distància més baixa, més propera al, a priori, *matching* més consistent. El llindar s'aplicarà de la següent manera.

$$Distance_{feature} < Valor_{llindar} \quad (24)$$

4. Algorisme: detecció i matching.

Aquest apartat de la implementació del programa en C++ es troba a un pas entremig del programa però cal explicar-lo abans que altres apartats per a la seva correcta comprensió. Més específicament s'explicarà l'apartat 2 del esquema general de l'algorisme (Figura 25).

Més endavant explicarem el pas 3 en l'apartat de càlcul de moviment i les parts 1 i 4 les explicarem en l'apartat de paràmetres d'entrada i sortida del nostre programa.

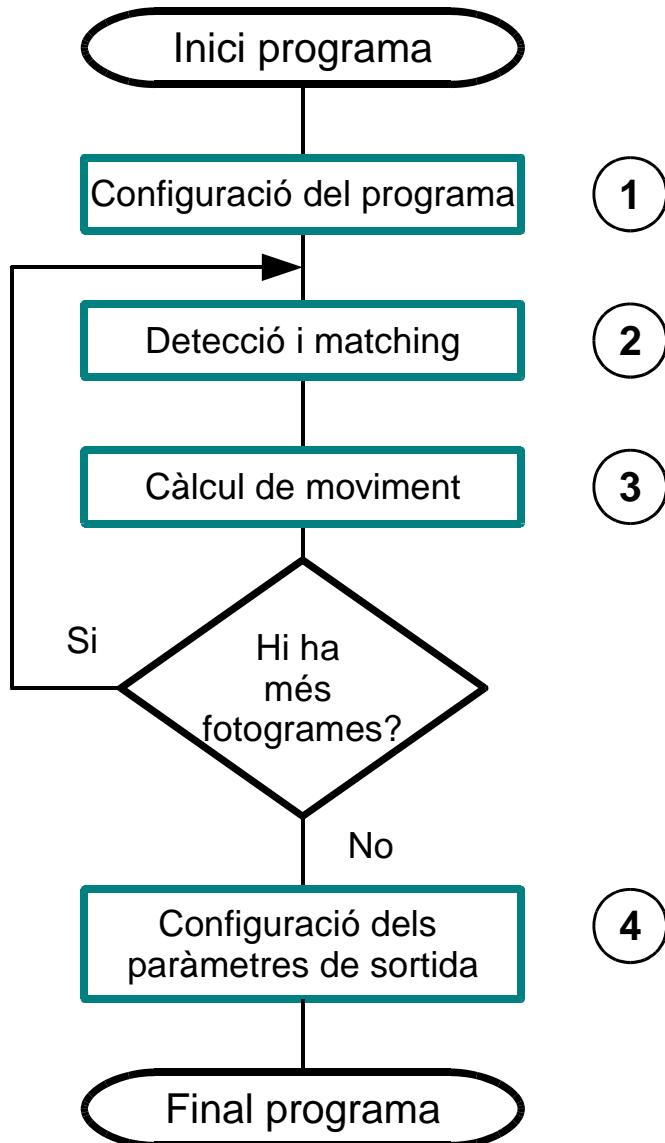


Figura 25 Esquema general del algorisme.

Per explicar aquest pas s'ha dividit en dos subapartats (Figura 26) que descriuran tota la tasca realitzada en quant a detecció, descripció, *matching* i altres afegits. Com per exemple la lectura de les imatges o la impressió per pantalla dels *features* amb el *matching* de les dues imatges.

- **Part 1.** Comprèn la lectura de les imatges i l'anàlisi de les mateixes (SIFT) per a obtenir tota la informació necessària ja que posteriorment es vol establir les

correspondències. També es realitzarà l'eliminació de la distorsió de les localitzacions dels *keypoints*.

- **Part 2.** Una vegada tenim tota la informació de les imatges amb la distorsió dels punts eliminada, es realitzarà el *matching* de les imatges i la seva impressió per pantalla. Una vegada s'ha realitzat el *matching*, organitzarem els *keypoints* en els vectors que ens convindrà per a la seva posterior utilització. Tot seguit realitzarem una important transformació dels punts: canvi de centre de coordenades i passar els píxels a metres. La ubicació dels *keypoints* amb les mesures adequades queda enllestida per a poder calcular el moviment en el pas posterior.

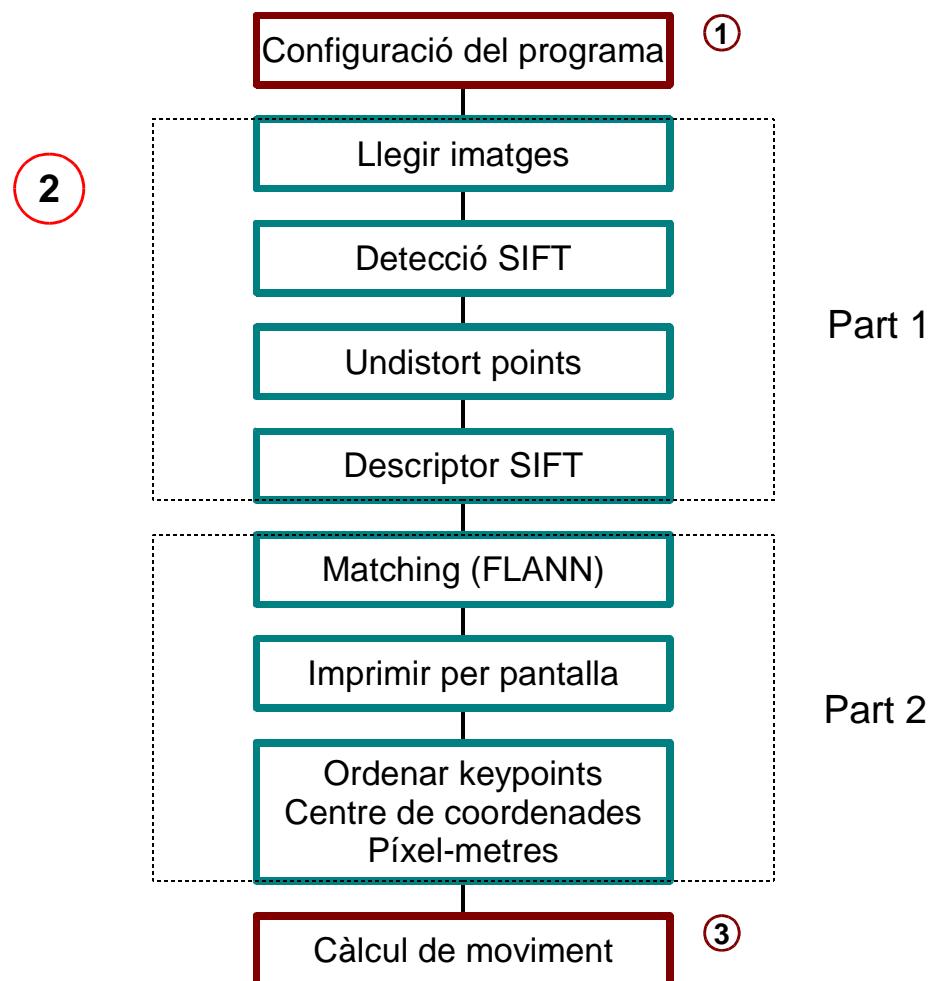


Figura 26 Esquema general de detecció i matching.

4.1 Apartat 1: Anàlisi de les imatges.

Aquest apartat correspon principalment a la lectura i l'anàlisi de les imatges computades (Figura 27). Ja que sempre treballarem amb dues imatges que són consecutives necessitarem un cert ordre: guardar informació necessària per a optimitzar del programa, fer còpies segures de les matrius de les imatges, etc. Es poden distingir tres principals accions dins aquest apartat: lectura de les imatges, detecció/descriptor SIFT i eliminació dels efectes de la distorsió en les coordenades dels *keypoints*.

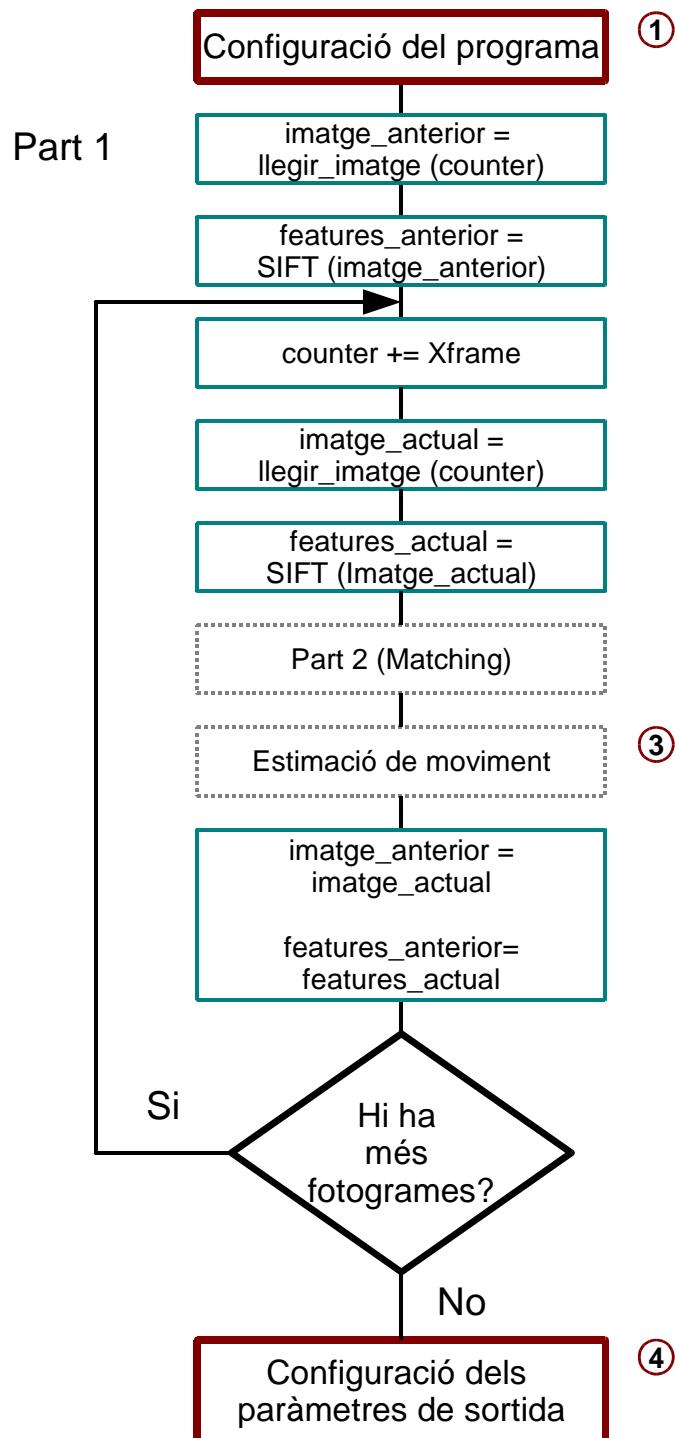


Figura 27 Estructura del algoritme de detecció i matching: apartat 1.

Abans d'entrar al bucle de lectura consecutiva d'imatges el programa realitzarà primer la lectura i anàlisi de la primera imatge (counter =1). Guardant la matriu de imatge a imatge_anterior. Posteriorment es realitza l'anàlisi de la imatge. S'estreuen els *features* amb els seus corresponents descriptors guardant les dades a feature_anterior. També s'elimina la distorsió dels punts com abans hem explicat.

Una vegada hem entrat al bucle el comptador incrementa el valor assignat a "Xframe". La qual és una constant general del programa on ens referim a cada quants *frames* volem llegir. Aquest paràmetre s'explicarà detalladament més endavant.

La matriu de la imatge llegida dins el bucle quedarà guardada com a imatge_actual. Es pot dir que imatge_actual sempre serà la imatge de *frame* actual, mentre que imatge_anterior serà la imatge de *frame* anterior.

Una vegada ja hem realitzat la lectura i anàlisi de les dues imatges passarem a l'apartat 2 (el quadre que està en gris i puntejat, ja que s'explicarà més endavant amb més detall) on es cercaran les correspondències. Amb les dues imatges guardades a les variables imprimirem per pantalla les dues imatges amb els *features* corresponents als *matching* realitzat. Aquesta ordre d'imprimir per pantalla es realitza al segon apartat.

A continuació es realitzarà l'estimació de moviment entre imatges. Aquest apartat també es troba en gris a l'esquema, per tant l'explicarem amb més detall més endavant.

Després d'haver realitzat l'estimació de moviment, passarem tota la informació de la imatge actual (matriu de imatge, posició de *features* i descriptors) al *frame* anterior. Aquest pas ens estalvia tornar a analitzar la imatge de nou i només ho farem una vegada per imatge.

4.1.1 Lectura de les imatges.

La lectura de les imatges és el pas inicial del tractament de dades ja que necessitarem computar la informació inicial de les imatges obtingudes per l'usuari. La llibreria OpenCV ens proporciona funcionalitats que permeten llegir imatges tant en color com en escala de grisos. En el nostre cas ens centrem en imatges en escala de grisos. Aquestes seran transformades per OpenCV a un format matricial anomenat Mat.

Ja que són imatges consecutives i necessitarem haver guardat les imatges ordenades, hem nomenat totes les imatges amb un nom "I" seguit pel seu número en quant a posició seqüencial del vídeo. Aquest numero sempre serà de 4 xifres per exemple "I0001, I0002, I0003...". A la Figura 28 es mostra un exemple.

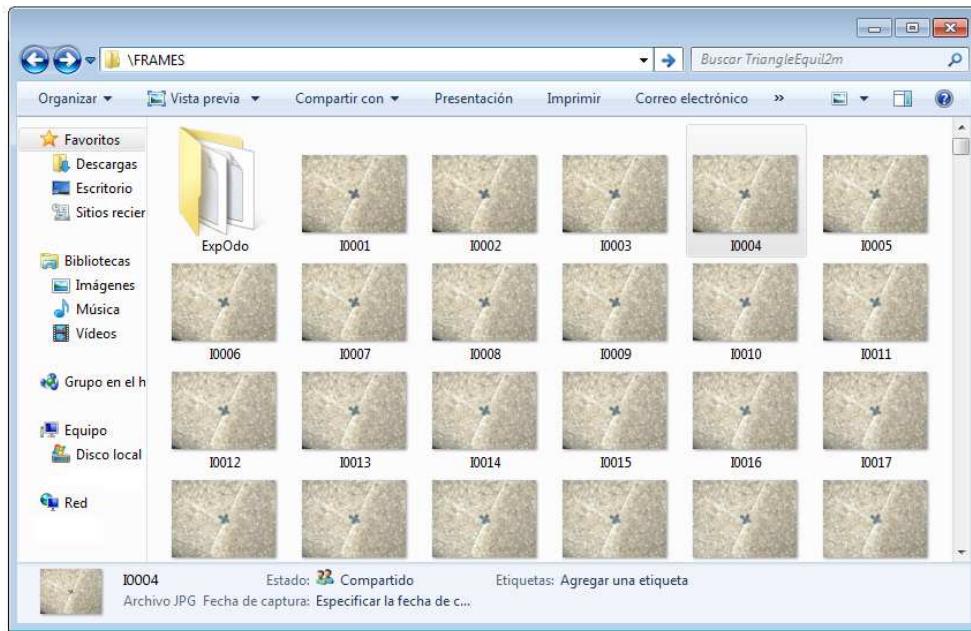


Figura 28 Carpeta, localització de les imatges.

Aquesta numeració facilita la utilització d'un comptador que ens permetrà anar llegint de manera seqüencial les imatges. Aquest comptador l'hem anomenat “counter” com es pot veure a la Figura 27. També podrem configurar cada quants *frames* volem llegir. Aquest serà un aspecte important a avaluar. Per a la correcta lectura de les imatges necessitarem que a l'arxiu d'entrada del programa se li indiqui la ubicació de la carpeta on hem guardat les imatges.

Les imatges guardades han d'esser del tipus JPG, mentre que la mida és a elecció de l'usuari, sempre i quant totes les imatges tenguin les mateixes dimensions. Cal destacar que unes dimensions excessivament grans poden alentir considerablement l'execució del programa.

4.1.2 Detecció i descriptor SIFT.

Les llibreries de OpenCV incorporen el detector i el descriptor SIFT que detecten els *features* amb els corresponents descriptors. L'entrada de la funció és la matriu de valor de píxel de la imatge en escala de grisos. La sortida serà un vector de dades amb les coordenades dels *features* i una matriu amb els descriptors dels *features* trobats.

Tota aquesta informació ens servirà per al posterior *matching*. Els dos vectors i les dues matrius de les imatges a comparar seran l'entrada a la funció per a realitzar el *matching*.

4.1.3 Eliminació de la distorsió dels punts.

Les càmeres distorsionen les imatges capturades depenent de la seva qualitat i del seu objectiu. Aquesta distorsió és més alta o més baixa com hem esmentat i explicat al capítol 2

de calibració de la càmera. OpenCV ens ofereix diverses opcions per a poder tractar aquest error sistemàtic.

Per a eliminar la distorsió de les imatges tenim dues opcions:

- **Opció A:** Capturar imatge, eliminar la distorsió de la imatge i extreure els *features*. Una vegada llegida la imatge, OpenCV ens permet modificar-la mitjançant unes funcions que eliminaran la distorsió de la imatge. Una vegada eliminada la distorsió iniciarem la recerca de *features*.
- **Opció B:** Capturar imatge, extreure *features* i eliminar la distorsió als punts. A diferència de l'opció A, s'extrauran els *features* directament de la imatge amb la corresponent distorsió. Una vegada tenim les localitzacions dels *features*, OpenCV ens permet corregir la distorsió als punts que hem extret anteriorment.

La via que utilitzarem serà l'opció B. L'opció A té grans inconvenients, ja que s'està modificant la imatge original. La modificació de la imatge es realitza mitjançant tècniques que tenen el risc de fer perdre informació sobre punts clau o fins i tot d'originar falsos feature. Aquesta opció es pot realitzar per una visualització de la imatge sense distorsió com es pot veure a la Figura 13 però no és recomanable el seu posterior anàlisi.

OpenCV compta amb una funció per eliminar la distorsió de les coordenades dels punts. Aquesta funció té com a entrada la matriu de la càmera (distància focal i centre de coordenades) i la matriu de distorsió de la càmera que hem calculat abans al capítol anterior (Taula 1) als resultats de la calibració de la càmera.

Aquests paràmetres es trobaran inicialment guardats a l'arxiu d'entrada del programa. La matriu de la càmera, matriu de distorsió i altres paràmetres d'entrada del programa depenen de la càmera que utilitzem o de la situació de la gravació del vídeo. Per tant s'hauran de canviar els paràmetres d'entrada si varia la càmera que utilitzem o la condició de la gravació.

S'eliminarà la distorsió dels punts cada vegada que una imatge sigui analitzada per SIFT i els *features* siguin detectats. Aplicarem la funció a les coordenades (X,Y) individualment de cada *feature* del vector de dades proporcionat per SIFT.

A la sortida de la funció les coordenades obtingudes estaran sense distorsió i normalitzades, és a dir, que no depenen dels paràmetres de la càmera. En el nostre cas necessitarem les coordenades en píxels, ja que posteriorment haurem d'imprimir els *features* emparellats quan el *matching* estigui realitzat. Per tant passarem un altre cop les coordenades normalitzades a píxels.

4.2 Apartat 2: Matching i configuració.

En aquest apartat es realitza el *matching* entre les dues imatges analitzades a l'apartat 1 i es configuren les dades obtingudes per posteriorment poder dur a terme els càlculs d'odomètria correctament.

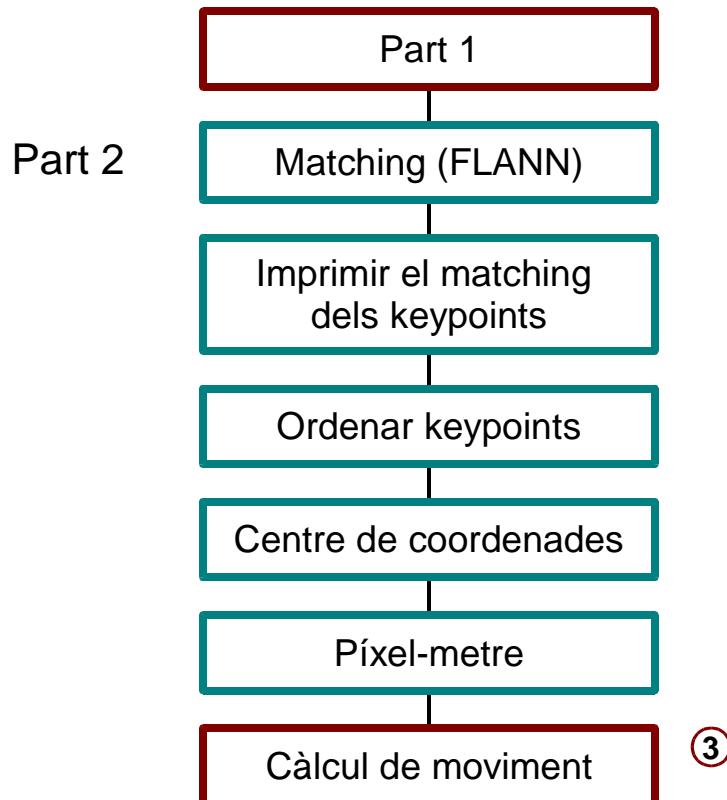


Figura 29 Estructura del algoritme de detecció i matching: apartat 2.

Aquest apartat és totalment lineal com podem veure a la Figura 29 ja que es realitzen les tasques per ordre. Una vegada s'ha realitzat el *matching* amb les dades proporcionades per l'apartat 1 podem passar a mostrar per pantalla les imatges, guardades a les Img_1 i Img_2, amb el seu corresponent *matching*.

Amb totes les dades obtingudes i filtrades mitjançant “numMin”, necessitarem ordenar els punts obtinguts en dos vectors. Aquest pas només implica guardar ordenadament en parelles dins dos vectors diferents tots els *matchings* com ens indica el vector dels *matchings* bons “m_good_matches” explicat a l'apartat 3.1. Una vegada ordenades les coordenades, passarem a la conversió de les coordenades per al seu posterior us.

4.2.1 Parametrització.

El nostre algorisme principal llegeix les imatges del vídeo que capture el moviment del mòbil el qual volem estimar el seu moviment. Aquesta lectura és consecutiva i per això el nostre programa fa servir un comptador el qual ens permet accedir fàcilment a les imatges prèviament emmagatzemades amb el sistema de numeració descrit anteriorment.

No necessàriament haurem de llegir les imatges *frame a frame*. Al nostre programa podrem especificar cada quants *frames* volem llegir les imatges. Indicant el nombre de *frames* al paràmetre “Xframe” modificarem aquesta condició de lectura.

En ocasions la lectura entre *frames* més separats pot ser beneficiós per al resultat final de l’odometria. Per exemple en un ambient on hi ha molt de soroll com la rapida vibració de la càmera en un sòl rugós o obstacles com pedres que desenfoquin la càmera en breus espais de temps. És recomanable definir un paràmetre Xframe elevat.

4.2.2 Origen de coordenades.

De forma predeterminada a OpenCV l’origen de coordenades (0,0) d’una imatge està situat al cantó superior esquerra. Per tant cada vegada que el punt de la imatge s’allunyi de l’origen tant en X com en Y els valors de ambdós augmenta.

Les coordenades dels *features* també estan referenciades a aquest origen de coordenades. Això és un problema ja que quan haguem de realitzar el càlcul de moviment aquest centre de coordenades no ens servirà.

El moviment consta de 3 variables (moviment en X, moviment en Y i rotació). Si l’origen de coordenades es troba al cantó superior esquerra o al centre de la imatge no afecta al moviment tant en X com en Y, però sí afecta significativament quan volem calcular la rotació entre ambdues imatges.

Per a realitzar la rotació s’agafa l’origen de coordenades com a punt de referència. A partir d’aquest punt de referència tots els altres punts de la imatge rotaran al seu voltant segons el valor donat de rotació. Per tant aquest origen de coordenades és necessari que es situï al centre de la imatge.

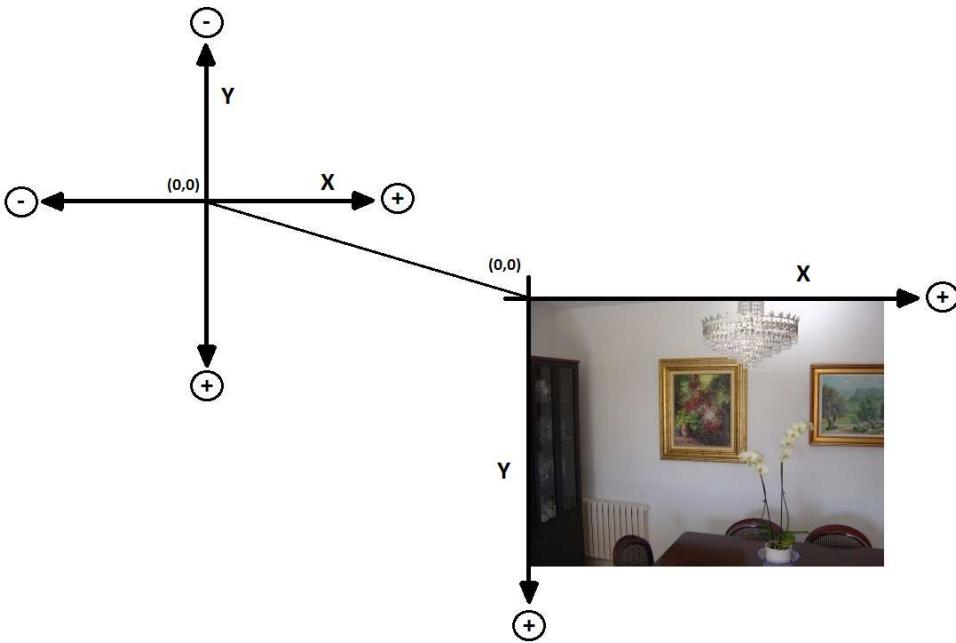


Figura 30 Transformació centre de coordenades.

Per a realitzar el canvi de referència de centre de coordenades (Figura 30) haurem d'aplicar una senzilla formula a les coordenades dels *features* guardats, tant al seu eix X com al seu eix Y. Calculant la mida de la imatge tant en files com en columnes de píxels, restarem a les coordenades dels *features* la meitat del total de files de la imatge a les coordenades Y i la meitat del total de les columnes a les coordenades X de cada feature.

$$\text{Coordenades}X_{nou} = \text{Coordenades}X_{inicial} - \left(\frac{\text{Total}_{\text{columnnes}}}{2} \right) \quad (25)$$

$$\text{Coordenades}Y_{nou} = \text{Coordenades}Y_{inicial} - \left(\frac{\text{Total}_{\text{files}}}{2} \right) \quad (26)$$

4.2.3 Transformació píxel - metre.

Arribats a aquest punt ja tenim les coordenades dels *features*, filtrats mitjançant el llindar “numMin” i ordenats. Aquesta informació és essencial per a poder calcular les distàncies entre imatges. Durant tot aquests passos sempre hem estat parlant en píxels, ja que la càmera treballa en píxels i les coordenades dels nostres *features* estan ubicats i referenciats en píxels. Si volem calcular la distància real entre imatges necessitarem fer una transformació de píxel a metres. Per tant necessitarem saber la relació que tenen els píxels capturats per la càmera amb la distància real representada en la imatge.

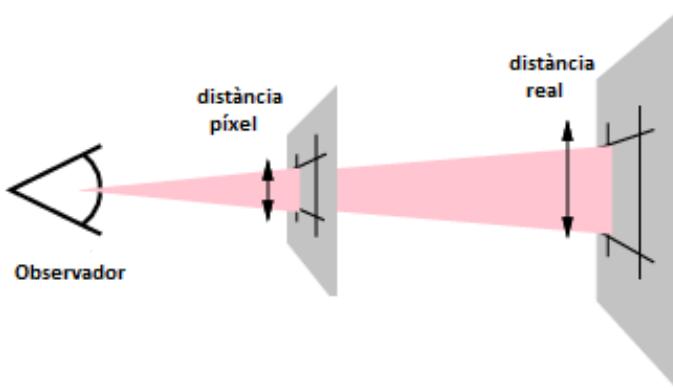


Figura 31 Distància píxel i distància real.

Per a saber quina és la relació entre la distància entre píxels en la imatge i la distància real representada en la mateixa (Figura 31) necessitarem utilitzar de nou el model Pin-Hole. Utilitzant els paràmetres intrínsecs i la altura a la qual esta la càmera del sòl podrem calcular la relació. Aquesta relació ve donada per la següent fórmula.

$$\begin{bmatrix} X_m \\ Y_m \\ H \end{bmatrix} = \begin{bmatrix} F_x & 0 & 0 \\ 0 & F_y & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} X_p \\ Y_p \\ 1 \end{bmatrix} \quad (27)$$

F_x i F_y són les distàncies focals de la càmera. H és la distància de la càmera al sòl en metres en el moment en el que es va prendre la imatge. Aquest valor és un valor que pot anar variant a mesura que la càmera es mou ja que l'altura de la càmera pot anar variant. En el nostre cas aquest valor sempre serà constant ja que la nostra càmera no experimentarà cap variació d'altura. X_p i Y_p són els valors de les coordenades expressades en píxels. X_m i Y_m són els valors finals de les coordenades expressats en metres.

Una vegada tenim l'equació general amb les matrius i dades corresponents podem operar-les i deixar dues equacions senzilles per a la transformació de les dades.

$$X_m = (F_x * X_p + 0 * Y_p + 0 * 1) * H = F_x * X_p * H \quad (28)$$

$$Y_m = (0 * X_p + F_y * Y_p + 0 * 1) * H = F_y * Y_p * H \quad (29)$$

Aquestes operacions s'aplicaran una per una a cada coordenada (X, Y) dels dos llistats dels *matchings* que tenim guardats, passant completament tota la informació que tenim a metres.

Els càlculs que realitzarem per a estimar l'odometria passaran a ser distàncies reals (metres). Els quals ens serviran per a una estimació real i podrem comprovar empíricament el moviment realitzat pel nostre mòbil.

Capítol 4. Càcul de moviment

La relació de moviment entre les dues imatges consecutives analitzades és vital per al càlcul de l'odometria. En el present projecte aquest pas té com a base la utilització de Mínims Quadrats, algorisme que ens calcularà la translació i rotació que hi ha entre dues imatges consecutives. I una de les parts més importants és la implementació de RANSAC per a aquesta tasca, que ens donarà algunes solucions a possibles problemes en el càlcul del moviment fent al nostre programa més robust. Explicarem l'algorisme utilitzat al nostre programa en quant a opcions que podrà elegir l'usuari en la utilització de Mínims Quadrats o RANSAC i també les possibles solucions quan RANSAC no trobi solució.

Al final del capítol s'explicaran també els paràmetres d'entrada i sortida del nostre programa. Així com la lectura de les dades de sortida i visualització de resultats amb el programa MATLAB.

1. Mínims Quadrats.

Per a estimar el moviment complet que ha realitzat el robot durant una trajectòria, hi ha diverses maneres per poder calcular-lo. Una de les maneres és la utilització de les formules per a l'estimació de moviment de Mínims Quadrats utilitzant punts en 2D. Donades dues bateries de punts (Figura 32), que corresponen a *features* emparellats de ambdues imatges, l'aplicació d'aquestes formules ens estimaran el moviment que ha realitzat el robot entre les dues imatges consecutives capturades. Aquest procés es durà a terme amb totes les imatges adquirides per la càmera durant tot el recorregut del robot.

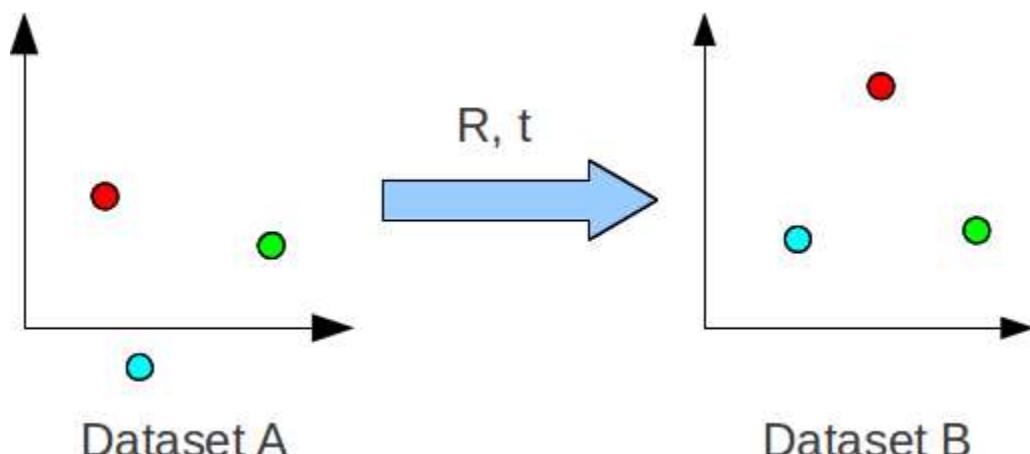


Figura 32 Dataset de imatges consecutives.

Si comparem la imatge A amb la imatge B, suposant que els punts de colors iguals són el resultat del *matching* prèviament realitzat. La pregunta és ¿quant hem de moure la imatge A en quant a rotació i translació per a que els punts encaixin dins els punts del mateix color de la imatge B?.

1.1 Equacions.

Per a realitzar el càlcul d'estimació del moviment ens basarem en l'adaptació de l'algorisme ICP que realitzaren Feng Lu i Evangelos Milios [7] per a l'ús amb sensors de distància, com llàser i sonar, en robòtica i que proporcionen una solució de càlcul al problema de Mínims Quadrats. En el present projecte hem adoptat aquesta adaptació de l'algorisme ICP.

Com hem esmentat abans, tenim dues llistes de punts que corresponen a *features* emparellats entre les dues imatges. Les seves coordenades "x" e "y" estan expressades en metres: $P = (x_i, y_i)$, $P' = (x'_i, y'_i)$. Sabent que "n" és el nombre total de punts de cada imatge.

$$E_{dist}(\theta, T) = \sum_{i=1}^n d^2(P_i, X \oplus P'_i) \quad (30)$$

$$X = \begin{pmatrix} T_x \\ T_y \\ \theta \end{pmatrix} \quad (31)$$

On E_{dist} ve donat per el sumatori de la distància euclidiana " d^2 " entre cada punt emparellat de ambdues imatges on "X" és la matriu de moviment. \oplus és l'operació que compon els punts P'_i amb la roto-translació especificada en X.

E_{dist} és l'error total en la rotació i translació de tots els punts entre ambdues imatges. El que es pretén és trobar el valor de $X = (T_x, T_y \ i \ \theta)$ que minimitza aquest error total perquè és molt poc probable tenir un error zero. Els punts entre les dues imatges difícilment estarán situats sense cap error de posició i per l'alta probabilitat de falsos positius.

Podem dir que Lu and Milios proporcionen una solució tancada als valors de X per a minimitzar l'error. Si desenvolupem l'equació principal ens queda la següent equació.

$$\begin{aligned} E_{dist}(\theta, T) = & \sum_{i=1}^n \left((x_i \cos(\theta) - y_i \sin(\theta) + T_x - x'_i)^2 \right. \\ & \left. + (x_i \sin(\theta) + y_i \cos(\theta) + T_y - y'_i)^2 \right) \end{aligned} \quad (32)$$

Mitjançant la solució que proposen Lu and Milios podem trobar el valor de θ que minimitza l'error amb la següent equació.

$$\theta = \arctan \left(\frac{S_{xy'} - S_{yx'}}{S_{xx'} + S_{yy'}} \right) \quad (33)$$

On S_{xx} , S_{yy} , S_{xy} i S_{yx} es detallen més endavant.

Una vegada s'ha calculat l'estimació de l'orientació " θ ", passarem a calcular la translació en quant a l'eix X com a l'eix Y mitjançant dues equacions per a cada cas.

$$T_x = X'_m - (X_m * \cos(\theta) - Y_m * \sin(\theta)) \quad (34)$$

$$T_y = Y'_m - (X_m * \text{Sin}(\theta) + Y_m * \text{Cos}(\theta)) \quad (35)$$

Primerament haurem de calcular tots els sumatoris corresponents. També realitzarem el sumatori del producte de dos eixos entre imatges suposant totes les combinacions possibles.

Una vegada ja tenim tots els sumatoris calculats passarem a calcular les mitjanes aritmètiques i altres càlculs necessaris .

$$X_m = \frac{\sum_{i=1}^n x_i}{n}, X'_m = \frac{\sum_{i=1}^n x'_i}{n} \quad (36)$$

$$Y_m = \frac{\sum_{i=1}^n y_i}{n}, Y'_m = \frac{\sum_{i=1}^n y'_i}{n}$$

$$S_{xx'} = \sum_{i=1}^n x_i * x'_i - \left(\sum_{i=1}^n x_i * \sum_{i=1}^n x'_i \right) \quad (37)$$

$$S_{yy'} = \sum_{i=1}^n y_i * y'_i - \left(\sum_{i=1}^n y_i * \sum_{i=1}^n y'_i \right)$$

$$S_{xy'} = \sum_{i=1}^n x_i * y'_i - \left(\sum_{i=1}^n x_i * \sum_{i=1}^n y'_i \right)$$

$$S_{yx'} = \sum_{i=1}^n y_i * x'_i - \left(\sum_{i=1}^n y_i * \sum_{i=1}^n x'_i \right)$$

Ara que ja tenim tots els càlculs inicials realitzats podem passar a aplicar les equacions principals esmentades abans que ens donaran l'estimació de moviment entre ambdues imatges $X = (T_x, T_y \ i \ \theta)$.

1.2 Aplicació al nostre programa.

A continuació es mostrerà un exemple de resultat de l'aplicació de Mínims Quadrats entre dues imatges. Aquestes dues imatges s'han seleccionat d'un vídeo de prova al qual la càmera es mou en paral·lel al sòl. De manera semblant a com es duran a terme les proves finals.

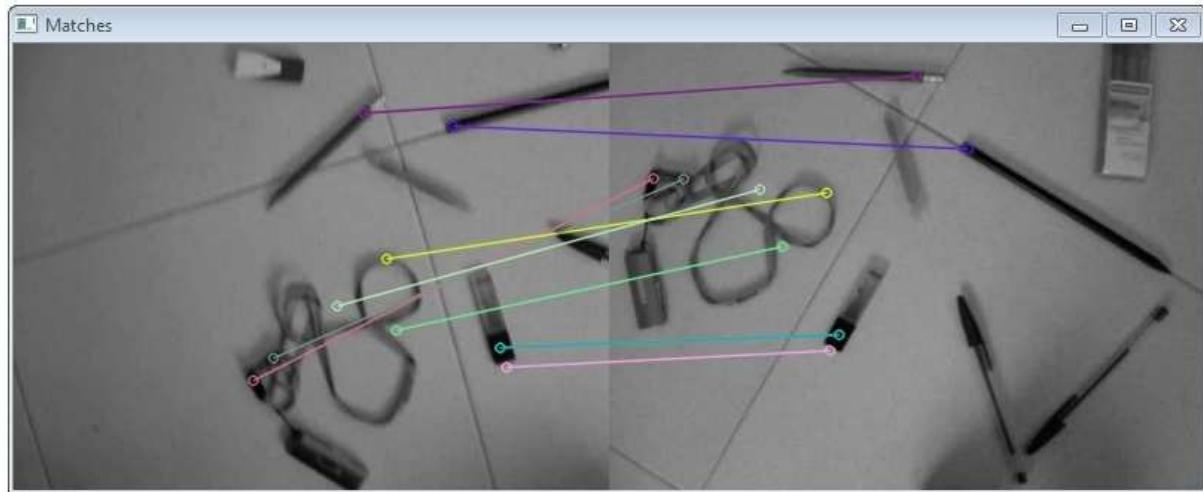


Figura 33 Matching de les dues imatges de prova: imatge A (esquerra), imatge B (dreta).

Primerament el programa realitzarà el corresponent *matching* dels *features* i l'imprimim per pantalla com es veu a la Figura 33. Una vegada hem realitzat les modificacions adients a les dades obtingudes s'aplicarà Least Square al llistat de *features* emparellats entre les dues imatges i imprimirem per pantalla el resultat en píxels i graus centígrads (Figura 34).

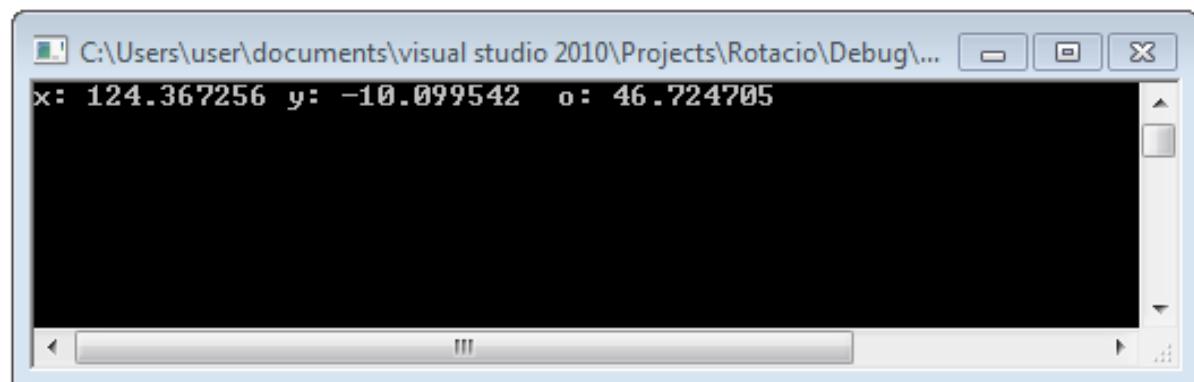


Figura 34 Resultat de estimació de moviment amb mínims quadrats(píxels i graus centígrads).

Per veure d'una manera més gràfica el moviment de la imatge, s'ha rotat la segona imatge tants de graus com hem estimat amb Mínims Quadrats (Figura 35). En aquest exemple la imatge s'ha rotat 46,72 graus.

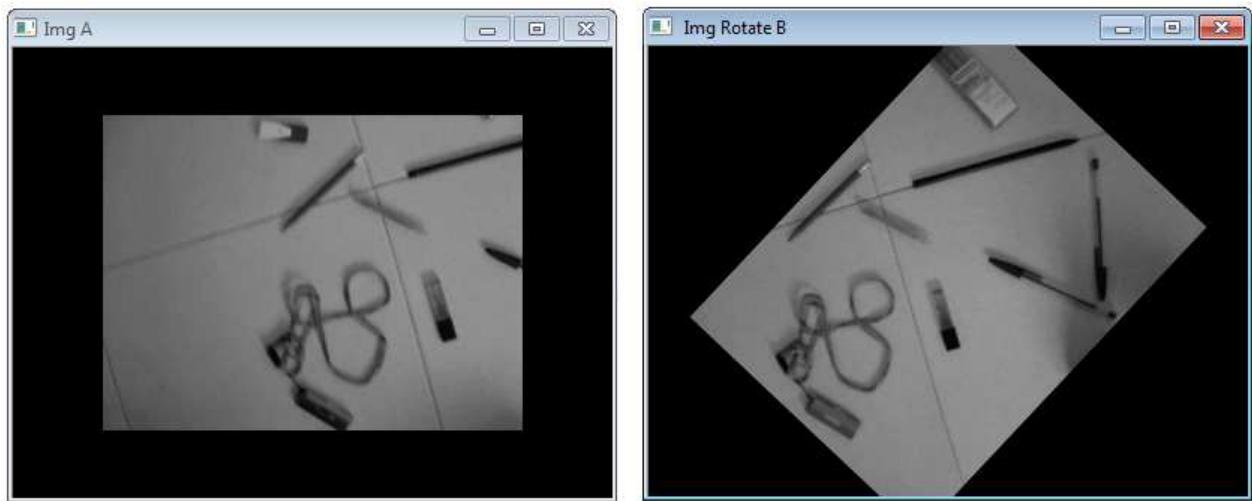


Figura 35 Rotació de la imatge B.

Una vegada hem rotat la imatge, compararem les dues imatges superposant-les amb el moviment del píxels estimat per Mínims Quadrats (Figura 36). Com podem observar les dues imatges encaixen quan les superposem, així podem provar que els resultats són fiables si no hi ha grans errors en *matching*.

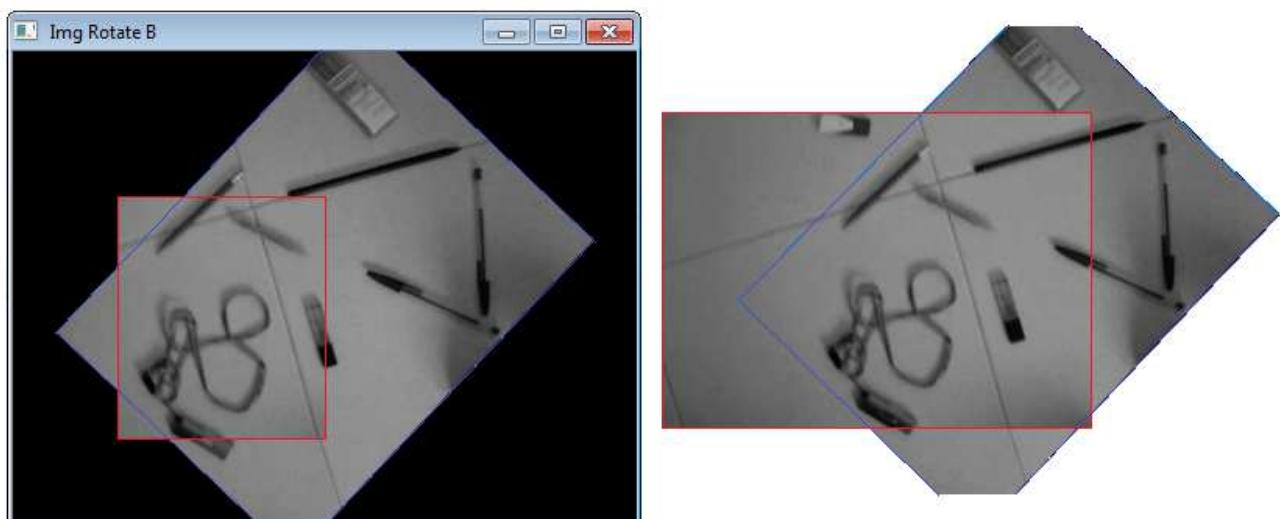


Figura 36 Superposició de imatges: superposició parcial (esquerra), superposició completa (dreta).

1.3 Problemes.

Mínims Quadrats no és un estimador robust ja que es veu afectat per espuris. Aquests espuris afecten a l'estimació de moviment de manera semblant a com afecten els espuris al càclul d'una recta en un núvol de punts.

Existeixen estimadors robustos que van un pas més enllà. Com per exemple M-estimation, Least trimmed squares, S-estimation, MM-estimation, Theil-Sen estimator, Relaxed intersection, RANSAC [8].

L'algorisme RANSAC ha demostrat sobradament que ser capaç d'eliminar espuris [9],[10] i, per tant proporcionar estimacions fins i tot amb gran quantitat de dades errònies. A més, es considera un algorisme eficient.

Per aquest motiu, entre totes les tècniques d'estimació robusta ens hem decantat per RANSAC.

2. RANSAC.

RANSAC és la abreviació de “Random Sample Consensus”, és un algorisme per a estimar els paràmetres d'un model matemàtic que ve donat per un conjunt de dades les quals contenen valors atípics. En el nostre cas aquest valors provindran, majoritàriament, de falsos positius o emparellaments erronis. Com abans hem esmenat els principals errors venen majorment d'aquests valors atípics els qual RANSAC tractarà en la mesura de lo possible detectar i eliminar. A continuació veurem un exemple de dos casos diferents en quant a proporció de falsos *matchings*.

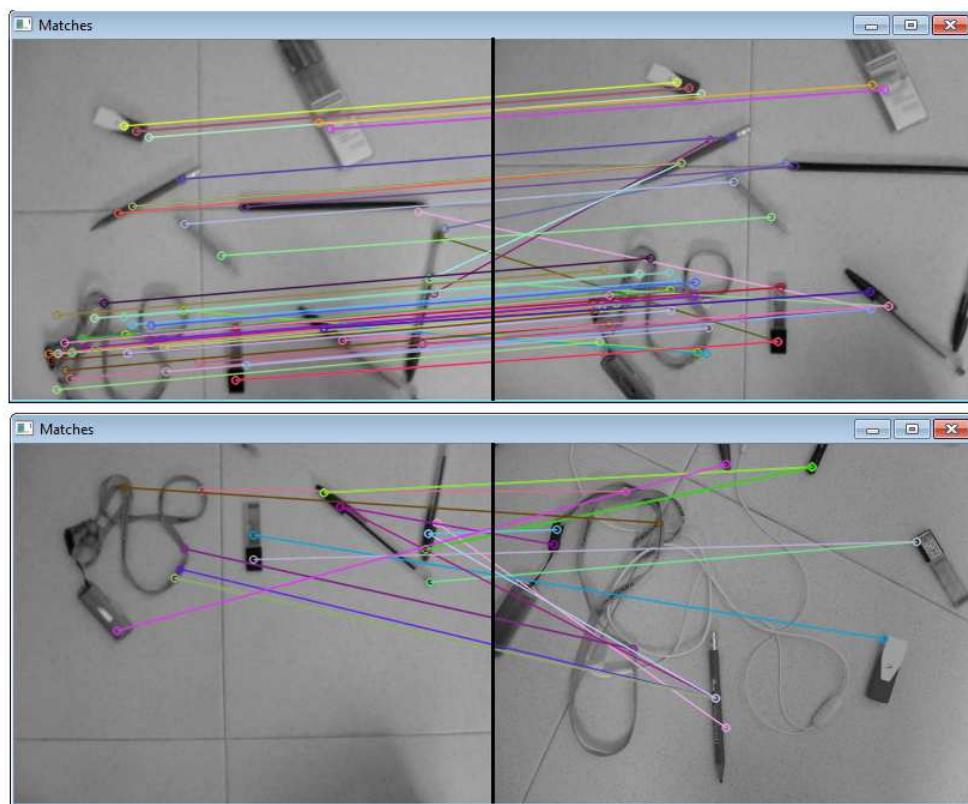


Figura 37 Matching entre imatges: els features parells tenen un patró clar (dalt), els features parells no tenen un patró clar (baix).

Com es pot observar a (Figura 37-dalt), es pot veure un patró clar dels *matchings* realitzats amb alguns falsos positius però en una baixa proporció i per tant que ha pogut capturar el moviment. Mentre que a (Figura 37-baix) no es troba cap patró clar i per tant podem arribar a la conclusió de que s'han trobat molts de *matchings* erronis.

En un conjunt de *matchings* entre dues imatges on hi ha *matchings* bons i dolents, es pot dir que el càlcul de la roto - translació de qualsevol combinació de *matchings* bons serà sempre el mateix. En canvi el càlcul amb *matchings* dolents serà diferent. En altres paraules els *matchings* bons es poden expressar amb un sol model mentre que els *matchings* erronis no, ja que constitueixen els valors atípics del conjunt de dades. Per tant RANSAC tractarà de treure aquest model en la mesura de lo possible descartant els falsos *matchings*.

Aquest algorisme té les seves limitacions. Per un elevat percentatge de falsos *matchings* RANSAC no serà capaç de trobar-nos una solució ja que la informació que ens aporten les imatges no és suficient com per establir un model. En aquests casos haurem de cercar altres vies alternatives que posteriorment explicarem.

2.1 Algorisme.

A continuació explicarem l'algorisme de RANSAC que hem aplicat al nostre programa. Com podem observar, a la Figura 38 s'exposa l'algorisme en pseudocodi i que posteriorment explicarem detalladament.

```
INPUT:
    dades: dos vectors de dades amb les coordenades dels matchings
    numIterations: nombre total d'iteracions de l'algorisme
    numDatapoints: nombre de matchings aleatoris inicials
    maxError: llindar de distància entre punts
    minConsensus: nombre mínim de matchings per tenir un bon model
    n: nombre total de matchings
OUTPUT:
    goodProcess: com a mínim tenim un model bo
    bestModel: roto-translació d'un model bo

Iniciem
for(k=0; k <= numIterations; k++){
    for(i=0; i<=numDatapoints; i++){
        maybeInliers = guardem aleatoriament un matching de tot el
                      conjunt
    }
    maybeModel = realitzem Min. Quadrats amb els punts maybeInliers
    for(j=0; j <= n; j++){
        rotem i traslladem el feature "j" de tot el conjunt amb el
          model maybeModel
        si el punt transformat té un error < maxError guardarem
          el punt a maybeInliers
    }
    Si numero de maybeInliers > minConsensus
        goodProcess = true
        thisModel = realitzam Min. Quadrats amb els punts guardats a
                   maybeInliers
        thisError = computem l'error comés per thisModel

        Si thisError < bestError
            bestError = thisError
            bestModel = thisModel
    }
    Return bestModel
```

Figura 38 Algorisme de RANSAC.

– **Bucle principal.**

L'algorisme està compost per un bucle principal que nosaltres fixarem prèviament amb un màxim de repeticions. Aquesta constant es anomenada “numIterations”. Aquest nombre de repeticions sol ser un nombre bastant alt ja que l'algorisme de RANSAC ha de seleccionar aleatoriament un conjunt de *matchings* que ens servirà per a estimar prèviament el model.

Podem arribar a tenir entre 100 i 300 *matchings* bons entre les dues imatges que emprarem, per tant, un nombre elevat de dades on hi ha infinitat de possibles combinacions entre *matchings*. Per tant es necessita un nombre elevat de repeticions per a obtenir amb més probabilitat solucions adequades.

Es seleccionarà el nombre inicial de punts aleatoris per al càlcul del model previ, aquest paràmetre s'anomena “numDatapoints”. Aquest nombre com a mínim haurà d'esser 2, ja que amb només un *matching* no és possible realitzar Mínims Quadrats. Es recomanable que el nombre no sigui molt elevat, ja que quant més elevat sigui el nombre més probabilitats de seleccionar un fals *matching*.

– **Eliminació de matchings dolents.**

Utilitzant la roto-translació calculada amb els *matchings* aleatoris anomenat “maybeModel”, rotarem tots els punts de la segona imatge (*frame* actual) menys els que ja hem seleccionat aleatoriament.

A la vegada que transformem els punts, els comparem amb els seus respectius *matchings* de la primera imatge (*frame* anterior). Mesurant l'error comés per el model de roto-translació calculat. Aquest error s'esbrina calculant la distància que hi ha entre els dos punts.

Es compara la distància calculada (error) amb una distància llindar estableguda anteriorment amb una constant anomenada “maxError”. Si la distància calculada és menor que la distància llindar podem suposar que es tracta d'un bon *matching* i per tant es guardarà al vector “maybelnliers”. Aquest llindar d'error es pot suposar com un perímetre circular al voltant del feature, si el *matching* transformat entra dins el perímetre d'error es donarà per bo.

– **Distinció de models bons i dolents.**

Una vegada rotats i comparats tots els punts tindrem un nombre determinat de *matchings* que hauran estat seleccionats com a bons *features* en el vector “maybelnliers”. Per a poder decidir si el model calculat inicialment pels *matchings* aleatoris és bo, es mira si el nombre de *matchings* guardats a “maybelnliers” és superior a un percentatge llindar de *matchings* bons amb respecte al total inicial.

Aquest percentatge ve estableert per la constant anomenada “minConsensus”, constant que serà posteriorment analitzada amb les proves finals que realitzarem.

Si el resultat de la comparació amb el percentatge mínim de consens és positiu, es calcula la roto-translació definitiva per aquest model en concret amb tots els *matchings* que hem anat calculat anteriorment. Quants més *matchings* bons tinguem, més podrem minimitzar l'error amb Mínims Quadrats.

- **Computació de l'error del model.**

Generalment es tindran varis models que haurem definit com a bons. Haurem de quedar-nos amb el model més bo de tot el conjunt de models. Per tant, per a cada model calcularem el seu error comés i seleccionarem el que menor error tengui.

Aquest error es calcularà com abans hem fet, tornarem a traslladar cada *matching* bo del *frame* actual prèviament guardat amb el model calculat amb tots els *matchings* bons. Es mesura la distància “error” de cada *matching* roto-traslladat, realitzant un sumatori d’error acumulat. El model que tingui el sumatori d’error més baix es guardarà a “bestModel” i serà la dada de sortida de l’algorisme RANSAC.

2.2 Parametrització de RANSAC.

La implementació de RANSAC en el programa requereix d’una configuració per al mateix algorisme. Aquesta configuració serà clau per al funcionament de l’algorisme i del resultat de les nostres posteriors proves. Alguns d’aquests paràmetres els seleccionarem i avaluarem posteriorment.

Els paràmetres que emprarem per RANSAC són els següents:

- **NumIterations:** Paràmetre que permet fixar el nombre de vegades que realitzarem l’algorisme aleatori de RANSAC en la recerca d’un model bo.

Aquest paràmetre haurà d’esser un nombre alt sempre i quan el nombre de *matchings* que manegem sigui alt. En canvi si el nombre de *matchings* és baix en un determinat *frame* no seran necessaris tantes iteracions. En el nostre programa aquest paràmetre es mantindrà constant.

- **Numdatapoints:** Paràmetre que permet fixar el nombre de *features* aleatoris inicials per al primer càlcul del possible bon model.

Aquest paràmetre necessita un nombre mínim, ja que per a calcular la roto-translació inicial és d’un nombre de 2. En canvi no es recomanable que s’empri un nombre elevat ja que hi ha més possibilitats de seleccionar aleatòriament un fals *matching* i que el model calculat per Mínims Quadrats acumuli error.

- **MaxError:** Paràmetre que permet fixar l’error el qual acceptarem *features* per a un possible bon model. Aquest paràmetre és clau per a eliminar falsos *features* ja que un “radi d’error” més ampli pot fer seleccionar un fals *feature* com a bo.

Es pot dir doncs que aquest paràmetre necessàriament ha d'esser baix, però no excessivament ja que totes les localitzacions compten amb un cert error.

- **MinConsensus:** Paràmetre que permet fixar el percentatge de *features* bons que són necessaris per a poder establir que un model ha estat bo. Aquest paràmetre pot ser més variable que la resta ja que depèn també de la qualitat i quantitat dels *matchings* obtinguts inicialment.

En aquest apartat no especificarem els valors concrets per a cada paràmetre ja que es realitza més endavant en el document. Els valors dels paràmetres els determinarem experimentalment.

2.3 Aplicació al programa.

A continuació es mostrerà visualment els resultats de la implementació de RANSAC (Figura 39). Per a explicar l'exemple s'han pres dos *frames* consecutius d'un vídeo simulant el moviment d'un mòbil. Analitzarem ambdós *frames* mitjançant les anteriors explicacions (SIFT i FLANN) i posteriorment aplicarem RANSAC el qual podrem observar els seus resultats.

A la part superior de la Figura 35 podem veure els *matchings* obtinguts mitjançant SIFT i FLANN. Com es pot observar de forma general els *matchings* segueixen un patró clar però també es poden observar alguns falsos *matchings* o valors atípics.

A partir de la informació inicial obtinguda, aplicarem l'algorisme de RANSAC. Com podem veure als *matchings* de la part superior, si es realitza l'odometria amb tota la informació sense aplicar RANSAC s'acumularia l'error dels falsos *matchings*.

Una vegada realitzat RANSAC, s'ha dibuixat a la finestra inferior de la Figura 39 el llistat de *matchings* bons (*maybelnliers*) del millor model estimat per RANSAC. Com es pot observar tots els *matchings* falsos han estat eliminats. També s'han eliminat *matchings* que a priori pareixien bons però al no entrar dins la distància mínima d'error “maxError” RANSAC els ha eliminat. En el cas d'aquest exemple (Figura 39) és de 3 píxels de distància.

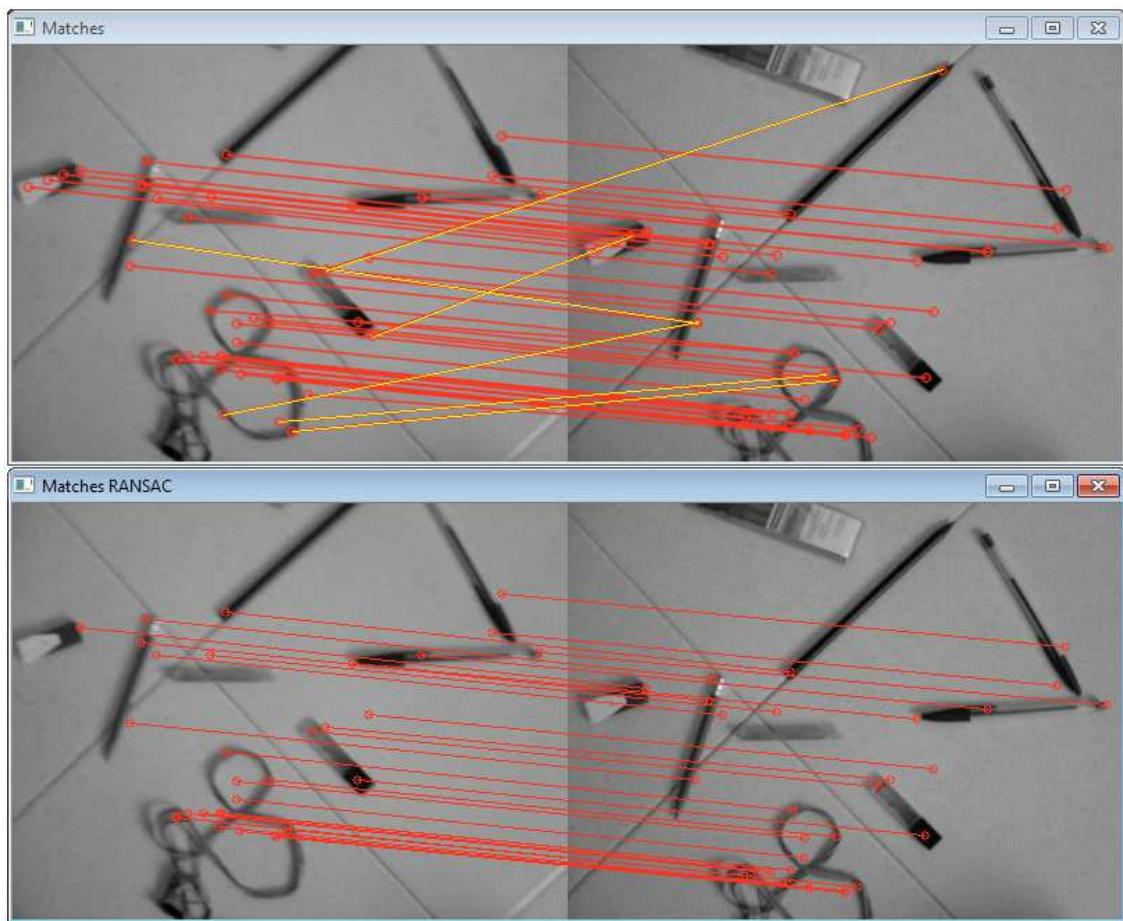


Figura 39 Exemple d'eliminació de matchings amb RANSAC. Matching sense aplicar RANSAC (imatge superior) amb espuris remarcats en color gros. Eliminació de matchings dolents mitjançant RANSAC (imatge inferior).

3. Implementació de l'algorisme.

A continuació s'explicarà com s'ha implementat l'estimació de moviment en el nostre algorisme. Aquesta implementació es correspon a l'apartat (3) de l'esquema general de l'algorisme (Figura 25).

En aquest apartat s'explica principalment com hem utilitzat RANSAC i Mínims Quadrats. També les possibles solucions quan RANSAC no troba una solució a l'estimació de moviment.

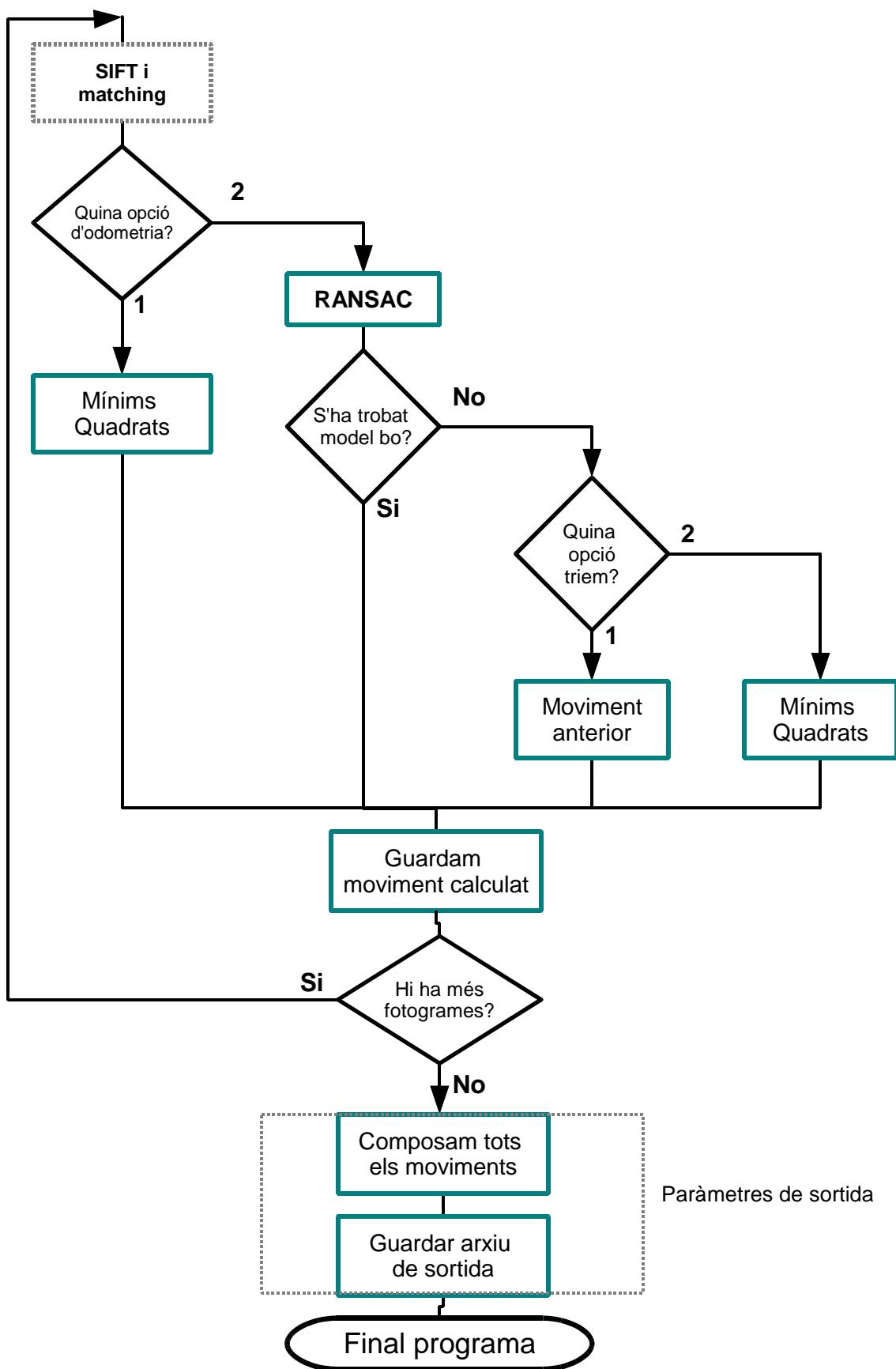


Figura 40 Esquema general del càlcul de moviment.

L'esquema general de la Figura 40 ens dóna un concepte general del que realitza el nostre programa segons les opcions que haguem escollit per a realitzar l'odometria. Aquestes opcions són les que posteriorment analitzarem amb les proves que més endavant explicarem.

Els passos que apareixen en gris puntejat no els explicarem en aquest apartat, això vol dir que ja han estat explicats anteriorment.

– **Opció general d'odometria.**

Una vegada s'ha realitzat la detecció i el *matching* es realitzarà el càlcul de moviment que ens donarà quantitativament el moviment realitzat per la càmera entre dos *frames*. Per això al nostre programa es proposen dues vies principals per aquest càlcul: Mínims Quadrats o RANSAC.

Aquesta opció estarà guardada a una constant que prèviament haurem indicat a l'arxiu d'entrada. Amb aquesta opció es pot elegir fer l'estimació de moviment amb Mínims Quadrats o RANSAC.

L'elecció de Mínims Quadrats fa que el programa realitzi l'estimació de moviment directament amb les dades proporcionades per el *matching*. Realitzant Mínims Quadrats entre els *features* emparellats. Totes les altres opcions relacionades amb RANSAC no es tindran en compte si aquesta opció es selecciona. Guardarem directament el resultat de l'estimació de moviment al nostre vector de dades.

Si l'elecció és RANSAC, hem de tenir en compte que a diferencia de Mínims Quadrats, no sempre es tindrà una solució a l'estimació de moviment. Si el percentatge de *features* emparellats que RANSAC dóna per bons són superiors al percentatge llindar seleccionat per l'usuari, l'estimació de moviment es donarà per conclosa. Guardarem el resultat final de RANSAC al vector de moviment.

Però si RANSAC no ens troba cap model bo, haurem d'optar per altres vies alternatives.

– **Opció de RANSAC.**

Si al aplicar RANSAC entre les dues imatges no s'aconsegueix cap model bo es passarà a realitzar una de les dues vies alternatives que el nostre programa integra.

Aquesta opció serà seleccionada amb antelació a l'arxiu d'entrada i l'usuari pot seleccionar una de les dues opcions (*optionRansac*). Aquesta opció és una de les que seran estudiades amb les proves que realitzarem.

Les dues opcions són:

- **Mínims Quadrats:** Es realitzarà Mínims Quadrats amb la llista sencera de *matchings*. Guardarem el resultat al vector de moviment.
- **Moviment anterior:** Es guardarà el moviment anterior com a moviment actual. Durant el *frame* actual no hem pogut estimar el moviment i per tant suposarem

que s'ha realitzat el mateix moviment que l'anterior *frame* com a mesura per assegurar un resultat estable.

Cada vegada que es realitza l'odometria de cada *frame*, aquesta odometria calculada es guarda en tres vectors. Cada vector correspon a cada component que expressa la distància calculada: un vector per a X, un per Y i un per θ . Acumulant així els moviments realitzats pel nostre mòbil en cada *frame*. Més endavant s'hauran de compondre tots els moviments de cada *frame* amb la informació que es té guardada dins aquests tres vectors. Aquest pas l'explicarem més endavant.

4. Visualització per pantalla.

El nostre programa mostra per pantalla la informació que s'extreu entre cada *frame*. Això permet a l'usuari poder fer un seguiment de l'anàlisi del vídeo mentre s'executa (Figura 41). Cada vegada que s'analitza un *frame* es mostrerà per pantalla les dades més destacades que ens interessa saber.

```
-- Frame 22 --
FLANN // Total matches: 88 --> Good Matches: 78
Porcentaje:70 // MatchesTot:78 --> NumeroMin:54
Error, cap model amb RANSAC trobat!
FRAME ANTERIOR --> X:0.003275 Y:0.052939 0:2.023557
CPU time used: 140 ms
-- Frame 23 --
FLANN // Total Matches: 82 --> Good Matches: 23
Porcentaje:70 // MatchesTot:23 --> NumeroMin:16
model trobat amb 17 matches bons
X:0.013410 Y:0.034464 0:-1.218611
CPU time used: 141 ms
-- Frame 32 --
FLANN // Total Matches: 61 --> Good Matches: 35
Porcentaje:70 // MatchesTot:35 --> NumeroMin:24
Model trobat amb 26 matches bons
X:0.003081 Y:-0.009960 0:-0.129383
CPU time used: 156 ms
-- Frame 35 --
FLANN // Total Matches: 83 --> Good Matches: 67
Porcentaje:70 // MatchesTot:67 --> NumeroMin:46
Error, cap model amb RANSAC trobat!
FRAME ANTERIOR --> X:0.003081 Y:-0.009960 0:-0.129383
CPU time used: 172 ms
```

Figura 41 Visualització per pantalla de l'anàlisi.

Com podem veure a la Figura 41, dins el quadre verd es pot veure el nom “Frame 22” que ens indica el *frame* que ha estat analitzat i comparat amb l'anterior. Aquest anterior *frame* no significa necessàriament que sigui el *frame* 21. Com abans hem explicat el paràmetre “Xframe” ens defineix cada quants *frames* volem realitzar l'anàlisi (en aquest cas cada 5 *frames*).

A continuació ens indica l'estat dels *matches* entre els dos *frames* analitzats (quadre taronja). Aquí ens apareix el nombre total de *matches* trobats per FLANN “Total matches” i el filtrat realitzat també per FLANN “Good Matches” on utilitza les distàncies dels *matchings* i el paràmetre “numMin” seleccionat per l'usuari .

Si l'opció seleccionada és RANSAC, ens apareix indicat l'opció de percentatge de consens en "Percentatge". Aquest percentatge l'aplicarem al nombre de total de *matches* bons que tenim i sabrem quin mínim de *matches* necessita RANSAC per un model bo. Aquesta apartat serveix per veure el nombre de *matches* que maneja el programa dins aquest interval.

Si s'ha trobat un model bo de RANSAC, ens apareixerà un missatge com el del quadre blau. El programa ens indica amb quants *matches* bons ha seleccionat el model (sempre superior a "NumeroMin"). A continuació el programa ens imprimeix el resultat del model (X, Y i θ) que posteriorment es guardarà al vector d'odometria.

Si RANSAC no ha trobat cap model bo, el programa ens ho indicarà (quadre vermell). Indicant a continuació quina opció ha escollit l'usuari, en el cas de l'exemple s'ha escollit l'odometria del *frame* anterior.

Al final de l'anàlisi de cada *frame* es mostrarà per pantalla la duració del procés en milisegons (quadre blau).

5. Paràmetres de sortida del programa.

Aquesta part del programa representa l'apartat 4 de l'esquema general del programa (Figura 25). Aquest apartat explica la configuració final de les dades i guardar-les en un arxiu final de sortida.

Una vegada el programa ha realitzat l'anàlisi complet de tots els *frames* que componen el vídeo, es tenen guardats tots els moviments de la càmera dins els vectors de moviment.

A continuació necessitarem que les dades de moviment obtingudes pel nostre programa estiguin configurades per a la lectura final del moviment del robot. Per tant necessitarem compondre les dades de moviment obtingudes. Compondre els moviments significa que estableirem el mateix origen de coordenades a tots els moviments realitzats per el nostre mòbil.

Finalment necessitarem guardar tots els moviments compostos en un fitxer de sortida que el nostre programa realitzat en MATLAB llegirà i interpretarà per a visualitzar els resultats de l'odometria.

5.1 Composició del moviment.

Les dades d'odometria inicials que obtenim entre cada *frame* són moviments que per si mateixos no tenen cap interpretació pràctica, ja que l'origen de coordenades de cada moviment és diferent i no tenim cap punt de referència clar. Aquests posicions s'anomenen *posicions relatives*.

$$X_k = [X_1^0, X_2^1, X_3^2, X_4^3, \dots, X_k^{k-1}]^T \quad (38)$$

On X_i^{i-1} és la roto-traslació entre el *frame* (i-1) i el *frame* (i). Podem dir que, cada moviment guardat dins els vectors de dades és un increment respecte del moviment anterior. X_1^0 és la primera posició relativa del robot que fixa l'origen de coordenades global del sistema. Aquest origen de coordenades normalment es sol indicar com $X_1^0 = [0, 0, 0]^T$.

Mitjançant l'origen de coordenades global X_1^0 podem referenciar totes les posicions relatives i passar-les a posicions absolutes, és a dir, que tinguin el mateix origen de coordenades. Aquesta és una senzilla operació que es pot realitzar de la següent manera:

$$X_k^0 = X_1^0 \oplus X_2^1 \oplus X_3^2 \oplus \dots \oplus X_k^{k-1} \quad (39)$$

On X_k^0 és la posició absoluta entre origen de coordenades X_1^0 i la darrera posició relativa X_k^{k-1} . L'operació \oplus és la composició entre els moviments relatius. Aquesta operació es pot descriure de la següent manera:

$$x_i^0 = x_{i-1}^0 + x_i^{i-1} * \cos(\theta_{i-1}^0) - y_i^{i-1} * \sin(\theta_{i-1}^0) \quad (40)$$

$$y_i^0 = y_{i-1}^0 + x_i^{i-1} * \sin(\theta_{i-1}^0) + y_i^{i-1} * \cos(\theta_{i-1}^0) \quad (41)$$

$$\theta_i^0 = \theta_{i-1}^0 + \theta_i^{i-1} \quad (42)$$

On sabem que la posició relativa $X_i^{i-1} = [x_i^{i-1}, y_i^{i-1}, \theta_i^{i-1}]$. Doncs podem dir que x_i^0, y_i^0, θ_i^0 són els components de la posició absoluta X_i^0 . Aquesta operació de composició es realitza i s'acumula el resultat fins que ($i = k$), és a dir, el darrer *frame*.

Al nostre programa guardarem cada posició absoluta calculada a un vector de dades. Mitjançant el programa que hem realitzat amb MATLAB podem llegir i imprimir totes les posicions del nostre robot i visualitzar el recorregut realitzat.

5.2 Fitxer de sortida.

Una vegada s'han realitzat totes les tasques del nostre algorisme, es guardarà la informació per a poder ser interpretada posteriorment pel nostre programa en MATLAB. Per aquest propòsit el nostre programa crearà un arxiu de sortida amb tota la informació necessària per ser classificada i interpretada.

Guardarem tots els moviments realitzats per el robot dins un arxiu del tipus ".txt". També hi guardarem el temps total d'execució del programa per a calcular tota l'odometria.

```

A1B3C4: Bloc de notas
Archivo Edición Formato Ver Ayuda
182200.000000 0 0
0.000000 0.000000 0.000000
0.000159 -0.000063 -0.000918
-0.000518 0.000091 0.001305
-0.000504 0.000044 0.001261
-0.003800 0.000808 -0.006973
-0.004478 -0.000694 -0.005952
-0.011685 -0.010409 -0.009745
-0.014378 -0.031930 -0.009334
-0.012115 -0.056846 0.008220
-0.012639 -0.081814 0.000310
-0.012757 -0.106689 -0.004126
-0.010902 -0.130076 -0.000135
-0.007004 -0.156220 0.007117
-0.003854 -0.184118 0.014271
-0.001294 -0.212650 0.021181
0.006520 -0.248954 0.031056
0.015494 -0.290481 0.061118
0.013479 -0.336066 0.057243
0.008916 -0.387081 0.034205
0.007508 -0.441045 0.026348
0.015894 -0.491728 0.039688
0.023390 -0.545630 0.069741

```

Figura 42 Arxiu de sortida del programa.

Com podem observar a la Figura 42 la primera fila es reserva per a guardar el temps total utilitzat per la CPU (requadre vermell). El nostre programa mesura el temps que tarda en llegir un nou *frame* i realitzar l'estimació de moviment amb l'anterior. Es realitza un sumatori de tots aquests temps per a poder saber el temps total emprat. En la imatge d'exemple (Figura 42), es pot veure que el programa ha tardat 182200 Milisegons en realitzar tota l'estimació de moviment del robot.

L'odometria es guarda a partir de la segona fila, on cada columna correspon a cada component de la posició absoluta del robot (requadre verd). Es guardaran d'esquerra a dreta els components X, Y i θ respectivament.

Ja que el nostre programa empra diversos paràmetres els quals haurem de donar valors i realitzar proves amb les mateixes seqüències de vídeo. És necessari diferenciar els arxius creats pel nostre programa.

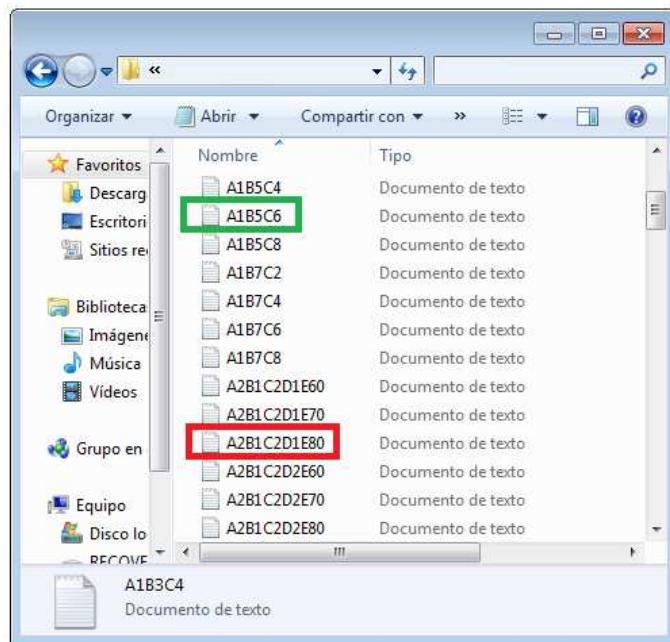


Figura 43 Noms dels fitxers de sortida.

Aquesta diferenciació es realitzarà mitjançant el nom de l'arxiu creat pel programa (Figura 43). Es nomenarà cada prova amb els valors seleccionats per l'usuari dels paràmetres. Aquests paràmetres i els seus valors els explicarem més endavant en l'apartat de proves.

Depèn de l'elecció de l'usuari hi haurà mes o menys paràmetres a avaluar, ja que RANSAC (requadre vermell) consta de més paràmetres que Mínims Quadrats (requadre verd). Per tant el nom de l'arxiu podrà ser més curt o més llarg. Els paràmetres estaran anomenats amb lletres, seguides per el valor establert per l'usuari. Els possibles noms seran els següents:

Mínims Quadrats i RANSAC.

- **A:** Paràmetre “selec3” que selecciona la utilització de Mínims Quadrats o RANSAC.
- **B:** Paràmetre “Xframe” que permet a l'usuari seleccionar cada quants *frames* volem analitzar.
- **C:** Paràmetre “numMin” que selecciona la proporció del llindar de FLANN.

RANSAC.

- **D:** Paràmetre “optionRansac” que permet seleccionar l'opció quan RANSAC no troba un model bo.
- **E:** Paràmetre “minConsensus” que estableix el percentatge de *matching* bons per a establir un bon model.

L'arxiu de sortida es guardarà automàticament en el mateix directori on es troben les imatges dels *frames* del vídeo de prova. L'arxiu es trobarà dins una carpeta anomenada “ExpOdo”.

6. Configuració del programa.

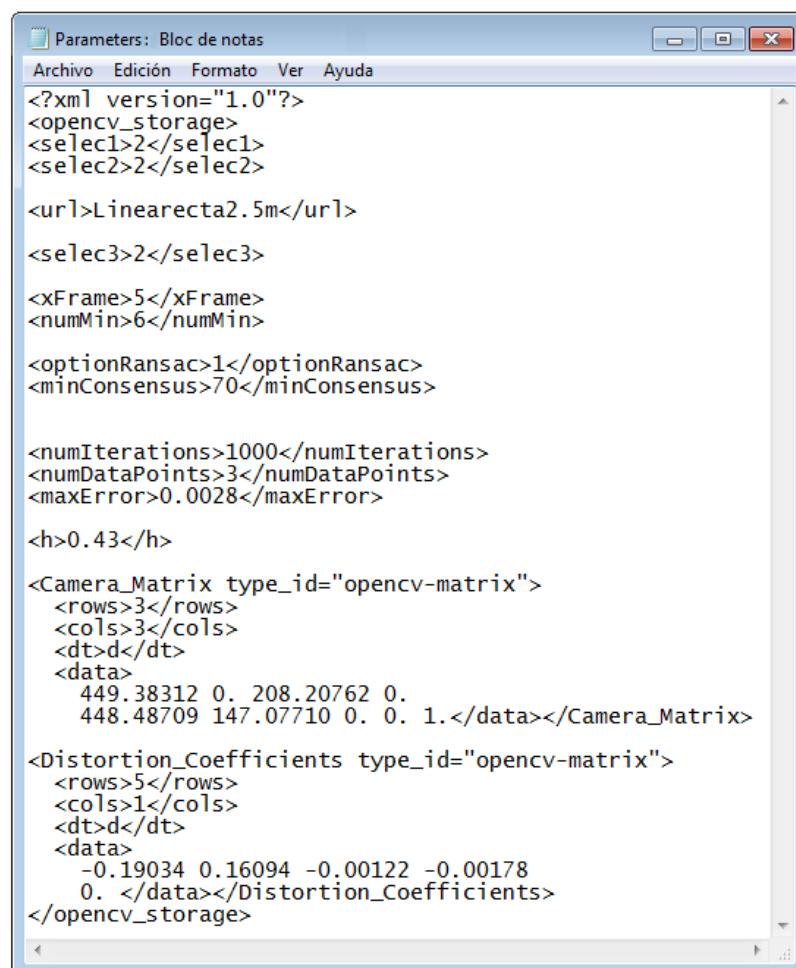
Aquesta part del programa representa l'apartat 1 de l'esquema general del programa (Figura 25).

Degut a la parametrització del nostre programa és necessària una configuració inicial del programa. També hi ha aspectes del programa com la selecció de la carpeta on volem llegir les imatges o els paràmetres de la càmera que necessiten ser configurats.

Per aquesta funció el nostre programa utilitza un fitxer on s'han d'indicar tots els paràmetres necessaris per al correcte funcionament del programa.

6.1 Arxiu d'entrada.

L'arxiu d'entrada es compon d'un arxiu del tipus ".xml" que conté tots els valors dels paràmetres importants necessaris per al funcionament del programa. OpenCV ens permet llegir els valors d'aquests paràmetres dins aquest fitxer de manera còmoda. El contingut de l'interior de l'arxiu d'entrada és el següent.



The screenshot shows a Windows Notepad window titled "Parameters: Bloc de notas". The content is an XML configuration file:

```
<?xml version="1.0"?>
<opencv_storage>
<selec1>2</selec1>
<selec2>2</selec2>

<url>Linearecta2.5m</url>

<selec3>2</selec3>

<xFrame>5</xFrame>
<numMin>6</numMin>

<optionRansac>1</optionRansac>
<minConsensus>70</minConsensus>

<numIterations>1000</numIterations>
<numDataPoints>3</numDataPoints>
<maxError>0.0028</maxError>

<h>0.43</h>

<Camera_Matrix type_id="opencv-matrix">
<rows>3</rows>
<cols>3</cols>
<dt>d</dt>
<data>
    449.38312 0. 208.20762 0.
    448.48709 147.07710 0. 0. 1.</data></Camera_Matrix>

<Distortion_Coefficients type_id="opencv-matrix">
<rows>5</rows>
<cols>1</cols>
<dt>d</dt>
<data>
    -0.19034 0.16094 -0.00122 -0.00178
    0. </data></Distortion_Coefficients>
</opencv_storage>
```

Figura 44 Arxiu d'entrada al programa.

Com es pot observar a la Figura 44 hi ha incorporats tots els paràmetres que anteriorment han estat explicats. Com també algunes que no s'han anomenat. és el cas de “selec1” i “selec2”. Aquests paràmetres permeten seleccionar entre alguns dels diferents detectors i descriptors que incorpora OpenCV a les seves llibreries com per exemple SURF i FAST. A priori aquests paràmetres no seran d'utilitat ja que en aquest projecte només s'emprarà SIFT.

L'arxiu també incorpora el nom de la carpeta on volem llegir les imatges. Aquest paràmetre s'anomena “url”, on dins l'espai situat entre el inici de la variable i el final de la mateixa s'indica el nom de la carpeta. En el cas de la imatge d'exemple el nom de la carpeta és “Linearecta2.5m”.

Tots els paràmetres explicats al llarg del document estan presents dins l'arxiu: xFrame, numMin, optionRansac, minConsensus, numIterations, numDataPoints, maxError. Aquests paràmetres es fixaran cada vegada que voldrem realitzar proves com els valors que es veuen a la Figura 44.

A la part final d'aquest arxiu es troben les condicions en les quals han estat realitzades les proves. Aquestes condicions són dues: altura de la càmera i els paràmetres de la càmera.

La altura a la qual la càmera ha realitzat les imatges es troba fixat per el paràmetre “h”. Ja que aquest paràmetre serà constant durant tot el vídeo no el necessitarem canviar durant l'estimació de moviment.

Els paràmetres intrínsecos de la càmera com els paràmetres de distorsió estan guardats com matrius. Aquestes matrius s'anomenen “Camera_Matrix” i “Distortion_Coefficients”. Per tant es guardarà de manera diferent a la resta de paràmetres. S'haurà d'indicar que el paràmetre és del tipus matriu i les files-columnes que la componen.

7. Interfície de sortida.

La interfície de sortida permet visualitzar a l'usuari el resultat final de l'odometria. Aquest apartat s'ha realitzar amb el programa MATLAB, mostrant per pantalla tot el recorregut realitzat pel nostre robot.

El nostre programa de MATLAB llegirà l'arxiu de sortida del nostre programa. Posteriorment imprimirà totes les posicions com també algunes de les orientacions del nostre robot dins el recorregut (Figura 45).

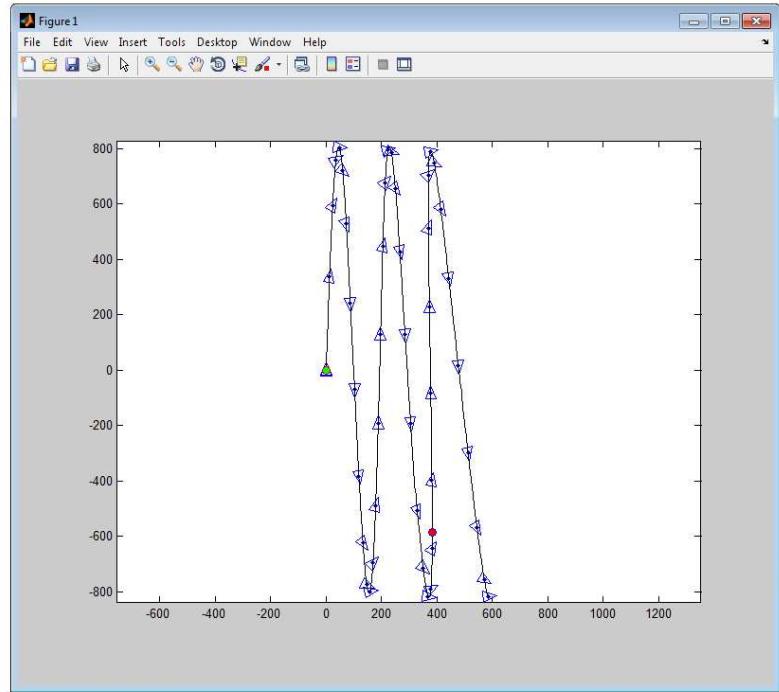


Figura 45 Interfície amb l'usuari: Odometria.

Per a facilitar la visualització de l'odometria, el programa ens superposarà alguns símbols damunt les posicions del robot.

- Un punt verd es situarà a la posició inicial (0,0) per a la seva ràpida localització.
- Un punt vermell realitzarà un recorregut virtual per totes les posicions simulant el moviment del robot. Finalment quedarà situat a la posició final de l'odometria.
- Cada cert nombre de posicions de l'odometria s'imprimirà un triangle indicant l'orientació en aquella posició del robot donant una idea general dels moviments realitzats.

Capítol 5. Proves experimentals

En aquest capítol evaluarem experimentalment els resultats de les localitzacions estimades pel nostre programa. Es definirà la manera de mesurar l'error de les diferents proves a realitzar. També es definiran diferents tipus d'experiments com també els vídeos a gravar per a aconseguir realitzar els experiments desitjats. Seleccionarem els paràmetres que fixarem o els que seran evaluats a les proves. Al final del capítol s'analitzaran les dades quantitatives extretes de les proves realitzades i per tant poder obtenir una conclusió amb uns paràmetres que podrem definir com a òptims. Haurem de tenir en compte que només hem realitzat una prova per vídeo i per tant aquestes evaluacions seran simples, a mode d'exemple, i per això a vegades no es troben tendències clares.

1. Experiments i vídeos.

La primera passa de les proves experimentals consisteix en definir els tipus d'experiment que voldrem realitzar. Una vegada definits els experiments podrem definir els vídeos que voldrem realitzar.

Per a poder avaluar el nostre odòmetre d'una manera més completa realitzarem diferents tipus d'experiments que posin a prova el seu funcionament de manera que es pugui elegir els paràmetres que millor s'adaptin.

Les proves que realitzarem seran les següents:

- **Moviment amb vibració:** La càmera sofrirà tremolor constantment degut al lliscament de les rodes que transporten la càmera per sobre el sòl rugós. Serà el tipus d'experiment que més proves realitzarem ja que posa més a prova el nostre programa.
 - **Moviment senzill:** Moviment que no posarà en molts de problemes el posicionament, com realitzar una línia recta sense molta velocitat.
 - **Moviment complet:** Moviments amb corbes, més velocitat, possibilitat d'ombres del mateix robot degut a canvis de direcció i distàncies més llargues.
- **Moviments sense vibració:** La càmera no sofrirà cap tipus de vibració deguda al sòl.
 - **Moviment senzill:** Moviment simple, amb una velocitat no necessàriament alta i sense problemes d'ombres.

Una vegada tenim definits els tipus d'experiments que volem realitzar definirem els vídeos que realitzarem. Els moviments més simples seran els menys estudiats mentre que els moviments complets seran els més repetits, ja que seran els que més errors ens poden causar.

Els vídeos els realitzarem amb un càmera que estarà fixada en una cadira com es veu a la Figura 46 simulant el moviment d'un robot. El sòl rugós i les rodes de plàstic dur fan l'efecte de vibració continu.



Figura 46 Instruments emprats per dur a terme les proves.

Per a realitzar les proves, hem situat al sòl marques amb les quals podrem saber amb exactitud on hem de dirigir la nostra càmera i poder tenir mesures reals del moviment. Els vídeos realitzats són els següents:

- **Vídeo 1:** Es pot classificar dins els moviments simples amb vibració. El robot es mou traçant una línia recta de 2,5 metres de longitud. Vídeo gravat a una velocitat baixa de moviment i sense problemes amb ombres.
- **Vídeo 2:** Es classifica dins els moviments complets amb vibració. El robot descriu un moviment semblant a un triangle equilàter de 2 metres de costat. La posició final coincideix amb la posició inicial del robot en el vídeo. Vídeo gravat a velocitat normal.
- **Vídeo 3:** Es classifica dins els moviments complets amb vibració. En aquest cas el robot descriu un moviment semblant a la forma del numero vuit. Aproximadament el diàmetre dels cercles és de 1,25 metres. La darrera posició del robot coincideix amb la posició inicial. Vídeo gravat a velocitat normal.
- **Vídeo 4:** Es classifica dins els moviments complets amb vibració. En aquest cas el robot descriu el moviment d'un rectangle de 2 x 3,38 metres. El robot realitzarà tres quadrats idèntics passant per les mateixes posicions i acabant en la posició inicial. Vídeo gravat a una velocitat més elevada i amb problemes d'ombres.
- **Vídeo 5:** Es classifica dins els moviments simples sense vibració. El robot descriurà un moviment circular de 0,52 metres de radi. Es rotarà sobre si mateixa la cadira utilitzada per a gravar el vídeo a fi d'evitar la vibració de la càmera. En aquest cas el robot realitzarà el mateix moviment 3 vegades passant per les mateixes posicions i

acabant a la posició de partida. Vídeo gravat a velocitat normal i sense problemes d'ombres.

2. Mesures de l'error.

Per a cada prova realitzada calcularem l'error que ha anat acumulant el nostre odòmetre. En aquest apartat fixarem la manera en la qual mesurarem aquest error.

- **Posició final en l'odometria.**

Principalment es calcularà l'error mitjançant la darrera posició de l'odometria. Aquest error s'obtindrà calculant la distància que hi ha entre la posició final real i la posició final estimada. L'error comés per la darrera posició la calcularem emprant l'error absolut i l'error relatiu.

- **Error absolut:** Distància que hi ha entre el punt final real i entre el punt final estimat.
- **Error relatiu:** Percentatge d'error que hi ha entre l'error absolut i la distància total estimada a l'odometria. És a dir, error absolut partit la distància recorreguda per 100.

En el cas de les proves realitzades en el vídeo 4 l'error es calcularà per a cada quadrat realitzat per la càmera, que en total realitzarà 3 voltes al mateix. Per a calcular l'error total, es realitzarà la mitjana dels tres errors.

- **Distància recorreguda pel robot.**

Calcularem la distància recorreguda total pel robot mitjançant les posicions que s'hauran estimat pels mètodes de localització i la compararem amb la real. Amb el càlcul d'aquestes distàncies podem esbrinar l'error acumulat durant tot el trajecte. Aquest càlcul es realitzarà mitjançant la suma de les distàncies que hi ha entre cada posició de l'odometria.

Realitzarem les mitjanes dels resultats agrupats per la separació de *frames* de cada vídeo. Es calcularà l'error relatiu comparant la distància total real recorreguda i la distància mitja estimada de cada grup com també la desviació típica d'aquestes mitjanes. En el cas de RANSAC només explicarem els resultats més rellevants. Tots els resultats obtinguts amb RANSAC són als apèndixos . En aquest cas hem seleccionat de entre els tres percentatges de consens la del 70% per contenir els errors més petits.

3. Problemes amb el càlcul d'errors.

Per a calcular l'error que hem comés a l'odometria de forma precisa es necessita utilitzar un recorregut real anomenat *Ground truth* (Figura 47). Aquest *Ground truth* és el recorregut real sense error per on el nostre robot ha passat, de manera que s'haurien de comparar

totes les posicions de l'odometria estimada amb les corresponents posicions reals de *Ground truth*.

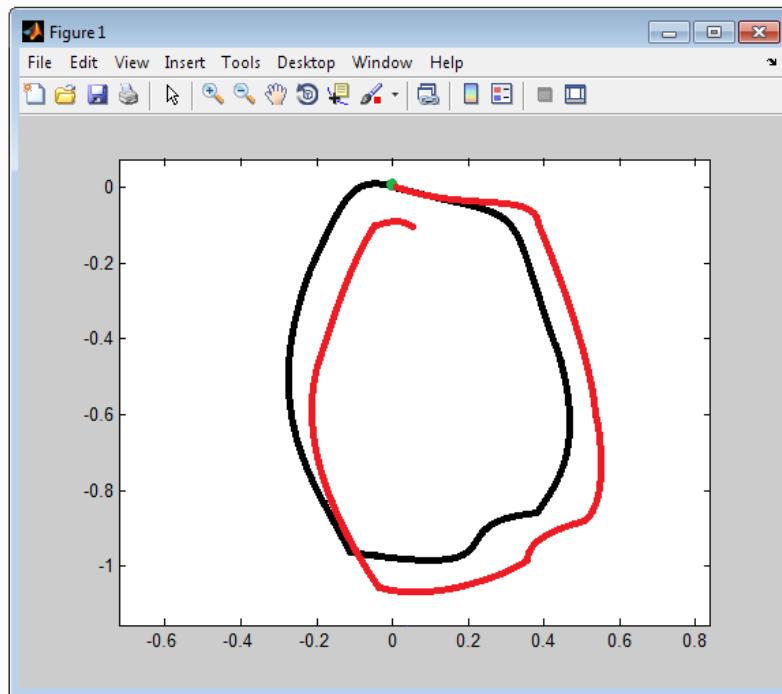


Figura 47 Exemple de Ground Truth (recorregut negre) i odometria (recorregut vermell).

El recorregut negre és el recorregut real seguit pel robot (*Ground truth*) mentre que el recorregut vermell és el recorregut estimat per l'dometre. El punt verd indica el inici i el final del recorregut.

El problema és que obtenir el *Ground truth* és complicat, sobretot en moviments curts ja que una de les maneres de obtenir-lo és utilitzant GPS. El marge d'error del GPS és aproximadament de 1 a 3 metres i els nostres recorreguts no superen els 4 metres de longitud. Per tant no ens serveix ja que l'error és massa gran.

Nosaltres no tenim el *Ground truth* dels nostres experiments per tant haurem d'emprar un mètode menys eficaç. Compararem la posició final de la nostra odometria amb el punt exacte on ha acabat el moviment del robot respecte del inici. Per tant només podrem comparar un punt. La comparació d'un únic punt no dóna un càlcul d'error tan precís com amb *Ground truth* i pot donar falsos errors. Per exemple una mala odometria pot donar per casualitat que el punt final d'aquesta està molt a prop del punt exacte final del robot i per tant ens donaria un error molt baix com en el cas de la Figura 48.

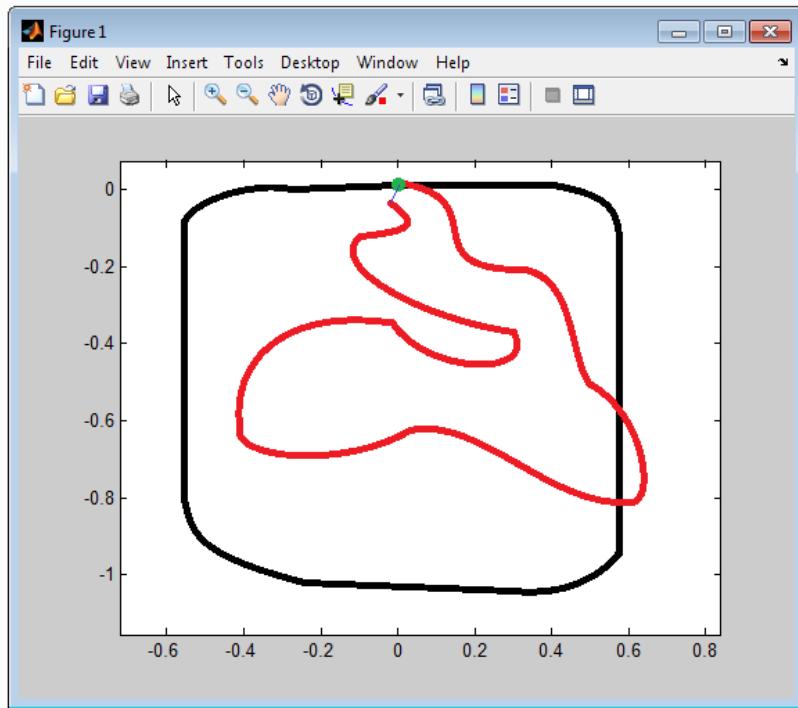


Figura 48 Exemple mala odometria amb baix error.

Com a suport de l'error final també calcularem l'error en distància total recorreguda. L'error en distància té el desavantatge de que amb la tremolor de la càmera pel sòl rugós augmenti l'error comés per l'odometre per acumulació. Per això, un dels paràmetres a avaluar és el nombre de *frames* amb els que llegim les imatges. Amb aquest paràmetre s'intenta minimitzar aquest efecte i per tant l'error.

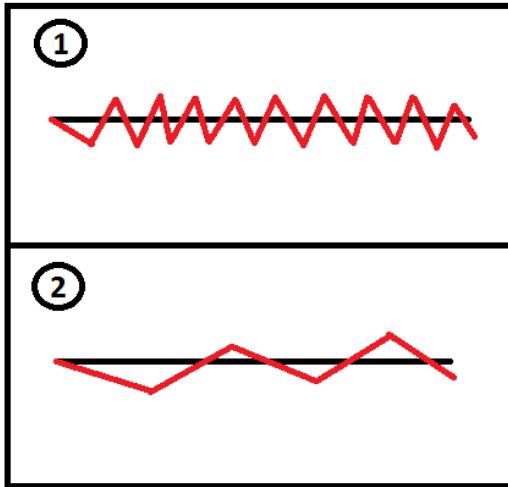


Figura 49 Odometria amb vibració de la càmera.

A la Figura 49 es pot veure un exemple de com es modifica l'odometria augmentant el nombre dels *frames* els qual volem llegir. Al quadre numero 1 es veu l'odometria (vermell) respecte al *Ground truth* (negre) amb un nombre baix de diferència entre *frames* llegits i per tant una major distància en trajecte del robot per acumulació. Mentre que al

numero 2 l'error en distància total ha disminuït augmentant el nombre de separació entre *frames* llegits.

4. Paràmetres avaluats.

La principal finalitat d'aquest apartat és la d'analitzar les dades quantitatives de les proves i esbrinar quins valors dels paràmetres són els òptims per al nostre odòmetre. Per tant haurem de seleccionar els paràmetres més importants que ens afectaran en gran mesura als resultats finals de l'odometria.

Degut al gran nombre de paràmetres que tenim i a la quantitat de vídeos realitzats, haurem de fixar els paràmetres que menys ens interessin. Aquesta mesura ens servirà per alleujar en un gran nombre de proves i poder comprovar més valors de paràmetres més importants. Els valor dels paràmetres fixats serà el mateix durant totes les proves.

- **Paràmetres fixats.**

Els paràmetres fixats estan tots relacionats amb RANSAC. S'han seleccionat aquests paràmetres ja que són els que la seva variació influeix menys dins el resultat final de l'odometria.

A continuació s'explicaran els paràmetres fixats amb els valors que emprarem a les proves.

- **NumIterations = 1000:** Aquest paràmetre ens indica quantes vegades realitzarem el bucle general de RANSAC. Tindrà un nombre considerablement alt ja que RANSAC és un algorisme amb un important component aleatori i per tant necessitem moltes combinacions per obtenir models bons en situacions difícils. Hem comprovat que unes 1000 iteracions són suficients per a trobar solucions raonables en un temps reduït.
- **NumDatapoints = 3:** Paràmetre que ens indica quants *keypoints* seran escollits aleatoriament per a calcular per primera vegada Mínims Quadrats a RANSAC. Aquest nombre té un mínim de 2 i no és recomanable realitzar-lo amb un nombre molt superior ja que aquest pas està especialment realitzat per evitar *matchings* dolents. Hem comprovat que amb un nombre de 3 és més que suficient per a trobar solucions bones.
- **MaxError = 0,0028:** Paràmetre que ens indica el marge el qual els *keypoints* rotats amb el model inicial coincideixen amb els *keypoints* de la segona imatge. Aquest paràmetre necessita ser baix per a una precisió més alta en el tractament dels *matchings* dolents. En el nostre cas hem trobat experimentalment que el radi on els *keypoints* són bons és de 0,0028 m.

- **Paràmetres evaluats.**

Són els paràmetres que seran variables durant totes les proves. Fixarem un rang màxim i mínim dels valors per a cada paràmetre. Dins aquest rang de valors fixarem els valors que

donarem als paràmetres per a realitzar les proves i posteriorment analitzar les dades obtingudes amb les possibles configuracions.

A continuació s'explica quins són els paràmetres a avaluar i els valors que utilitzarem a les proves empíriques.

- **Selec3:** Valors utilitzats a les proves: 1, 2. Paràmetre que selecciona si utilitzarem Mínims Quadrats o RANSAC. Aquest paràmetre només ens indicarà el sistema utilitzat per a l'estimació de l'odometria. A l'opció 1 el programa utilitzarà únicament Mínims Quadrats, mentre que l'opció 2 emprarà RANSAC.
- **Xframe:** Valors utilitzats a les proves: 1, 3, 5, 7. Paràmetre que ens indica cada quants *frames* llegirem les imatges. Hem elegit un interval de 1 a 7 *frames* escollint quatre valors dins el interval. Hem establert el límit de 7 *frames* perquè a partir d'aquest valor la pèrdua de informació entre 7 *frames* ja és significativa.
- **NumMin:** Valors utilitzats a les proves: 2, 4, 6, 8. és el paràmetre que ens indica el llindar mínim de distància entre *matchings* de FLANN. Hem elegit un interval de 2 a 8 escollint quatre valors dins el interval. El màxim hem establert que sigui 8 ja que un filtre menys estricte poden entrar molts *matchings* dolents i per tant augmentar molt l'error dins l'odometria final.
- **OptionRansac:** Valors utilitzats a les proves: 1, 2. és el paràmetre que ens indica l'opció que es escollida quan RANSAC no ha trobat cap model bo. L'opció 1 RANSAC utilitzarà l'odometria anterior com a actual i a l'opció 2 utilitzarà Mínims Quadrats.
- **MinConsensus:** Valors utilitzats a les proves: 70, 80, 90. És el paràmetre que ens indica el percentatge de *matchings* bons que necessitem per a que RANSAC consideri que un model sigui bo. Hem seleccionat un rang de valors de 70% a 90% elegint tres valors. Hem considerat amb proves prèvies que un percentatge inferior el filtre és massa poc restrictiu amb altes probabilitats de seleccionar models dolents com a bons. Un percentatge superior a 90% les probabilitats de trobar un model bo són més baixes.

5. Resultat de les proves.

En aquest apartat s'exposen els resultats de les proves realitzades als diferents vídeos. Es mostrerà un exemple visual de cada prova. També es mostraran els errors commesos per l'odometre mitjançant unes taules que indiquen el valor dels paràmetres per a cada error. Tant per l'error en punt final d'odometria com per l'error en distància recorreguda. Completant la informació de les taules es mostren unes gràfiques dels resultats de les taules. També s'adjuntarà una taula amb els temps que s'han emprat per a realitzar les proves.

5.1 Resultats obtinguts del vídeo 1.

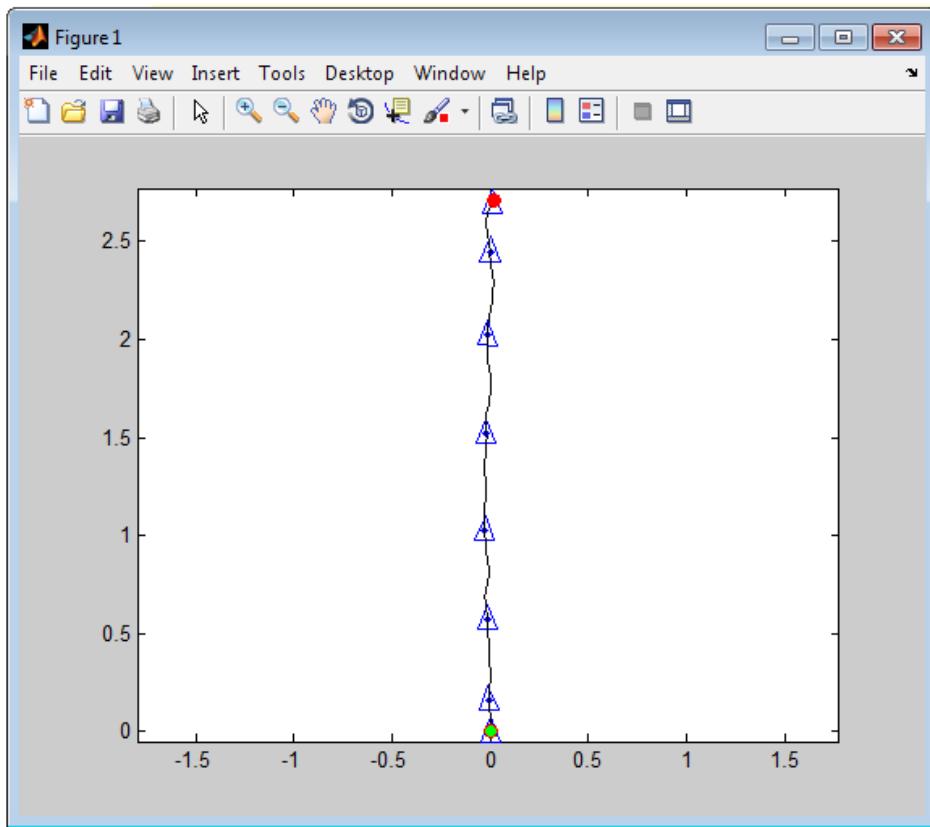


Figura 50 Resultat del vídeo 1. El triangle mostra l'orientació del robot durant el trajecte i els punts verds i vermell indiquen el inici i el final del trajecte respectivament.

En aquest apartat es mostraran les proves experimentals realitzades pel recorregut 1 o vídeo 1. A la Figura 50 es mostra un exemple d'odometria calculada per aquest recorregut.

A continuació s'exposaran els resultats més significatius de les proves experimentals del Vídeo 1 tant de Mínims Quadrats com de RANSAC. També es mostrerà una taula del temps emprat per a l'execució del programa.

- **Error en posició final de l'odometria (Mínims Quadrats).**

En aquesta secció es mostren els resultats de les proves obtingudes amb Mínims Quadrats per al vídeo 1 (Taula 3).

Xframe numMin	1		3		5		7	
	E.Relatiu	E.Absolut	E.Relatiu	E.Absolut	E.Relatiu	E.Absolut	E.Relatiu	E.Absolut
2	8,4 %	0,211 m	8,3 %	0,207 m	8,4 %	0,210 m	8,4 %	0,209 m
4	8,4 %	0,211 m	8,4 %	0,210 m	8,4 %	0,210 m	8,4 %	0,211 m
6	8,5 %	0,212 m	8,4 %	0,211 m	8,5 %	0,212 m	8,3 %	0,206 m
8	8,4 %	0,210 m	7,9 %	0,198 m	4,8 %	0,120 m	6,7 %	0,167 m

Taula 3 Taula d'errors de Mínims Quadrats del vídeo 1. Es mostra tant l'error relatiu (vermell) com l'error absolut (gris).

Podem veure les variables NumMin i Xframe amb els distints valors que hem emprat. Dins el quadre gris es marca l'error relatiu comés en metres i en el quadre vermell es marca l'error absolut. A la Figura 51 es mostra la corresponent gràfica de la Taula 3.

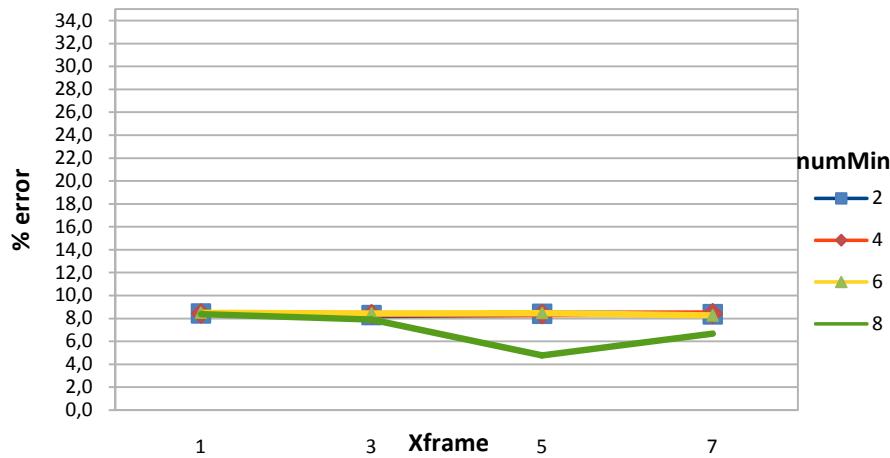


Figura 51 Gràfica de resultats de Mínims Quadrats del vídeo 1.

Com es pot observar a la gràfica, l'error disminueix a mesura que augmentem la separació entre *frames* fins a 5 i torna a augmentar a 7, amb l'opció de numMin = 8. L'error més baix amb un 4,8% es troba a la selecció: **Xframe = 5 i numMin = 8**.

- **Error en posició final de l'odometria (RANSAC).**

En aquesta secció es mostrerà l'error absolut comés per l'odometre amb l'opció seleccionada RANSAC en format tabular (Taula 4). La taula d'errors completa de RANSAC és massa gran i per tant només es mostrerà a l'apèndix "Error MinsQua". Hem seleccionat una porció de la taula que creiem té els millors resultats.

Xframe minCons					
	1	3	5	7	
numMin=8 SELEC=1	60%	8,6 %	8,5 %	8,5 %	4,9 %
	70%	8,6 %	8,5 %	8,5 %	4,1 %
	80%	8,6 %	8,5 %	7,3 %	4,3 %

Taula 4 Taula d'errors de RANSAC del vídeo 1. A la taula es mostra l'error relatiu per a diferents valors de Xframe, minConsensus i els valors fixos de numMin i SELEC (quadre gris puntejat).

A la figura 52 es representa la gràfica de la taula d'errors de RANSAC (Taula 4).

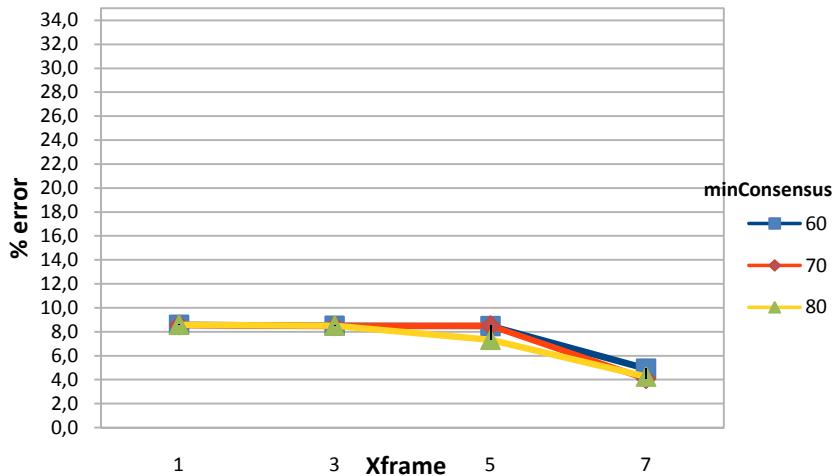


Figura 52 Gràfica d'errors RANSAC del vídeo 1.

Els tres percentatges de consens no es diferencien gaire amb l'error comés. Augmentant el nombre de separació entre *frames* l'error disminueix en tots els resultats a prop d'un 4%. L'error més baix amb un 4,1% es troba a la configuració següent: **Xframe = 7, numMin = 8, SELEC = 1 i minConsensus = 70%**.

- Error en distància recorreguda pel robot.**

Dins aquest apartat es mostrerà la distància acumulada per l'odometre durant tot el recorregut en format tabular. L'error es calcula sumant totes les distàncies que hi ha entre totes les posicions estimades per l'odometre i comparant-les amb la distància total teòrica. Mostrarem els resultats obtinguts amb Mínims Quadrats (Taula 5) i RANSAC (Taula 6). Per a RANSAC hem emprat un valor de 70% per al paràmetre minConsensus perquè és un valor que està entre un valor alt (80%) i baix (60%) de percentatge de consens. Realitzar les proves amb totes les possibilitats equivaldria a una inversió temporal molt gran.

L. Recta 2.5m	1	3	5	7	Xframe numMin
Distància: 2,5 m	2,792 m	2,767 m	2,752 m	2,751 m	2
	2,794 m	2,769 m	2,753 m	2,754 m	4
	2,797 m	2,770 m	2,755 m	2,751 m	6
	2,795 m	2,757 m	2,665 m	2,394 m	8
	2,794 m	2,766 m	2,731 m	2,662 m	Mitja
	0,003 m	0,010 m	0,076 m	0,310 m	Desviació típica
	11,8 %	10,6 %	9,2 %	6,5 %	Error %

Taula 5 Error acumulat utilitzant Mínims Quadrats del vídeo 1.

L. Recta 2.5m	1	3	5	7	Xframe numMin
Distància: 2,5 m	2,793 m	2,767 m	2,751 m	2,751 m	2
	2,795 m	2,769 m	2,753 m	2,754 m	4
	2,797 m	2,771 m	2,755 m	2,756 m	6
	2,799 m	2,772 m	2,756 m	2,517 m	8
	2,796 m	2,770 m	2,754 m	2,695 m	Mitja
	0,004 m	0,004 m	0,004 m	0,206 m	Desviació típica
	11,7%	10,8 %	10,1 %	8,9 %	Error %

Taula 6 Error acumulat utilitzant RANSAC (70%, SELEC=1) del vídeo 1.

A les taules es mostren les distàncies acumulades per a diferents valors de Xframe i numMin. En el quadre marron puntejat es mostren els metres teòrics realitzats pel robot, on es comparen amb la mitja (groc) i es treu l'error relatiu (vermell). També es pot veure la desviació típica de les dades (groc).

A Mínims Quadrats (Taula 5) l'error relatiu va disminuint a mesura que augmenta la separació entre frames com s'ha explicat a l'apartat 3. Al resultat de **Xframe = 7** i **numMin = 8** es pot veure que la distància total calculada és molt menor que la distància total teòrica, això indica que l'Odometria ha estat dolenta. La desviació típica en aquest cas augmenta considerablement al tenir aquest resultat.

A RANSAC (Taula 6) l'error relatiu va disminuint a mesura que augmenta la separació entre frames. La desviació típica augmenta en **Xframe = 7** ja que la distància acumulada a **numMin = 8** baixa considerablement respecte les altres proves. Podria ser un cas de mala odometria.

- **Temps d'anàlisi d'imatges.**

En aquest apartat es mostraran els temps d'anàlisi del vídeo 1. En les següents taules s'ha calculat les mitjanes d'aquests temps estimats tant per Mínims Quadrats (Taula 7) com per les proves agafades de RANSAC (Taula 8). En el cas de RANSAC hem seleccionat els mateixos valors que en el càlcul de l'error en distància recorreguda.

L. Recta 2.5m	1	3	5	7	Xframe numMin
Temps: 18 s	113,7 s	43,3 s	27,1 s	22,1 s	2
	107,5 s	42,1 s	25,8 s	22,2 s	4
	116,9 s	38,5 s	24,5 s	17,8 s	6
	127,2 s	43,2 s	28,3 s	18,4 s	8
	116,3 s	41,8 s	26,4 s	20,1 s	Mitja

Taula 7 Taula de temps estimat amb Mínims Quadrats del vídeo 1.

L. Recta 2.5m	1	3	5	7	Xframe numMin
Temps: 18 s	132,6 s	43,9 s	25,7 s	19,2 s	2
	140,5 s	48,8 s	30,3 s	23,1 s	4
	153,2 s	48,9 s	32,7 s	22,1 s	6
	156,2 s	49,5 s	36,1 s	22,5 s	8
	145,7 s	47,8 s	31,2 s	21,7 s	Mitja

Taula 8 Taula de temps estimat amb RANSAC (70%, SELEC=1) del vídeo 1.

A les taules es mostren els temps mesurats per als diferents valors de Xframe i numMin. En el cas de RANSAC amb els dos paràmetres fixos nomenats al peu de la taula. En el quadre marró puntejat es mostra el temps real que ha durat el vídeo.

Com es pot observar, el temps d'anàlisi disminueix considerablement a mesura que augmenta la separació entre *frames* ja que hem de processar menys imatges. RANSAC és un poc més lent que Mínims Quadrats ja que el seu algoritme és més complex. Aquesta observació serà una constant durant totes les proves realitzades.

5.2 Resultats obtinguts del vídeo 2.

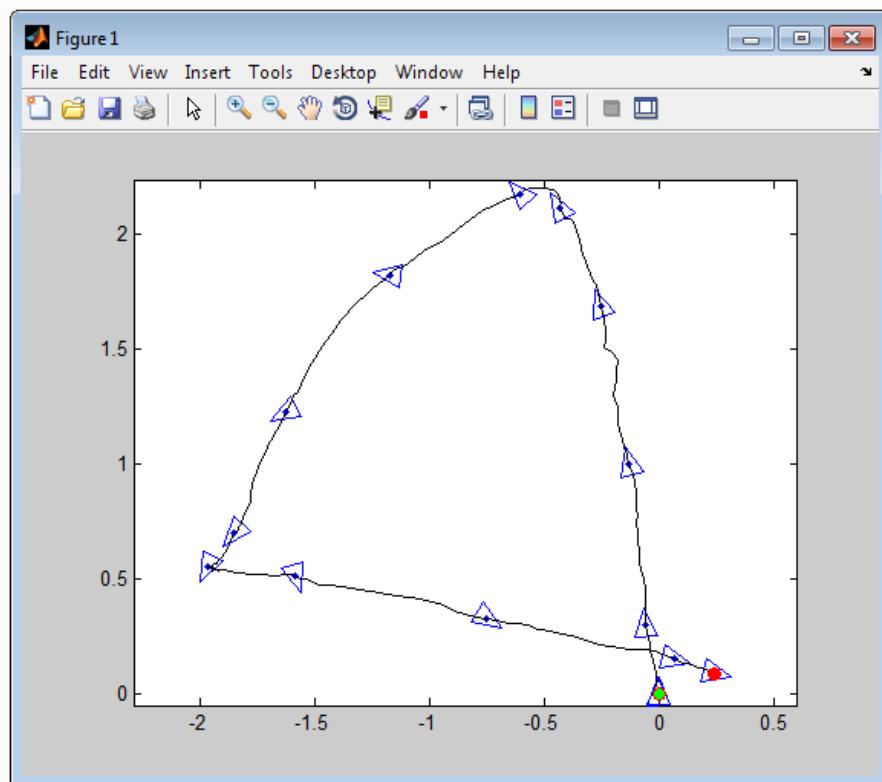


Figura 53 Resultat del vídeo 2. El triangle mostra l'orientació del robot durant el trajecte i els punts verds i vermell indiquen el inici i el final del trajecte respectivament.

En aquest apartat es mostraran les proves experimentals realitzades pel recorregut 2 o vídeo 2. A la Figura 53 es mostra un exemple d'odometria calculada per aquest recorregut.

A continuació s'exposaran els resultats més significatius de les proves experimentals del Vídeo 2 tant de Mínims Quadrats com de RANSAC. També es mostrarà una taula del temps emprat per a l'execució del programa.

- **Error en posició final de l'Odometria (Mínims Quadrats).**

En aquesta secció es mostren els resultats de les proves obtingudes amb Mínims Quadrats per al vídeo 2 (Taula 9).

Xframe numMin	1		3		5		7	
	E.Relatiu	E.Absolut	E.Relatiu	E.Absolut	E.Relatiu	E.Absolut	E.Relatiu	E.Absolut
2	4,0 %	0,239 m	4,3 %	0,256 m	5,1 %	0,304 m	5,7 %	0,343 m
4	4,7 %	0,282 m	4,5 %	0,270 m	13,7 %	0,825 m	10,3 %	0,617 m
6	4,6 %	0,277 m	16,1 %	0,968 m	3,0 %	0,178 m	13,7 %	0,820 m
8	3,1 %	0,188 m	21,1 %	1,267 m	11,1 %	0,666 m	15,0 %	0,900 m

Taula 9 Taula d'errors de Mínims Quadrats del vídeo 2. Es mostra tant l'error relatiu (vermell) com l'error absolut (gris).

Podem veure les variables NumMin i Xframe amb els distints valors que hem emprat. Dins el quadre gris es marca l'error relatiu comés en metres i en el quadre vermell es marca l'error absolut. A la Figura 54 es mostra la corresponent gràfica de la Taula 9.

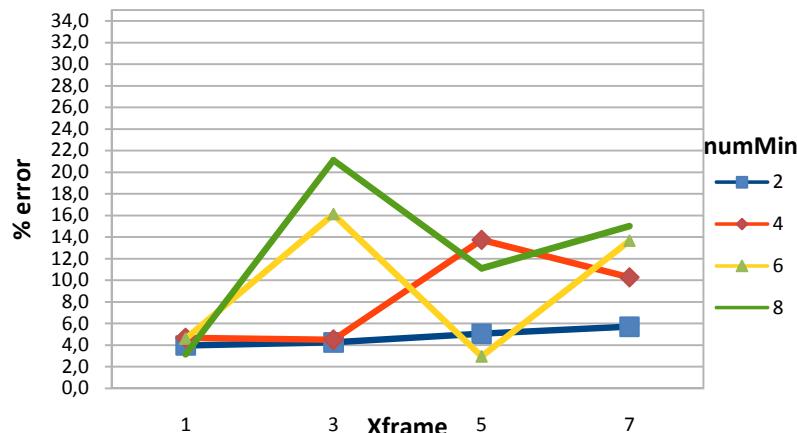


Figura 54 Gràfica de resultats de Mínims Quadrats del vídeo 2.

En aquest cas no es troba una tendència clara ja que només s'ha realitzat una vegada com hem esmentat. L'error augmenta al augmentar la distància entre frames, sobretot augmenta quan l'opció de numMin és elevada com 6 i 8. L'error més baix amb un **3,1%** es troba a l'opció: **Xframe = 1** i **numMin = 8**.

- **Error en posició final de l'odometria (RANSAC).**

En aquesta secció es mostrarà l'error absolut comés per l'odometre amb l'opció seleccionada RANSAC en format tabular (Taula 10).

	Xframe minCons	1	3	5	7
numMin=2	60	3,1 %	2,6 %	4,3 %	5,0 %
SELEC= 2	70	2,7 %	2,9 %	3,9 %	4,3 %
	80	3,0 %	3,7 %	4,7 %	4,7 %

Taula 10 Taula d'errors de RANSAC del vídeo 2. A la taula es mostra l'error relatiu per a diferents valors de Xframe, minConsensus i els valors fixos de numMin i SELEC (requadre gris puntejat).

A la figura 55 es representa la gràfica de la taula d'errors de RANSAC (Taula 10).

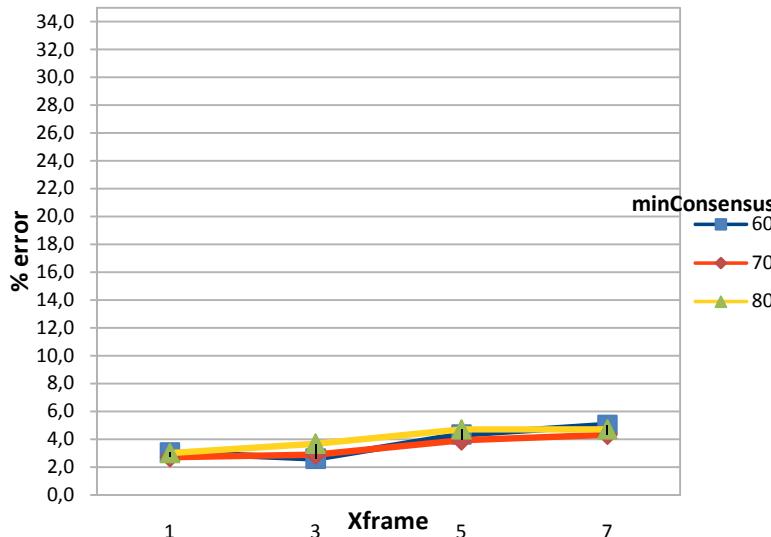


Figura 55 Gràfica de resultats de RANSAC del vídeo 2.

Com s'observa a la gràfica l'error augmenta al augmentar la separació entre frames. No hi ha gaire diferència entre percentatges de consens. L'error més baix amb un 2,7% es troba a l'opció: Xframe = 1, numMin = 2, selec = 2 i minConsensus = 70%.

- **Error en distància recorreguda pel robot.**

Dins aquest apartat es mostrerà la distància acumulada per l'odometre durant tot el recorregut en format tabular. Mostrarem els resultats obtinguts amb Mínims Quadrats (Taula 11) i RANSAC (Taula 12).

Triangle	1	3	5	7	Xframe numMin
Distància: 6 m	7,18 m	7,09 m	7,03 m	7,02 m	2
	7,18 m	7,05 m	6,82 m	6,54 m	4
	7,07 m	6,72 m	5,84 m	4,95 m	6
	6,53 m	5,70 m	4,31 m	3,16 m	8
	7,0 m	6,6 m	6,0 m	5,4 m	Mitja
	0,5 m	1,1 m	2,1 m	3,0 m	Desviació típica
	16,5 %	10,7 %	0,0 %	9,7 %	Error %

Taula 11 Error acumulat utilitzant Mínims Quadrats del vídeo 2.

Triangle	1	3	5	7	Xframe numMin
Distància: 6 m	7,18 m	7,09 m	7,03 m	7,02 m	2
	7,18 m	7,06 m	6,83 m	6,59 m	4
	7,10 m	6,80 m	5,99 m	5,12 m	6
	6,72 m	6,03 m	4,64 m	3,32 m	8
	7,0 m	6,7 m	6,1 m	5,5 m	Mitja
	0,4 m	0,9 m	1,9 m	2,9 m	Desviació típica
	17,4 %	12,4 %	2,0 %	8,1 %	Error %

Taula 12 Error acumulat utilitzant RANSAC (70%, SELEC=2) del vídeo 2.

A les taules es mostren les distàncies acumulades per a diferents valors de Xframe i numMin. En el quadre marron puntejat es mostren els metres teòrics realitzats pel robot, on es comparen amb la mitja (groc) i es treu l'error relatiu (vermell). També es pot veure la desviació típica de les dades (groc).

No es poden treure conclusions massa clares en ambdues proves, ja que degut a les errades comeses per l'dometre quan augmentem la distància entre frames amb numMin major a 2. Es pot veure la desviació típica com va augmentant amb la separació entre frames.

- **Temps d'anàlisi d'imatges.**

En aquest apartat es mostraran els temps d'anàlisi del vídeo 2. En les següents taules s'ha calculat les mitjanes d'aquests temps estimats tant per Mínims Quadrats (Taula 13) com per les proves agafades de RANSAC (Taula 14). En el cas de RANSAC hem seleccionat els mateixos valors que en el càlcul de l'error en distància recorreguda.

Triangle	1	3	5	7	Xframe numMin
Temps: 35 s	229 s	80 s	48 s	33 s	2
	219 s	85 s	46 s	36 s	4
	238 s	80 s	60 s	32 s	6
	250 s	87 s	54 s	33 s	8
	234 s	83 s	52 s	34 s	Mitja

Taula 13 Taula de temps estimat amb Mínims Quadrats del vídeo 2.

Triangle	1	3	5	7	Xframe numMin
Temps: 35 s	248 s	76 s	44 s	33 s	2
	263 s	86 s	53 s	36 s	4
	284 s	89 s	55 s	40 s	6
	287 s	93 s	56 s	39 s	8
	271 s	86 s	52 s	37 s	Mitja

Taula 14 Taula de temps estimat amb RANSAC (70%, SELEC=2) del vídeo 2.

A les taules es mostren els temps mesurats per als diferents valors de Xframe i numMin. En el cas de RANSAC amb els dos paràmetres fixos esmenats al peu de la taula. En el quadre marró puntejat es mostra el temps real que ha durat el vídeo.

El temps d'analisi disminueix a mesura que augmenta la separació entre *frames*. També es veu un augment en general del temps al augmentar el límit numMin ja que el programa ha de tractar amb més *features* emparellats per imatge.

5.3 Resultats obtinguts del vídeo 3.

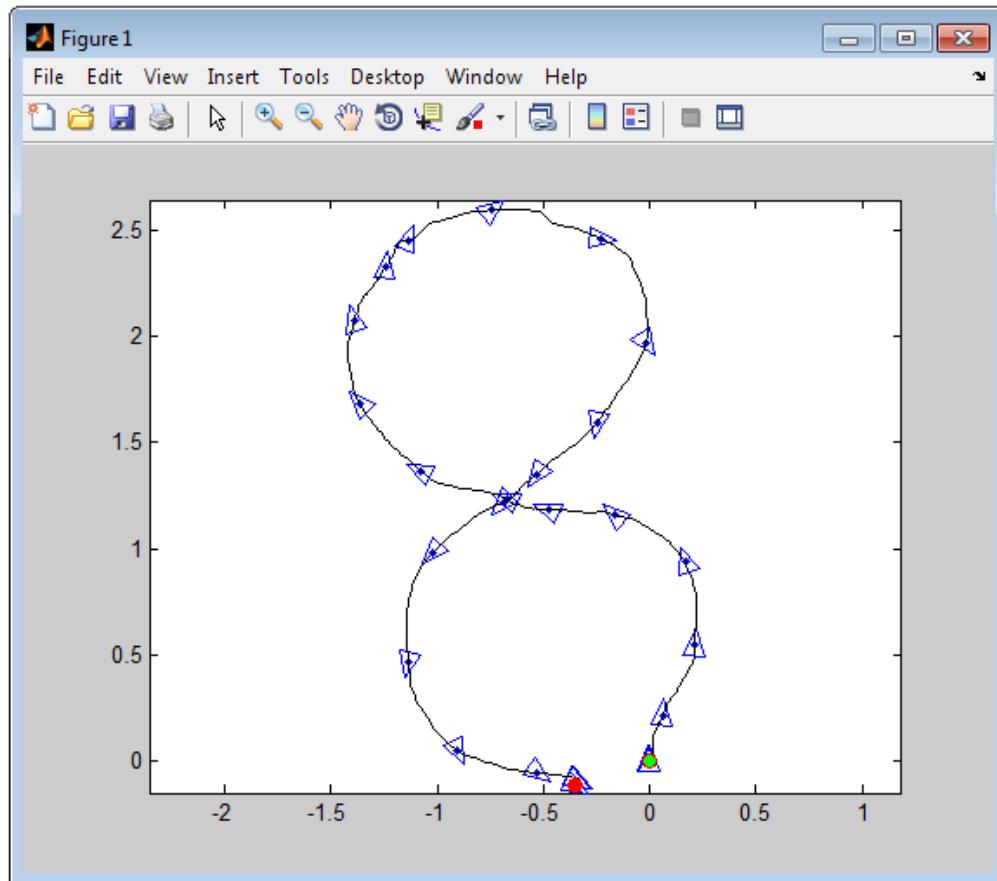


Figura 56 Resultat del vídeo 3. El triangle mostra l'orientació del robot durant el trajecte i els punts verds i vermells indiquen el inici i el final del trajecte respectivament.

En aquest apartat es mostraran les proves experimentals realitzades pel recorregut 3 o vídeo 3. A la Figura 56 es mostra un exemple d'odometria calculada per aquest recorregut.

A continuació s'exposaran els resultats més significatius de les proves experimentals del Vídeo 3 tant de Mínims Quadrats com de RANSAC. També es mostrarà una taula del temps emprat per a l'execució del programa.

- **Error en posició final de l'odometria (Mínims Quadrats).**

En aquesta secció es mostren els resultats de les proves obtingudes amb Mínims Quadrats per al vídeo 3 (Taula 15).

Xframe	1		3		5		7	
	numMin	E.Relatiu	E.Absolut	E.Relatiu	E.Absolut	E.Relatiu	E.Absolut	E.Relatiu
2	4,1 %	0,285 m	5,3 %	0,369 m	3,9 %	0,274 m	4,9 %	0,343 m
4	1,9 %	0,136 m	5,2 %	0,361 m	4,4 %	0,304 m	16,5 %	1,155 m
6	5,1 %	0,354 m	3,6 %	0,252 m	4,9 %	0,340 m	14,3 %	0,999 m
8	13,3 %	0,928 m	8,8 %	0,617 m	7,3 %	0,509 m	8,7 %	0,611 m

Taula 15 Taula d'errors de Mínims Quadrats del vídeo 3. Es mostra tant l'error relatiu (vermell) com l'error absolut (gris).

Podem veure les variables NumMin i Xframe amb els distints valors que hem emprat. Dins el quadre gris es marca l'error relatiu comés en metres i en el quadre vermell es marca l'error absolut. A la Figura 57 es mostra la corresponent gràfica de la Taula 15.

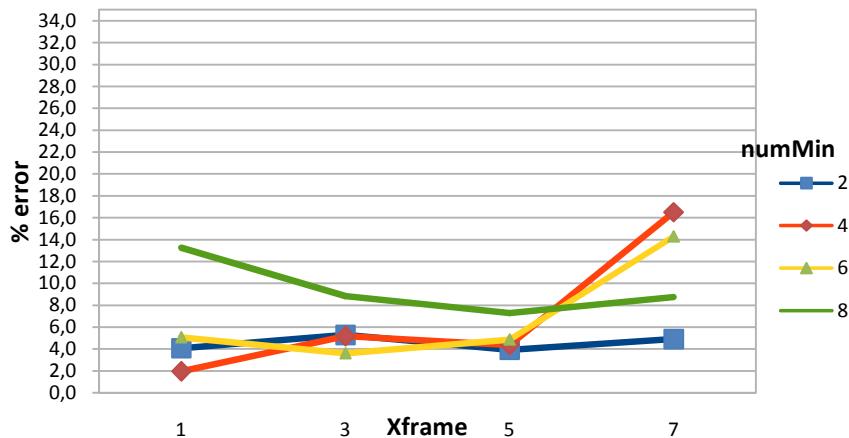


Figura 57 Gràfica de resultats de Mínims Quadrats del vídeo 3.

En aquest cas no es veu un patró clar, però es pot observar que els resultats més bons es situen amb l'opció numMin baixa i amb pocs frames de distància. L'error més baix amb un 1,9% es troba a l'elecció: Xframe = 1 i numMin = 4.

- **Error en posició final de l'odometria (RANSAC).**

En aquesta secció es mostrerà l'error absolut comés per l'odometre amb l'opció seleccionada RANSAC en format tabular (Taula 16).

	Xframe	1	3	5	7
minCons	1	2	3	4	5
numMin=2	60	2,1 %	3,6 %	4,5 %	4,1 %
SELEC= 1	70	1,9 %	3,1 %	4,4 %	4,1 %
	80	1,7 %	3,6 %	4,7 %	4,1 %

Taula 16 Taula d'errors de RANSAC del vídeo 3. A la taula es mostra l'error relatiu per a diferents valors de Xframe, minConsensus i els valors fixos de numMin i SELEC (quadre gris puntejat).

A la figura 58 es representa la gràfica de la taula d'errors de RANSAC (Taula 16).

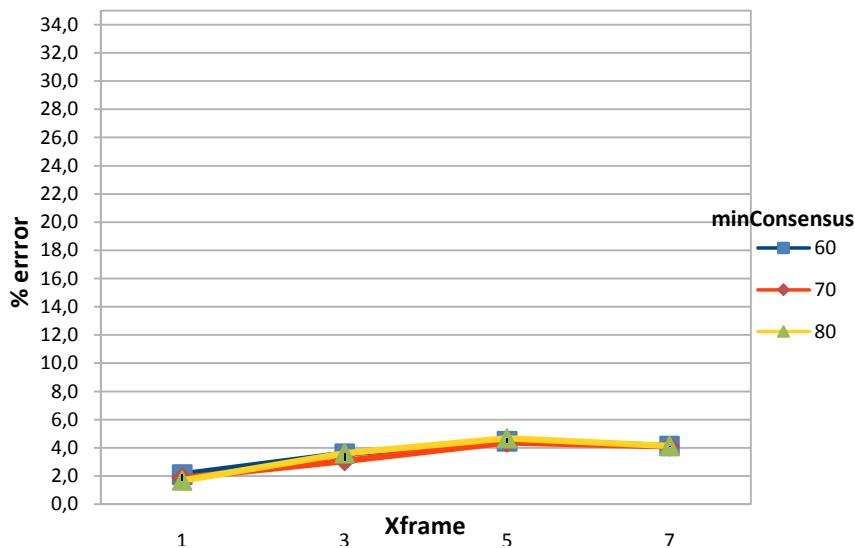


Figura 58 Gràfica de resultats de RANSAC del vídeo 3.

En aquest cas augmenta l'error a mesura que augmenta la distància entre *frames*. No hi ha gaires diferències entre els percentatges de consens. L'error més baix amb un **1,7%** es troba a l'opció: **Xframe = 1, numMin = 2, selec = 1 i minConsensus = 80%**.

- Error en distància recorreguda pel robot.**

Dins aquest apartat es mostrerà la distància acumulada per l'odometre durant tot el recorregut en format tabular. Mostrarem els resultats obtinguts amb Mínims Quadrats (Taula 17) i RANSAC (Taula 18).

Vuit	1	3	5	7	Xframe numMin
Distància: 7,854 m	8,61 m	8,47 m	8,43 m	8,37 m	2
	8,57 m	8,41 m	8,43 m	8,11 m	4
	8,59 m	8,30 m	7,67 m	6,32 m	6
	8,25 m	7,27 m	5,56 m	3,88 m	8
	8,5 m	8,1 m	7,5 m	6,7 m	Mitja
	0,3 m	1,0 m	2,4 m	3,6 m	Desviació típica
	8,3 %	3,3 %	4,2 %	15,1 %	Error %

Taula 17 Error acumulat utilitzant Mínims Quadrats del vídeo 3.

Vuit	1	3	5	7	Xframe numMin
Distància: 7,854 m	8,61 m	8,47 m	8,43 m	8,37 m	2
	8,59 m	8,41 m	8,44 m	8,16 m	4
	8,60 m	8,34 m	7,80 m	6,55 m	6
	8,41 m	7,56 m	5,85 m	4,13 m	8
	8,6 m	8,2 m	7,6 m	6,8 m	Mitja
	0,2 m	0,7 m	2,1 m	3,4 m	Desviació típica
	8,9 %	4,3 %	2,8 %	13,4 %	Error %

Taula 18 Error acumulat utilitzant RANSAC (70%, SELEC=1) del vídeo 3.

A les taules es mostren les distàncies acumulades per a diferents valors de Xframe i numMin. En el quadre marró puntejat es mostren els metres teòrics realitzats pel robot, on es comparen amb la mitja (groc) i es treu l'error relatiu (vermell). També es pot veure la desviació típica de les dades (groc).

A partir de Xframe = 5 l'error augmenta considerablement amb una selecció de numMin alta com es pot veure en l'augment de la desviació típica ja que els resultats varien molt per les males Odometries. No es poden treure conclusions clares en ambdues taules.

- **Temps d'anàlisi d'imatges.**

En aquest apartat es mostraran els temps d'anàlisi del vídeo 3. En les següents taules s'ha calculat les mitjanes d'aquests temps estimats tant per Mínims Quadrats (Taula 19) com per les proves agafades de RANSAC (Taula 20). En el cas de RANSAC hem seleccionat els mateixos valors que en el càlcul de l'error en distància recorreguda.

Vuit	1	3	5	7	Xframe numMin
Temps: 43 s	259 s	91 s	56 s	41 s	2
	253 s	96 s	53 s	40 s	4
	250 s	100 s	56 s	40 s	6
	252 s	89 s	56 s	40 s	8
	254 s	94 s	55 s	40 s	Mitja

Taula 19 Taula de temps estimat amb Mínims Quadrats del vídeo 3.

Vuit	1	3	5	7	Xframe numMin
Temps: 43 s	292 s	99 s	63 s	42 s	2
	310 s	121 s	59 s	42 s	4
	322 s	168 s	66 s	47 s	6
	372 s	112 s	82 s	48 s	8
	324 s	125 s	67 s	45 s	Mitja

Taula 20 Taula de temps estimat amb RANSAC (70%, SELEC=1) del vídeo 3.

A les taules es mostren els temps mesurats per als diferents valors de Xframe i numMin. En el cas de RANSAC amb els dos paràmetres fixos esmenats al peu de la taula. En el quadre marró puntejat es mostra el temps real que ha durat el vídeo.

En ambdues taules el patró es el mateix que amb les anteriors proves, baixa considerablement el temps d'execució a mesura que augmenta la separació entre frames. En aquest cas no s'aprecia tant l'augment del temps amb l'augment del límit numMin.

5.4 Resultats obtinguts del vídeo 4.

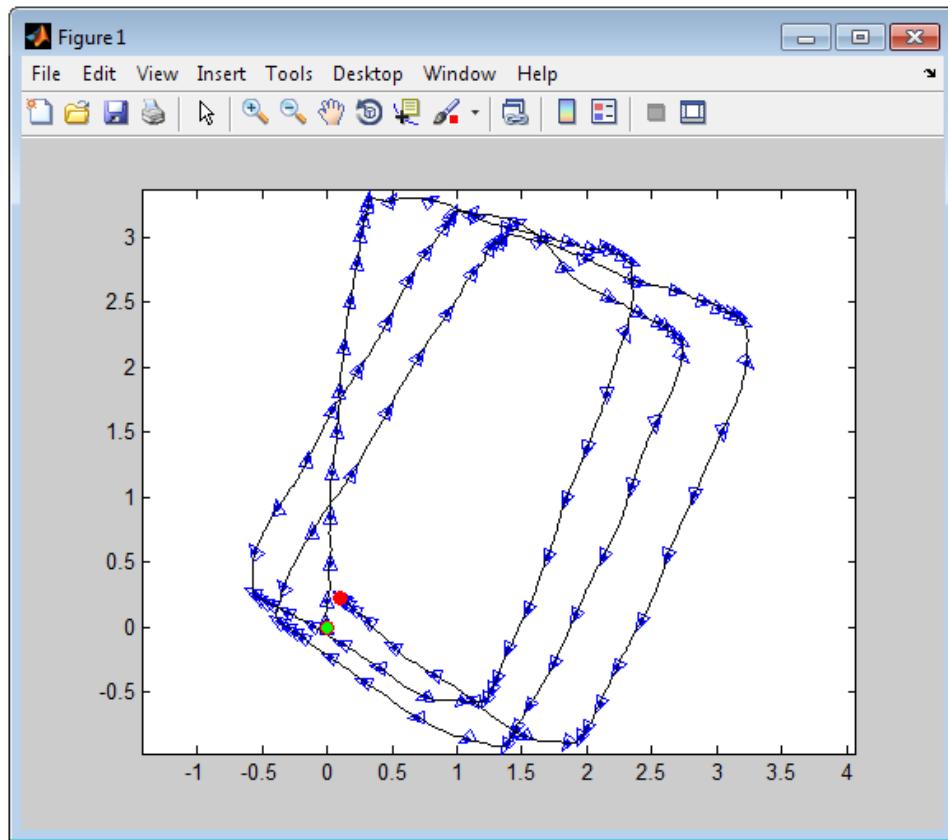


Figura 59 Resultat del vídeo 4. El triangle mostra l'orientació del robot durant el trajecte i els punts verds i vermells indiquen el inici i el final del trajecte respectivament.

En aquest apartat es mostraran les proves experimentals realitzades pel recorregut 4 o vídeo 4. A la Figura 59 es mostra un exemple d'odometria calculada per aquest recorregut.

A continuació s'exposaran els resultats més significatius de les proves experimentals del vídeo 4 tant de Mínims Quadrats com de RANSAC. També es mostrarà una taula del temps emprat per a l'execució del programa.

- **Error en posició final de l'odometria (Mínims Quadrats).**

En aquesta secció es mostren els resultats de les proves obtingudes amb Mínims Quadrats per al vídeo 4 (Taula 21).

Xframe numMin	1		3		5		7	
	E.Relatiu	E.Absolut	E.Relatiu	E.Absolut	E.Relatiu	E.Absolut	E.Relatiu	E.Absolut
2	4,9 %	0,531 m	5,4 %	0,580 m	6,9 %	0,738 m	34,2 %	3,676 m
4	6,3 %	0,683 m	6,1 %	0,659 m	33,6 %	3,613 m	22,5 %	2,419 m
6	6,8 %	0,732 m	8,7 %	0,932 m	32,9 %	3,544 m	12,0 %	1,292 m
8	12,4 %	1,333 m	15,7 %	1,693 m	13,2 %	1,422 m	7,0 %	0,749 m

Taula 21 Taula d'errors de Mínims Quadrats del vídeo 4. Es mostra tant l'error relatiu (vermell) com l'error absolut (gris).

Podem veure les variables NumMin i Xframe amb els distints valors que hem emprat. Dins el quadre gris es marca l'error relatiu comés en metres i en el quadre vermell es marca l'error absolut. A la Figura 60 es mostra la corresponent gràfica de la Taula 21.

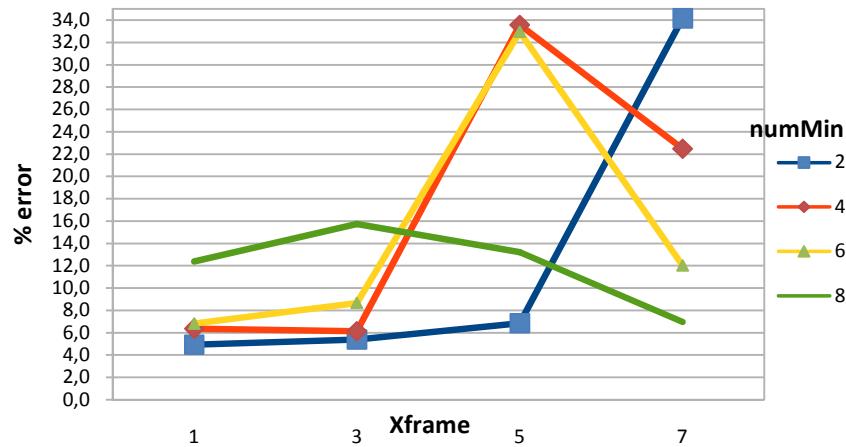


Figura 60 Gràfica de resultats de Mínims Quadrats del vídeo 4.

En aquestes proves tampoc es pot trobar una tendència clara ja que només s'ha realitzat una vegada. Es pot observar a la gràfica que l'error augmenta considerablement augmentant la distància entre frames. La selecció de numMin = 8 dóna error molts grans en totes les proves. L'error més baix amb un **4,9%** es troba a l'elecció: **Xframe = 1 i numMin = 2**.

- **Error en posició final de l'odometria (RANSAC).**

En aquesta secció es mostrerà l'error absolut comés per l'odometre amb l'opció seleccionada RANSAC en format tabular (Taula 22).

	Xframe	1	3	5	7
minCons					
numMin=2	60	6,5 %	5,1 %	6,3 %	1,9 %
	70	6,1 %	5,9 %	5,8 %	4,4 %
	80	4,3 %	7,0 %	5,9 %	5,1 %

Taula 22 Taula d'errors de RANSAC del vídeo 4. A la taula es mostra l'error relatiu per a diferents valors de Xframe, minConsensus i els valors fixos de numMin i SELEC (quadre gris puntejat).

A la figura 61 es representa la gràfica de la taula d'errors de RANSAC (Taula 22).

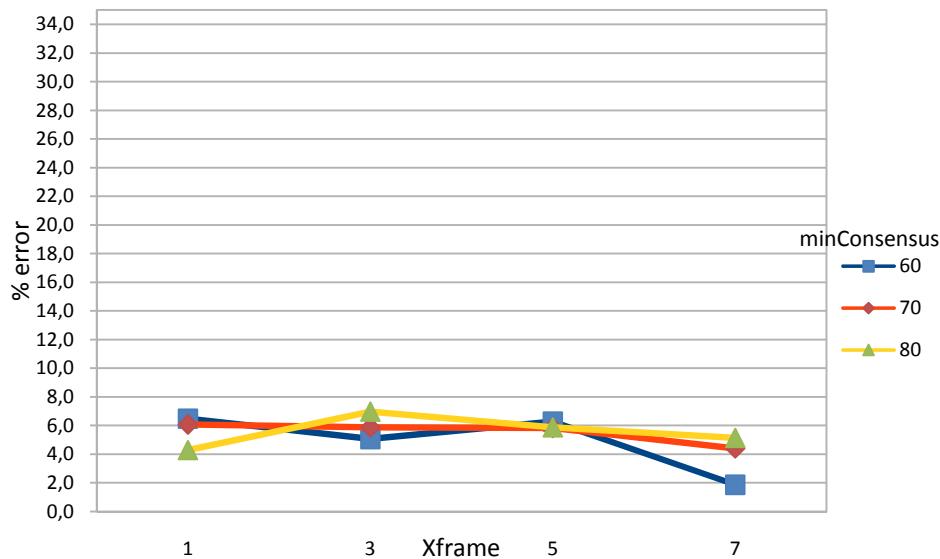


Figura 61 Gràfica de resultats de RANSAC del vídeo 4.

En aquest cas no es veu un patró clar en la gràfica ja que els errors varien molt. L'error més baix amb un **4,3%** es troba a l'elecció de paràmetres següent: **Xframe = 1, numMin = 2, selec = 1 i minConsensus = 80%**. S'haurien de realitzar més proves per a extreure conclusions clares.

- Error en distància recorreguda pel robot.**

Dins aquest apartat es mostrerà la distància acumulada per l'odometre durant tot el recorregut en format tabular. Mostrarem els resultats obtinguts amb Mínims Quadrats (Taula 23) i RANSAC (Taula 24).

Quadrat 3 voltes	1	3	5	7	Xframe \ numMin
Distància: 32,28 m	34,29 m	34,16 m	34,09 m	30,90 m	2
	34,33 m	33,67 m	27,32 m	17,14 m	4
	33,18 m	27,41 m	16,07 m	10,17 m	6
	27,89 m	17,92 m	11,10 m	7,67 m	8
	32,4 m	28,3 m	22,1 m	16,5 m	Mitja
	5,3 m	13,1 m	18,1 m	18,0 m	Desviació típica
	0,4 %	12,4 %	31,4 %	49,0 %	Error %

Taula 23 Error acumulat utilitzant Mínims Quadrats del vídeo 4.

Quadrat 3 voltes	1	3	5	7	Xframe \ numMin
Distància: 32,28 m	33,41 m	34,17 m	33,92 m	33,12 m	2
	34,35 m	34,24 m	32,46 m	26,88 m	4
	33,35 m	34,23 m	26,87 m	20,45 m	6
	26,52 m	31,92 m	21,32 m	15,37 m	8
	31,9 m	33,6 m	28,6 m	24,0 m	Mitja
	6,3 m	2,0 m	10,0 m	13,4 m	Desviació típica
	1,2 %	4,2 %	11,3 %	25,8 %	Error %

Taula 24 Error acumulat utilitzant RANSAC (70%, SELEC=1) del vídeo 4.

A les taules es mostren les distàncies acumulades per a diferents valors de Xframe i numMin. En el quadre marron puntejat es mostren els metres teòrics realitzats pel robot, on es comparen amb la mitja (groc) i es treu l'error relatiu (vermell). També es pot veure la desviació típica de les dades (groc).

Com podem observar a quasi totes les mitjanes que la desviació típica és molt elevada, ja que els errors comesos per l'dometre són molt grans, com per exemple a numMin = 8. Les proves amb menys error es trobarien a un valor Xframe baix i un valor de numMin baix en ambdues taules.

- **Temps d'anàlisi d'imatges.**

En aquest apartat es mostraran els temps d'anàlisi del vídeo 4. En les següents taules s'ha calculat les mitjanes d'aquests temps estimats tant per Mínims Quadrats (Taula 25) com per les proves agafades de RANSAC (Taula 26). En el cas de RANSAC hem seleccionat els mateixos valors que en el càlcul de l'error en distància recorreguda.

Quadrat 3 voltes	1	3	5	7	Xframe numMin
Temps: 100 s	516 s	175 s	121 s	73 s	2
	540 s	182 s	99 s	70 s	4
	593 s	156 s	102 s	70 s	6
	510 s	179 s	129 s	73 s	8
	540 s	173 s	113 s	72 s	Mitja

Taula 25 Taula de temps estimat amb Mínims Quadrats del vídeo 4.

Quadrat 3 voltes	1	3	5	7	Xframe numMin
Temps: 100 s	586 s	179 s	98 s	73 s	2
	629 s	198 s	115 s	80 s	4
	707 s	211 s	128 s	87 s	6
	649 s	203 s	136 s	98 s	8
	643 s	198 s	119 s	85 s	Mitja

Taula 26 Taula de temps estimat amb RANSAC (70%, SELEC=1) del vídeo 4.

Com hem anat esmenant durant totes les proves, el temps baixa al augmentar la distància entre frames. En aquest cas no es veu un patró clar en quant a l'augment del temps i el valor de numMin.

5.5 Resultats obtinguts del vídeo 5.

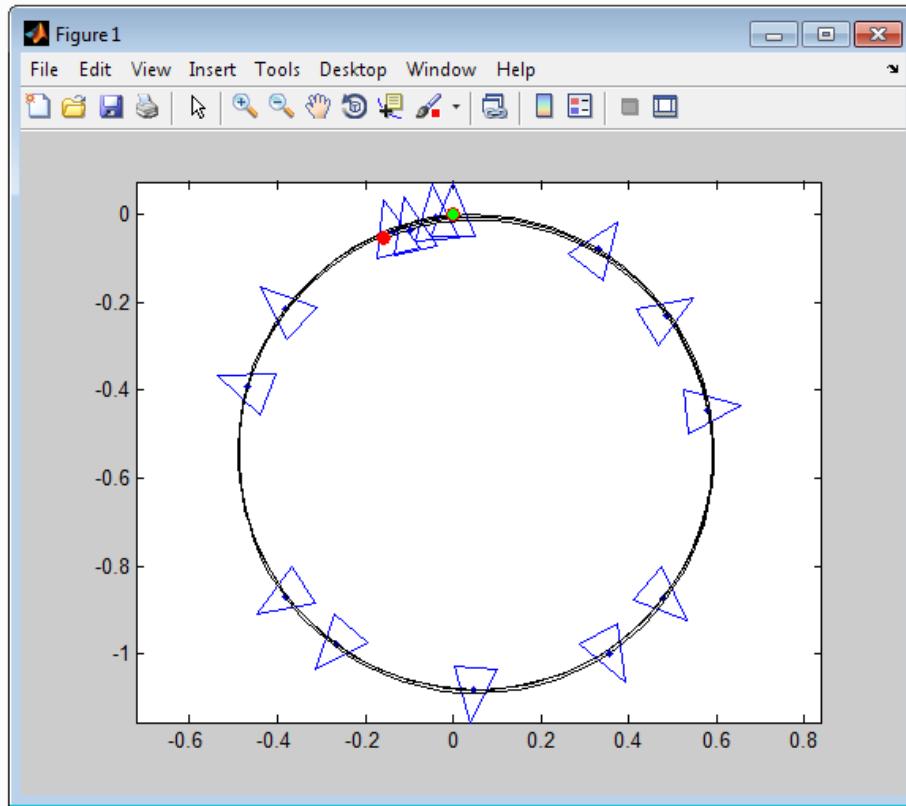


Figura 62 Resultat del vídeo 5. El triangle mostra l'orientació del robot durant el trajecte i els punts verds i vermell indiquen el inici i el final del trajecte respectivament.

En aquest apartat es mostraran les proves experimentals realitzades pel recorregut 5 o vídeo 5. A la Figura 62 es mostra un exemple d'odometria calculada per aquest recorregut.

A continuació s'exposaran els resultats més significatius de les proves experimentals del Vídeo 5 tant de Mínims Quadrats com de RANSAC. També es mostrarà una taula del temps emprat per a l'execució del programa.

- **Error en posició final de l'odometria (Mínims Quadrats).**

En aquesta secció es mostren els resultats de les proves obtingudes amb Mínims Quadrats per al vídeo 5 (Taula 27).

Xframe numMin	1		3		5		7	
	E.Relatiu	E.Absolut	E.Relatiu	E.Absolut	E.Relatiu	E.Absolut	E.Relatiu	E.Absolut
2	1,7 %	0,162 m	2,0 %	0,195 m	0,7 %	0,066 m	2,0 %	0,194 m
4	1,6 %	0,155 m	2,7 %	0,263 m	1,7 %	0,166 m	27,0 %	2,646 m
6	2,1 %	0,204 m	2,1 %	0,209 m	4,7 %	0,459 m	8,2 %	0,805 m
8	3,1 %	0,305 m	6,7 %	0,654 m	6,8 %	0,668 m	6,1 %	0,598 m

Taula 27 Taula d'errors de Mínims Quadrats del vídeo 5. Es mostra tant l'error relatiu (vermell) com l'error absolut (gris).

Podem veure les variables NumMin i Xframe amb els distints valors que hem emprat. Dins el quadre gris es marca l'error relatiu comés en metres i en el quadre vermell es marca l'error absolut. A la Figura 63 es mostra la corresponent gràfica de la Taula 27.

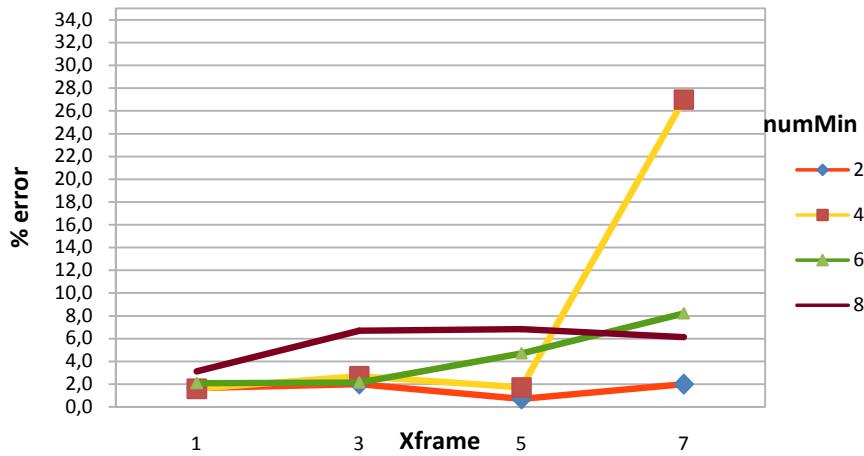


Figura 63 Gràfica de resultats de Mínims Quadrats del vídeo 5.

En aquest cas ens trobem que alguns resultats del càlcul d'errors de l'odometria són bons, quan realment no ho són. En el cas de ($C=2, B=5$) i ($C=4, B=5$) amb un error de 0,7% i 1,7% respectivament exemplifiquen clarament els casos d'errors baixos falsos explicats a l'apartat 3. L'error més baix amb **1,6%** es troba a l'opció: **Xframe = 1 i numMin = 4**. En molts de casos una alta separació entre frames provoca una pujada de l'error molt gran com és el cas de numMin = 4 i Xframe = 7.

- Error en posició final de l'odometria (RANSAC).**

En aquesta secció es mostrerà l'error absolut comés per l'odometre amb l'opció seleccionada RANSAC en format tabular (Taula 28).

	Xframe minCons	1	3	5	7
numMin=6	60	1,0 %	1,6 %	4,4 %	7,4 %
SELEC= 2	70	0,9 %	1,4 %	5,4 %	8,0 %
	80	0,2 %	1,7 %	5,8 %	9,1 %

Taula 28 Taula d'errors de RANSAC del vídeo 5. A la taula es mostra l'error relatiu per a diferents valors de Xframe, minConsensus i els valors fixos de numMin i SELEC (quadre gris puntejat).

A la figura 64 es representa la gràfica de la taula d'errors de RANSAC (Taula 28).

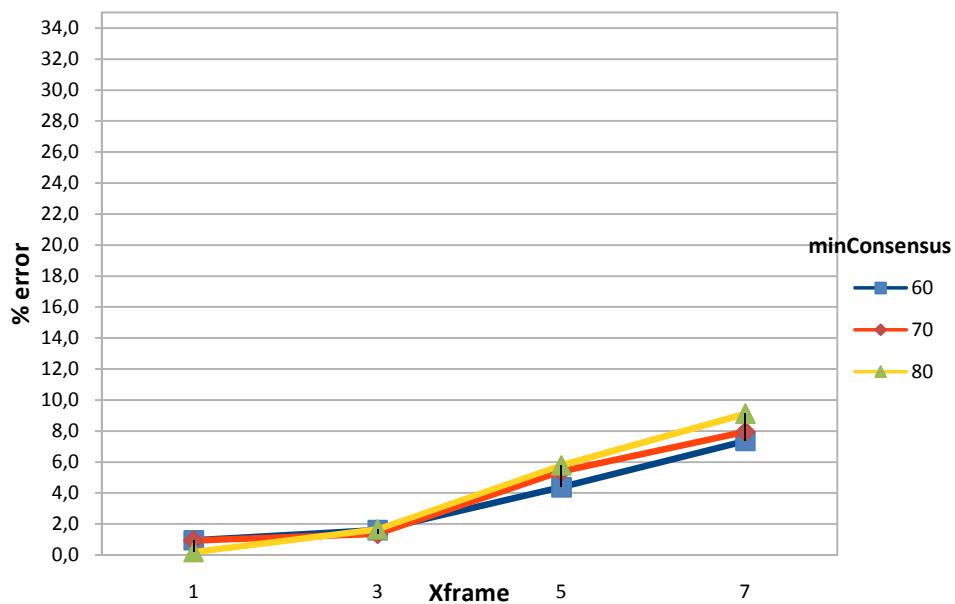


Figura 64 Gràfica de resultats de RANSAC del vídeo 5.

Es pot observar clarament que al augmentar la separació entre *frames* l'error també augmenta. No hi ha una clara diferència entre els percentatges de consens. L'error més baix amb **0,2%** es troba a l'opció: **Xframe = 1, numMin = 6, selec = 2 i minConsensus = 80%**.

- Error en distància recorreguda pel robot.**

Dins aquest apartat es mostrerà la distància acumulada per l'odometre durant tot el recorregut en format tabular. Mostrarem els resultats obtinguts amb Mínims Quadrats (Taula 29) i RANSAC (Taula 30).

Cercle 3 voltes	1	3	5	7	Xframe \ numMin
Distància: 9,801 m	10,36 m	10,35 m	10,33 m	10,33 m	2
	10,37 m	10,31 m	9,87 m	8,84 m	4
	10,30 m	9,87 m	7,84 m	5,81 m	6
	9,76 m	8,30 m	5,43 m	3,64 m	8
	10,2 m	9,7 m	8,4 m	7,2 m	Mitja
	0,5 m	1,7 m	3,9 m	5,2 m	Desviació típica
	4,0 %	0,9 %	14,6 %	27,0 %	Error %

Taula 29 Error acumulat utilitzant Mínims Quadrats del vídeo 5.

Cercle 3 voltes	1	3	5	7	Xframe \ numMin
Distància: 9,801 m	10,38 m	10,39 m	10,36 m	10,25 m	2
	10,37 m	10,37 m	10,23 m	10,05 m	4
	10,30 m	10,35 m	9,63 m	9,17 m	6
	9,83 m	10,35 m	8,85 m	8,50 m	8
	10,2 m	10,4 m	9,8 m	9,5 m	Mitja
	0,5 m	0,0 m	1,2 m	1,4 m	Desviació típica
	4,3 %	5,7 %	0,3 %	3,2 %	Error %

Taula 30 Error acumulat utilitzant RANSAC (70%, SELEC=2) del vídeo 5.

A les taules es mostren les distàncies acumulades per a diferents valors de Xframe i numMin. En el quadre marró puntejat es mostren els metres teòrics realitzats pel robot, on es comparen amb la mitja (groc) i es treu l'error relatiu (vermell). També es pot veure la desviació típica de les dades (groc).

Com més augmenta la distància entre *frames*, s'observa com la desviació típica augmenta i això ens dóna pistes de que l'error està augmentant molt, ja que hi ha grans diferències entre Odometries. En el cas de Mínims Quadrats es veu més clarament.

- **Temps d'anàlisi d'imatges.**

En aquest apartat es mostraran els temps d'anàlisi del vídeo 5. En les següents taules s'ha calculat les mitjanes d'aquests temps estimats tant per Mínims Quadrats (Taula 31) com per les proves agafades de RANSAC (Taula 32). En el cas de RANSAC hem seleccionat els mateixos valors que en el càlcul de l'error en distància recorreguda.

Cercle 3 voltes	1	3	5	7	Xframe numMin
Temps: 32 s	189 s	62 s	37 s	27 s	2
	182 s	61 s	39 s	28 s	4
	185 s	59 s	36 s	27 s	6
	187 s	61 s	36 s	28 s	8
	186 s	61 s	37 s	27 s	Mitja

Taula 31 Taula de temps estimat amb Mínims Quadrats del vídeo 5.

Cercle 3 voltes	1	3	5	7	Xframe numMin
Temps: 32 s	196 s	59 s	49 s	27 s	2
	238 s	75 s	51 s	30 s	4
	235 s	78 s	44 s	32 s	6
	267 s	85 s	47 s	33 s	8
	234 s	74 s	48 s	30 s	Mitja

Taula 32 Taula de temps estimat amb RANSAC (70%, SELEC=2) del vídeo 5.

A les taules es mostren els temps mesurats per als diferents valors de Xframe i numMin. En el cas de RANSAC amb els dos paràmetres fixos esmenats al peu de la taula. En el quadre marró puntejat es mostra el temps real que ha durat el vídeo.

En ambdues taules el temps disminueix a mesura que augmenta la distància entre *frames*. En el cas de Mínims Quadrats no es veu un patró clar relacionat amb el temps i el paràmetre numMin. A RANSAC es veu un augment del temps a mesura que augmenta el llindar numMin.

5.6 Paràmetres finals.

Una vegada hem seleccionat els millors resultats de totes les proves realitzades hem creat una taula amb les opcions seleccionades com a millors per a cada Vídeo realitzat. Hem marcat en vermell els valors seleccionats.

Mínims Q.	Xframe	numMin	Error
Vídeo 1	5	8	4,8%
Vídeo 2	1	8	3,1%
Vídeo 3	1	4	1,9%
Vídeo 4	1	2	4,9%
Vídeo 5	1	4	1,6%

Taula 33 Taula amb les millors opcions per vídeo(Mínims Quadrats).

Les millors opcions quan només utilitzem Mínims Quadrats es veu clarament a la (Taula 33), per nombre de separació entre *frames* és de 1. Mentre que l'opció per numMin no és tan clara ja que no hi ha cap valor que predomini sobre els altres.

RANSAC	Xframe	numMin	selec	minCons	Error
Vídeo 1	7	8	1	70%	4,1%
Vídeo 2	1	2	2	70%	2,7%
Vídeo 3	1	2	1	80%	1,7%
Vídeo 4	1	2	1	80%	4,3%
Vídeo 5	1	2	2	80%	0,2%

Taula 34 Taula amb les millors opcions per vídeo (RANSAC).

Les millors opcions quan utilitzem RANSAC es veuen de manera clara a la (Taula 34). Per a la separació entre *frames* (Xframe) el nombre més utilitzat és 1. En quant al filtre de *matching* (numMin) el més repetit és 2. L'elecció d'una opció externa a RANSAC la més utilitzada és la 1 (odometria anterior). El percentatge de consens el més utilitzat és 80%.

Visualitzant els errors comesos per Mínims Quadrats i RANSAC es pot veure que RANSAC millora els errors comesos per Mínims Quadrats ja que elimina en la mesura del possible els falsos *matchings*.

Per contra, el temps que utilitza RANSAC per a analitzar les imatges és mes costós que la utilització de únicament Mínims Quadrats com es pot veure a les Taules de temps de l'apartat anterior.

Ja que el nostre objectiu és aconseguir el mínim error possible d'odometria i la diferència de temps no és molt gran la millor opció per realitzar l'odometria és utilitzant RANSAC, anomenat paràmetre A. Amb els valors dels paràmetres abans especificats: **Xframe = 1**, **numMin = 2**, **selec = 1** i **minConsensus = 80%**. Com es veu a la Taula 35 dels paràmetres finals seleccionats.

Odometria	Xframe	numMin	selec	minCons
2	1	2	1	80

Taula 35 Elecció final de paràmetres.

Capítol 6. Conclusions

En aquest apartat mostrarem les conclusions extretes durant la realització del Projecte de Final de Carrera. Així mateix, mostrarem que els objectius marcats a cada un dels apartats s'han assolit correctament.

1. Conclusions de la calibració de la càmera.

El mètode que hem utilitzat és el mètode de Zhang i podem dir que ha proporcionat totes les funcions que necessitàvem. En particular, hem pogut extreure les matrius necessàries per al correcte desenvolupament del projecte. Aquest mètode és d'una fàcil implementació i és bastant eficient ja que amb 4 o 5 imatges és suficient per extreure la informació necessària. No té una total autonomia ja que s'han d'indicar alguns paràmetres coneguts de la imatge, com els quadrats del tauler d'escacs i algunes mesures més.

Ja que el nivell de distorsió de la nostra càmera no és molt elevat, la calibració d'aquesta no ha afectat en gran mesura al resultat final de la nostra odometria. Ara bé, fins i tot amb càmeres amb poca distorsió, la calibració és indispensable si s'han de dur a terme trajectòries llargues. En canvi, extreure les propietats de la càmera per a poder calcular la relació pixel-metre ha estat fonamental i cal destacar que el mètode és molt precís.

2. Conclusions de la recerca de característiques.

Pel que fa a la recerca de característiques s'han complert tots els objectius marcats inicialment. Les llibreries OpenCV, implementades en C++ ens ofereixen l'oportunitat d'utilitzar SIFT i FLANN amb una alta eficiència, una gran velocitat d'execució i una més que senzilla implementació. El nostre algoritme és capaç de captar gran quantitat de característiques en diversos tipus de superfícies. Ja que sempre existeixen diversos tipus de superfícies s'hauran d'especificar els valors llindars per a un correcte detecció de característiques i descart de falsos *matchings* com numMin de FLANN. La configuració d'aquest apartat és essencial per a una odometria final acurada.

3. Conclusions del càlcul de moviment.

En aquest apartat hem aconseguit obtenir, a partir de localitzar els punts interessants de dues imatges consecutives, el moviment que hi ha hagut entre elles. També es pot dir que s'han assolit en la seva totalitat els objectius ja que sempre que la part de recerca de característiques hagi funcionat de manera correcte aquest apartat no té gaires inconvenients en extreure els resultats de moviment. L' implementació de RANSAC és un dels responsables de que el càlcul del moviment sigui més robust, ja que Mínims Quadrats en alguns aspectes flaueja.

4. Conclusions de les proves finals.

Aquest apartat demostra que tots els apartats anteriors han estat correctament implementats, ja que en totes les proves que s'han realitzat l'odometria ha estat capaç de calcular-nos, amb major o menor error, una estimació de la posició acurada.

El programa de Matlab realitzat és poc autònom ja que requereix d'alguna informació per al correcte càlcul d'error com el nom de l'arxiu d'odometria que es vol llegir, o el valor del punt final real i en el cas del Vídeo 4 també els punts de reinici de l'Odometria.

En l'apartat d'anàlisi de les mesures de l'error, la utilització de la posició final de l'odometria ha donat bons resultats en un percentatge elevat. Aquesta mesura d'error és bona per a avaluar estimacions de posició en recorreguts curts als quals l'error acumulat no sigui molt elevat. El vídeo 4 ha permès mostrar un exemple en que aquesta mesura falla en trajectòries llargues. En aquest vídeo la mesura de l'error és molt elevada mentre que l'error real no és significativament gran.

La distància recorreguda pel robot és bona per a extreure conclusions sobre la tremolor de la càmera (Figura 49). Però en qualsevol cas no dóna informació sobre l'error cometut en la posició del robot.

La precisió del càlcul d'error es podria millorar amb la utilització d'un *ground truth*, ja que com s'ha explicat abans la utilització del punt final a vegades no dóna resultats reals i pot donar lloc a grans errors o confusions.

Finalment s'han pogut extreure els valors dels paràmetres per a una odometria més robusta mitjançant la realització de diverses proves i, per tant, s'ha complert l'objectiu final establert en aquest PFC.

Capítol 7. Propostes de millora

En aquest apartat exposarem una sèrie de propostes que permetrien millorar el sistema de localització presentat en aquest projecte. Aquestes propostes giren entorn als punts febles que hem extret a les conclusions.

1. Unificació del programa.

Com hem explicat al document, el nostre projecte utilitza diferents programes per a realitzar les proves necessàries. La utilització de diversos programes ens dóna una comoditat a l'hora de implementar les diferents funcions. Aquesta estratègia es recomanable per a començar a experimentar amb el programa per una fàcil implementació. Però a la llarga es recomanable unificar els programes utilitzats per una millora de l'eficiència per a realitzar proves i altres aplicacions.

Al nostre projecte, la calibració de la càmera, el càlcul d'errors i la visualització per pantalla es realitzen amb Matlab mentre que l'anàlisi de les imatges i el càlcul de l'odometria es realitza en C++. Aquestes podrien estar unificades amb un sol programa, fent molt més eficient la programació i obriria portes a altres aplicacions com per exemple la implementació de l'odometre en temps real.

2. Millora de programació.

El nostre programa necessita de molts passos previs per a realitzar les proves per tant podríem dir que no té l'autonomia suficient en algunes parts del nostre programa. La conversió del vídeo en un conjunt de imatges o *frames* amb els seus corresponents noms i l'arxiu d'entrada amb els valors dels paràmetres amb els quals es realitzaran les proves són un exemple. Aquestes tasques són molt tedioses si pretenem realitzar una gran quantitat de proves.

Les propostes serien següents:

- Implementació d'un codi que permeti donar com arxiu d'entrada el format de vídeo utilitzat (.avi) i realitzar les conversions a imatge i la modificació de la grandària de la mateixa. Aquest apartat s'implementaria amb C++ utilitzant les llibreries de OpenCV. Aquesta proposta milloraria la comoditat a l'hora de realitzar noves proves amb nous vídeos.
- Implementació d'un codi que permeti realitzar múltiples odometries en sèrie. En aquest cas l'arxiu d'entrada especificaria quantes proves volem realitzar, amb quins paràmetres i amb quins vídeos. Amb aquesta millora augmentaríem un esglao l'autonomia d'aquest programa.

3. Experimentació amb més vídeos.

La quantitat de vídeos que hem realitzat en el present projecte en molts de casos és insuficient per a treure conclusions clares, sobretot en vídeos que donen peu a molts d'errors. En les proves hem vist casos com a la Figura 61 que no es veu cap tendència clara. En aquests casos en especial s'haurien de realitzar una gran quantitat de proves. En la resta de casos també s'haurien de realitzar per descartar falsos positius.

Així com l'augment de vídeos, també s'hauria d'augmentar el rang de valors dels paràmetres i el nombre de paràmetres seleccionats dins d'aquest rang. En aquest aspecte augmentaria considerablement el nombre de proves a realitzar. Aquest augment de proves i vídeos ens podria acostar a conclusions molt més clares per una odometria robusta.

4. Utilització del Ground Truth.

Durant la realització de les proves hem comprovat que moltes odometries no eren tant dolentes o tant bones com l'error final donava a entendre. Per tant la necessitat d'observar totes les proves realitzades per comprovar si realment concorda aproximadament amb el valor de l'error és un problema greu. En aquest aspecte el nostre programa de detecció d'errors podria ser molt més precís utilitzant el *Ground Truth*. Aquest mètode ens donaria una fiabilitat més alta dins el càlcul d'errors i per tant llevaria la necessitat de revisar manualment els resultats proporcionats per l'odòmetre. També augmentaria la precisió en el càlcul de l'error.

La utilització d'aquest mètode té els seus inconvenients. Implementar el càlcul d'error mitjançant un *Ground Truth* dins la programació no seria el problema més important. El problema més important és la manera de generar un *Ground Truth* precís en distàncies curtes.

Apèndix

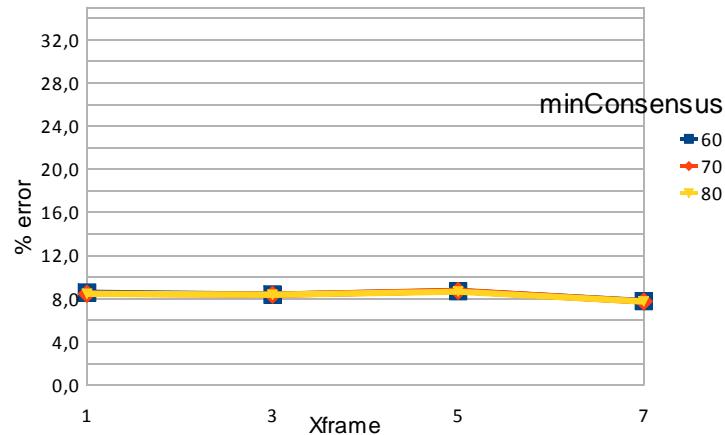
En aquest apartat exposarem tots els resultats obtinguts amb RANSAC. Aquests resultats provenen de totes les proves que hem realitzat amb RANSAC.

Aquesta taula ens mostra totes les proves realitzades en error de punt final d'odometria amb RANSAC del vídeo 1. Està dividit en colors per una ràpida visualització.

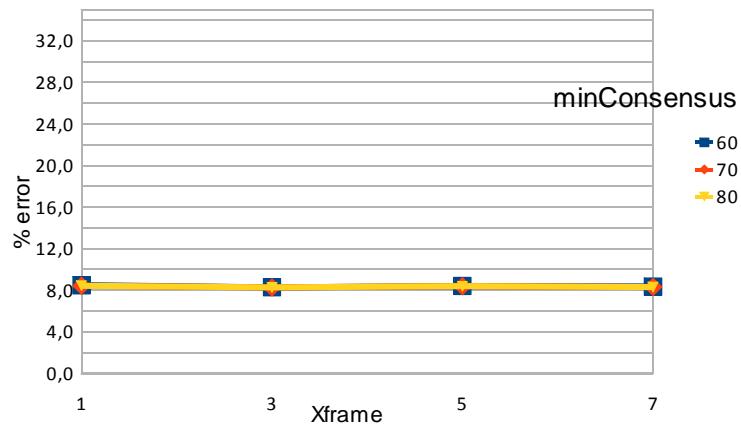
numMin	Xframe	Selec RANSAC	% Consensus	1		3		5		7	
				E. Relatiu	E. Absolut						
2	1	60 %	8,6 %	0,214 m	8,4 %	0,210 m	8,7 %	0,218 m	7,8 %	0,195 m	
		70 %	8,5 %	0,212 m	8,4 %	0,210 m	8,7 %	0,218 m	7,8 %	0,194 m	
		80 %	8,5 %	0,212 m	8,4 %	0,210 m	8,7 %	0,217 m	7,8 %	0,194 m	
	2	60 %	8,5 %	0,212 m	8,3 %	0,207 m	8,4 %	0,210 m	8,3 %	0,209 m	
		70 %	8,4 %	0,211 m	8,3 %	0,207 m	8,4 %	0,210 m	8,3 %	0,208 m	
		80 %	8,4 %	0,211 m	8,3 %	0,207 m	8,4 %	0,210 m	8,3 %	0,207 m	
4	1	60 %	8,5 %	0,213 m	8,4 %	0,210 m	8,4 %	0,210 m	8,4 %	0,210 m	
		70 %	8,5 %	0,212 m	8,4 %	0,210 m	8,4 %	0,210 m	8,4 %	0,210 m	
		80 %	8,5 %	0,212 m	8,4 %	0,210 m	8,4 %	0,210 m	8,4 %	0,211 m	
	2	60 %	8,5 %	0,213 m	8,4 %	0,210 m	8,4 %	0,210 m	8,4 %	0,210 m	
		70 %	8,5 %	0,212 m	8,4 %	0,210 m	8,4 %	0,210 m	8,4 %	0,210 m	
		80 %	8,5 %	0,212 m	8,4 %	0,210 m	8,4 %	0,210 m	8,4 %	0,211 m	
6	1	60 %	8,5 %	0,213 m	8,5 %	0,212 m	8,5 %	0,212 m	8,5 %	0,211 m	
		70 %	8,5 %	0,213 m	8,5 %	0,212 m	8,5 %	0,212 m	8,5 %	0,212 m	
		80 %	8,5 %	0,214 m	8,5 %	0,212 m	8,5 %	0,212 m	8,5 %	0,212 m	
	2	60 %	8,5 %	0,213 m	8,5 %	0,212 m	8,5 %	0,212 m	8,5 %	0,211 m	
		70 %	8,5 %	0,213 m	8,5 %	0,212 m	8,5 %	0,212 m	8,5 %	0,212 m	
		80 %	8,5 %	0,214 m	8,5 %	0,212 m	8,5 %	0,212 m	8,5 %	0,212 m	
8	1	60 %	8,6 %	0,215 m	8,5 %	0,213 m	8,5 %	0,212 m	4,9 %	0,123 m	
		70 %	8,6 %	0,214 m	8,5 %	0,212 m	8,5 %	0,212 m	4,1 %	0,102 m	
		80 %	8,6 %	0,214 m	8,5 %	0,212 m	7,3 %	0,183 m	4,3 %	0,106 m	
	2	60 %	8,6 %	0,215 m	8,5 %	0,213 m	8,5 %	0,212 m	5,2 %	0,129 m	
		70 %	8,6 %	0,214 m	8,5 %	0,212 m	8,5 %	0,212 m	9,3 %	0,232 m	
		80 %	8,6 %	0,214 m	8,5 %	0,212 m	7,0 %	0,176 m	5,0 %	0,124 m	

A continuació es mostraran les gràfiques de porcions de les proves del vídeo 1.

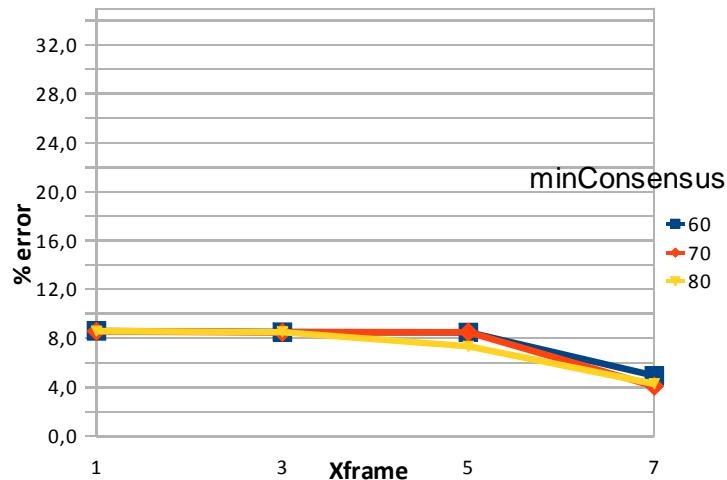
	minCons	Xframe	1	3	5	7
numMin=2	60 %		8,6 %	8,4 %	8,7 %	7,8 %
SELEC= 1	70 %		8,5 %	8,4 %	8,7 %	7,8 %
	80 %		8,5 %	8,4 %	8,7 %	7,8 %



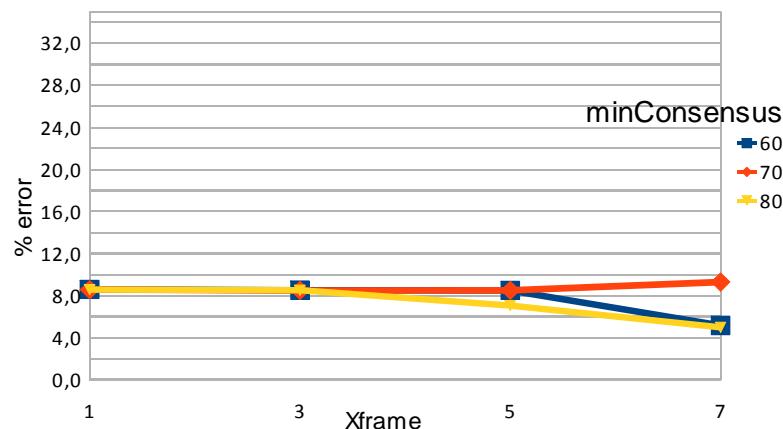
	minCons	Xframe	1	3	5	7
numMin=2	60 %		8,5 %	8,3 %	8,4 %	8,3 %
SELEC= 2	70 %		8,4 %	8,3 %	8,4 %	8,3 %
	80 %		8,4 %	8,3 %	8,4 %	8,3 %



	minCons	Xframe	1	3	5	7
numMin=8	60 %		8,6 %	8,5 %	8,5 %	4,9 %
SELEC= 1	70 %		8,6 %	8,5 %	8,5 %	4,1 %
	80 %		8,6 %	8,5 %	7,3 %	4,3 %



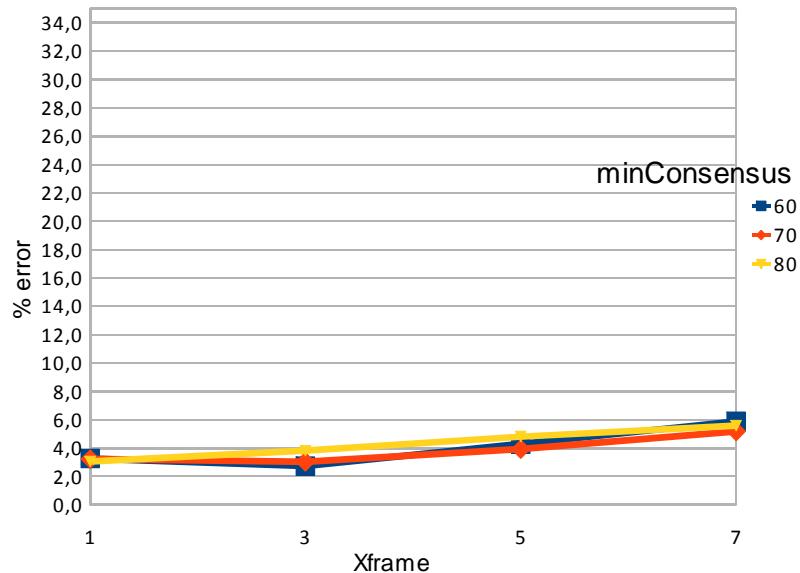
	minCons	Xframe	1	3	5	7
numMin=8	60 %		8,6 %	8,5 %	8,5 %	5,2 %
SELEC= 2	70 %		8,6 %	8,5 %	8,5 %	9,3 %
	80 %		8,6 %	8,5 %	7,0 %	5,0 %



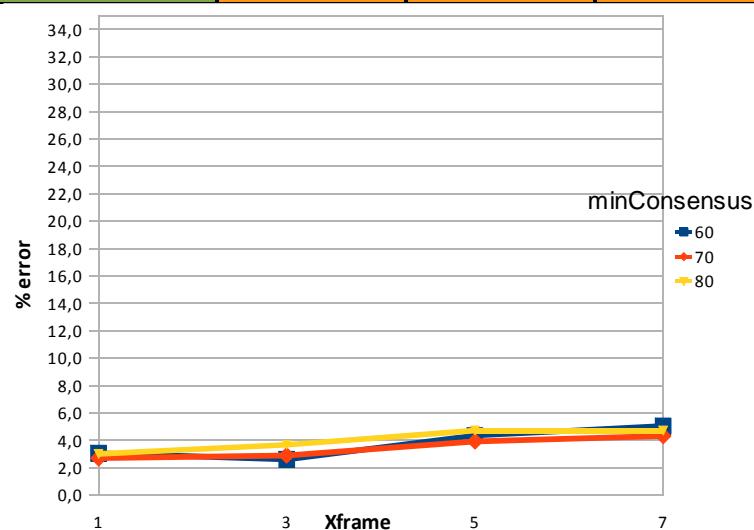
Aquesta taula ens mostra totes les proves realitzades en error de punt final d'odometria amb RANSAC del vídeo 2.

numMin	Xframe	Selec RANSAC	% Consensus	1		3		5		7	
				E. Relatiu	E. Absolut						
2	1	60 %	3,3 %	0,195 m	2,7 %	0,164 m	4,3 %	0,257 m	5,9 %	0,353 m	
			70 %	3,2 %	0,193 m	3,0 %	0,182 m	3,9 %	0,236 m	5,2 %	0,310 m
			80 %	3,1 %	0,184 m	3,8 %	0,228 m	4,8 %	0,288 m	5,6 %	0,334 m
	2	60 %	3,1 %	0,183 m	2,6 %	0,156 m	4,3 %	0,261 m	5,0 %	0,302 m	
			70 %	2,7 %	0,162 m	2,9 %	0,174 m	3,9 %	0,235 m	4,3 %	0,259 m
			80 %	3,0 %	0,180 m	3,7 %	0,220 m	4,7 %	0,282 m	4,7 %	0,283 m
4	1	60 %	3,7 %	0,222 m	3,6 %	0,218 m	2,9 %	0,175 m	4,7 %	0,283 m	
			70 %	4,1 %	0,244 m	4,0 %	0,238 m	3,0 %	0,180 m	4,6 %	0,278 m
			80 %	4,2 %	0,250 m	3,9 %	0,237 m	3,1 %	0,184 m	7,9 %	0,474 m
	2	60 %	3,7 %	0,222 m	5,2 %	0,314 m	10,8 %	0,645 m	2,4 %	0,146 m	
			70 %	4,1 %	0,244 m	5,7 %	0,341 m	14,0 %	0,839 m	4,1 %	0,245 m
			80 %	4,2 %	0,250 m	4,2 %	0,249 m	14,0 %	0,842 m	11,2 %	0,671 m
6	1	60 %	2,4 %	0,144 m	3,9 %	0,236 m	1,7 %	0,101 m	5,5 %	0,332 m	
			70 %	2,8 %	0,166 m	3,9 %	0,231 m	4,2 %	0,255 m	8,1 %	0,488 m
			80 %	2,7 %	0,162 m	3,9 %	0,232 m	3,8 %	0,225 m	17,6 %	1,058 m
	2	60 %	4,3 %	0,256 m	15,5 %	0,930 m	9,7 %	0,584 m	7,8 %	0,468 m	
			70 %	4,4 %	0,264 m	15,6 %	0,936 m	5,1 %	0,306 m	7,3 %	0,436 m
			80 %	6,8 %	0,407 m	16,1 %	0,965 m	4,3 %	0,259 m	8,8 %	0,527 m
8	1	60 %	3,6 %	0,217 m	4,1 %	0,248 m	4,7 %	0,284 m	13,3 %	0,797 m	
			70 %	5,9 %	0,356 m	4,7 %	0,279 m	3,4 %	0,205 m	14,1 %	0,848 m
			80 %	8,7 %	0,524 m	7,2 %	0,434 m	3,5 %	0,209 m	19,5 %	1,169 m
	2	60 %	7,3 %	0,436 m	18,8 %	1,130 m	11,0 %	0,662 m	14,7 %	0,882 m	
			70 %	7,0 %	0,420 m	22,8 %	1,370 m	12,3 %	0,738 m	14,1 %	0,844 m
			80 %	4,1 %	0,246 m	21,8 %	1,311 m	12,1 %	0,723 m	14,3 %	0,857 m

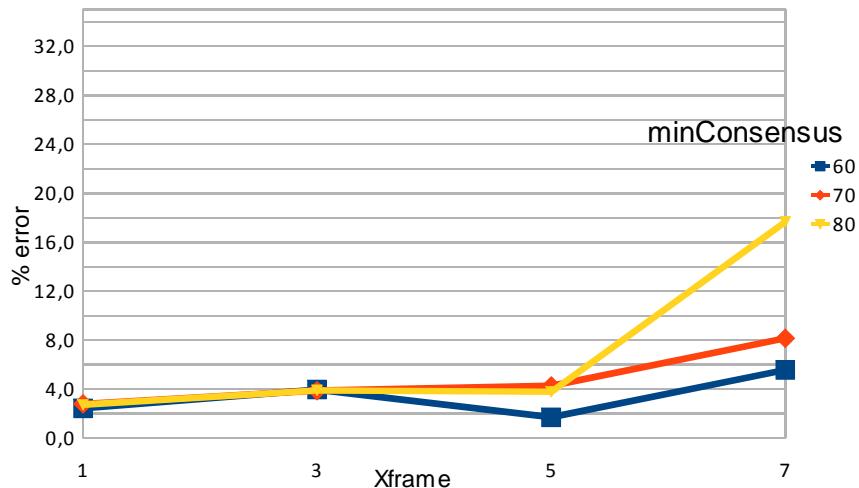
	minCons	Xframe	1	3	5	7
	60 %		3,3 %	2,7 %	4,3 %	5,9 %
	70 %		3,2 %	3,0 %	3,9 %	5,2 %
	80 %		3,1 %	3,8 %	4,8 %	5,6 %



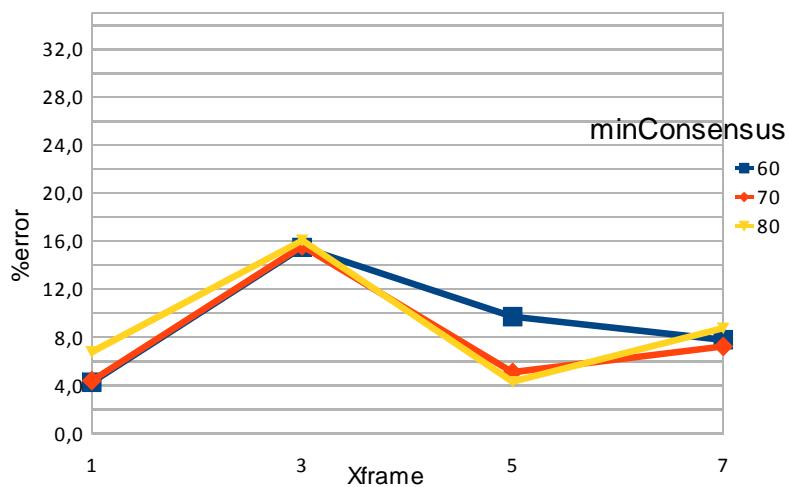
	minCons	Xframe	1	3	5	7
	60 %		3,1 %	2,6 %	4,3 %	5,0 %
	70 %		2,7 %	2,9 %	3,9 %	4,3 %
	80 %		3,0 %	3,7 %	4,7 %	4,7 %



	minCons	Xframe	1	3	5	7
numMin=6	60 %		2,4 %	3,9 %	1,7 %	5,5 %
SELEC= 1	70 %		2,8 %	3,9 %	4,2 %	8,1 %
	80 %		2,7 %	3,9 %	3,8 %	17,6 %



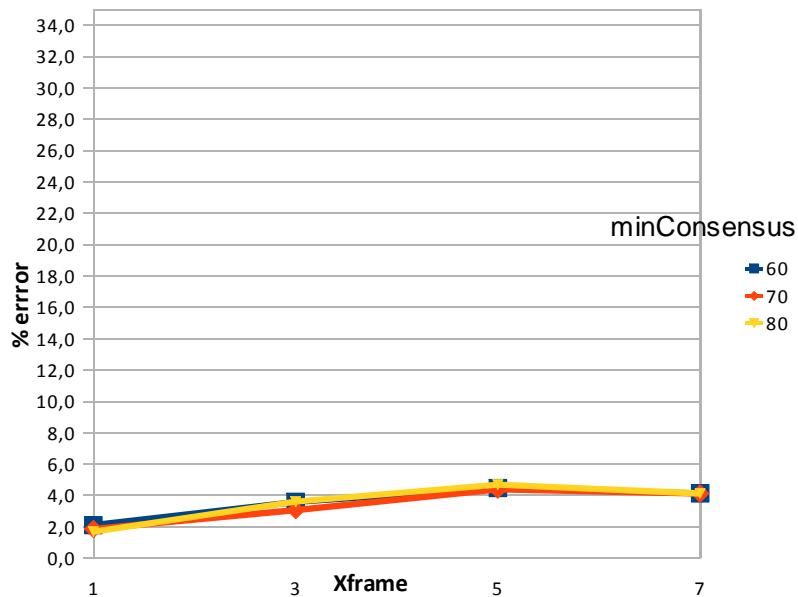
	minCons	Xframe	1	3	5	7
numMin=6	60 %		4,3 %	15,5 %	9,7 %	7,8 %
SELEC= 2	70 %		4,4 %	15,6 %	5,1 %	7,3 %
	80 %		6,8 %	16,1 %	4,3 %	8,8 %



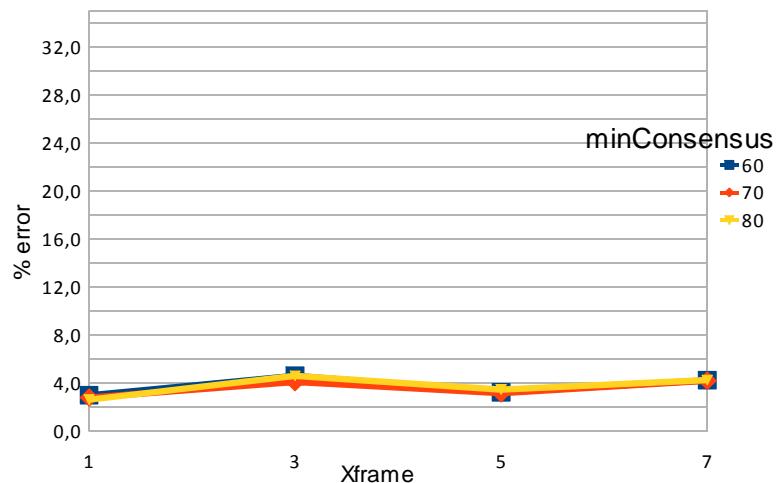
Aquesta taula ens mostra totes les proves realitzades en error de punt final d'odometria amb RANSAC del vídeo 3.

numMin	Xframe	Selec RANSAC	% Consensus	1		3		5		7	
				E. Relatiu	E. Absolut						
2	1		60 %	2,1 %	0,166 m	3,6 %	0,282 m	4,5 %	0,351 m	4,1 %	0,326 m
			70 %	1,9 %	0,146 m	3,1 %	0,241 m	4,4 %	0,344 m	4,1 %	0,324 m
			80 %	1,7 %	0,133 m	3,6 %	0,283 m	4,7 %	0,368 m	4,1 %	0,324 m
	2		60 %	3,0 %	0,233 m	4,6 %	0,360 m	3,2 %	0,252 m	4,2 %	0,331 m
			70 %	2,8 %	0,218 m	4,1 %	0,319 m	3,1 %	0,246 m	4,2 %	0,328 m
			80 %	2,6 %	0,203 m	4,6 %	0,360 m	3,4 %	0,270 m	4,2 %	0,332 m
4	1		60 %	5,0 %	0,391 m	3,9 %	0,303 m	4,1 %	0,325 m	3,6 %	0,283 m
			70 %	4,8 %	0,379 m	4,0 %	0,317 m	4,1 %	0,326 m	3,4 %	0,270 m
			80 %	4,1 %	0,320 m	4,1 %	0,321 m	4,3 %	0,336 m	3,8 %	0,295 m
	2		60 %	1,3 %	0,104 m	4,0 %	0,316 m	4,1 %	0,325 m	15,0 %	1,182 m
			70 %	1,5 %	0,115 m	4,2 %	0,329 m	4,1 %	0,326 m	15,1 %	1,187 m
			80 %	1,6 %	0,128 m	4,2 %	0,333 m	4,3 %	0,336 m	1,7 %	0,137 m
6	1		60 %	4,2 %	0,333 m	4,1 %	0,324 m	0,9 %	0,073 m	2,1 %	0,164 m
			70 %	4,6 %	0,360 m	4,3 %	0,342 m	1,5 %	0,121 m	3,6 %	0,282 m
			80 %	3,2 %	0,251 m	4,2 %	0,330 m	1,1 %	0,089 m	3,6 %	0,282 m
	2		60 %	5,4 %	0,425 m	1,1 %	0,084 m	4,4 %	0,346 m	11,2 %	0,881 m
			70 %	5,7 %	0,445 m	2,3 %	0,183 m	1,5 %	0,120 m	11,6 %	0,908 m
			80 %	5,2 %	0,410 m	2,2 %	0,172 m	2,3 %	0,177 m	12,7 %	1,000 m
8	1		60 %	2,9 %	0,230 m	2,4 %	0,189 m	3,7 %	0,293 m	5,4 %	0,425 m
			70 %	1,2 %	0,093 m	2,2 %	0,171 m	2,5 %	0,194 m	3,3 %	0,257 m
			80 %	1,0 %	0,078 m	2,4 %	0,185 m	11,1 %	0,875 m	17,7 %	1,393 m
	2		60 %	8,1 %	0,634 m	6,9 %	0,546 m	7,3 %	0,576 m	8,5 %	0,670 m
			70 %	13,7 %	1,073 m	9,4 %	0,738 m	6,2 %	0,490 m	7,8 %	0,615 m
			80 %	12,2 %	0,961 m	11,2 %	0,876 m	4,3 %	0,336 m	9,0 %	0,710 m

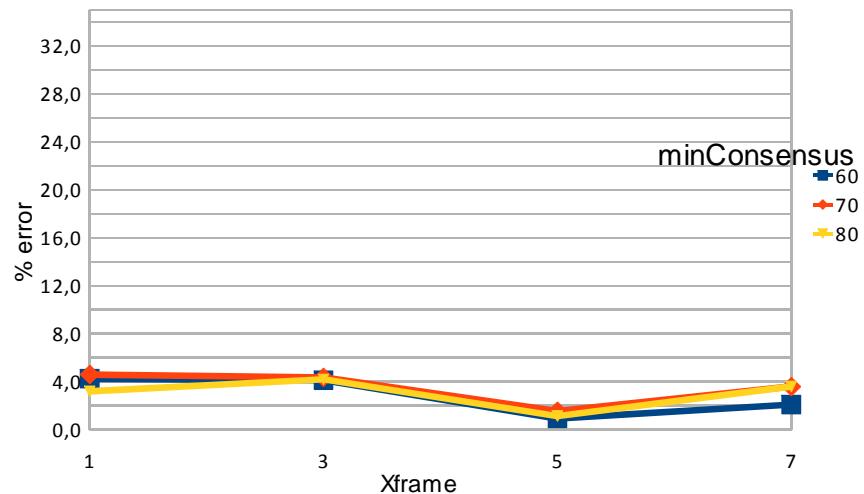
	minCons	Xframe	1	3	5	7
numMin=2	60 %		2,1 %	3,6 %	4,5 %	4,1 %
SELEC= 1	70 %		1,9 %	3,1 %	4,4 %	4,1 %
	80 %		1,7 %	3,6 %	4,7 %	4,1 %



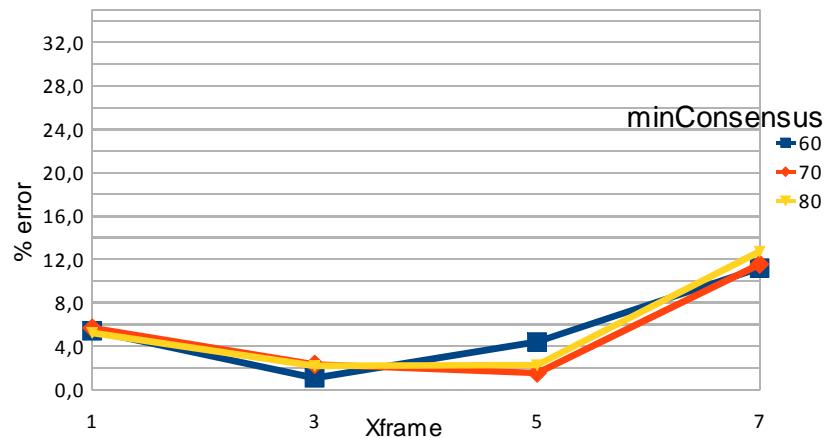
	minCons	Xframe	1	3	5	7
numMin=2	60 %		3,0 %	4,6 %	3,2 %	4,2 %
SELEC= 2	70 %		2,8 %	4,1 %	3,1 %	4,2 %
	80 %		2,6 %	4,6 %	3,4 %	4,2 %



	minCons	Xframe	1	3	5	7
numMin=6	60 %		4,2 %	4,1 %	0,9 %	2,1 %
SELEC= 1	70 %		4,6 %	4,3 %	1,5 %	3,6 %
	80 %		3,2 %	4,2 %	1,1 %	3,6 %



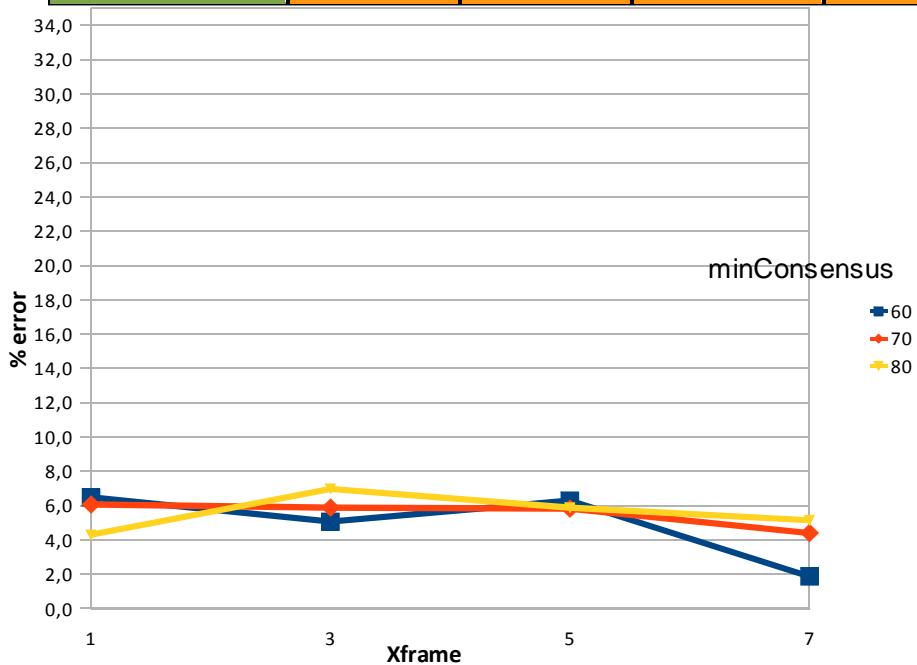
	minCons	Xframe	1	3	5	7
numMin=6	60 %		5,4 %	1,1 %	4,4 %	11,2 %
SELEC= 2	70 %		5,7 %	2,3 %	1,5 %	11,6 %
	80 %		5,2 %	2,2 %	2,3 %	12,7 %



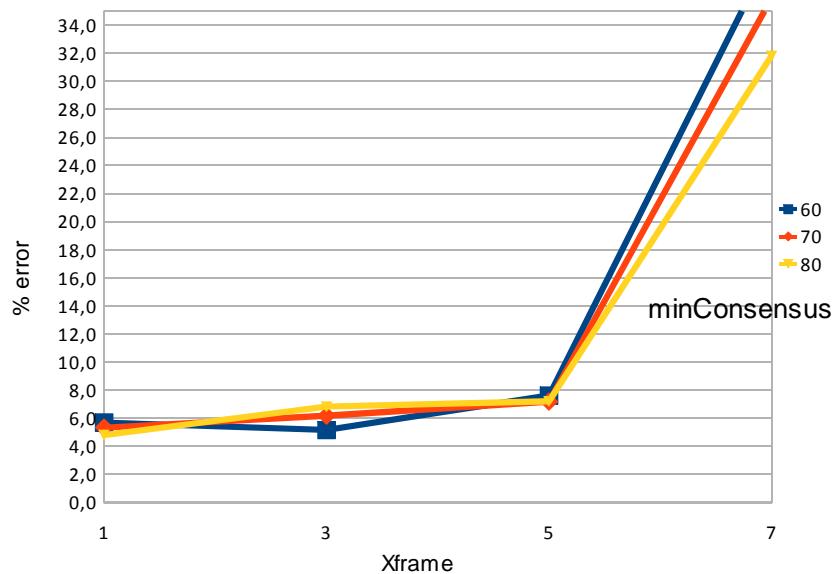
Aquesta taula ens mostra totes les proves realitzades en error de punt final d'odometria amb RANSAC del vídeo 4.

numMin	Xframe	Selec RANSAC	% Consensus	1		3		5		7	
				E. Relatiu	E. Absolut						
2	1	1	60 %	6,5 %	0,697 m	5,1 %	0,545 m	6,3 %	0,677 m	1,9 %	0,201 m
			70 %	6,1 %	0,653 m	5,9 %	0,632 m	5,8 %	0,627 m	4,4 %	0,474 m
			80 %	4,3 %	0,463 m	7,0 %	0,751 m	5,9 %	0,633 m	5,1 %	0,554 m
	2	2	60 %	5,7 %	0,612 m	5,2 %	0,555 m	7,6 %	0,819 m	39,2 %	4,219 m
			70 %	5,3 %	0,573 m	6,2 %	0,665 m	7,2 %	0,772 m	35,9 %	3,862 m
			80 %	4,8 %	0,516 m	6,8 %	0,734 m	7,2 %	0,776 m	31,8 %	3,422 m
4	1	1	60 %	5,6 %	0,605 m	7,4 %	0,801 m	5,8 %	0,621 m	21,5 %	2,319 m
			70 %	5,8 %	0,623 m	7,4 %	0,794 m	4,9 %	0,527 m	18,6 %	1,996 m
			80 %	6,7 %	0,722 m	7,6 %	0,817 m	5,3 %	0,570 m	22,9 %	2,463 m
	2	2	60 %	5,6 %	0,605 m	6,0 %	0,641 m	32,8 %	3,528 m	26,3 %	2,834 m
			70 %	6,3 %	0,677 m	5,4 %	0,584 m	34,5 %	3,707 m	24,9 %	2,680 m
			80 %	6,8 %	0,733 m	5,9 %	0,637 m	32,6 %	3,510 m	24,1 %	2,589 m
6	1	1	60 %	2,9 %	0,311 m	10,0 %	1,080 m	8,1 %	0,866 m	20,9 %	2,254 m
			70 %	4,3 %	0,464 m	10,5 %	1,132 m	10,2 %	1,095 m	21,4 %	2,302 m
			80 %	5,3 %	0,576 m	11,9 %	1,275 m	13,8 %	1,482 m	21,5 %	2,309 m
	2	2	60 %	5,4 %	0,584 m	7,9 %	0,850 m	34,5 %	3,707 m	13,2 %	1,421 m
			70 %	6,1 %	0,654 m	8,7 %	0,940 m	34,3 %	3,693 m	13,1 %	1,412 m
			80 %	6,1 %	0,661 m	8,9 %	0,961 m	34,1 %	3,664 m	12,9 %	1,391 m
8	1	1	60 %	18,8 %	2,026 m	16,1 %	1,732 m	17,1 %	1,838 m	27,3 %	2,933 m
			70 %	22,4 %	2,406 m	18,2 %	1,962 m	14,7 %	1,586 m	27,2 %	2,931 m
			80 %	25,9 %	2,784 m	21,6 %	2,325 m	16,6 %	1,788 m	24,6 %	2,652 m
	2	2	60 %	10,0 %	1,073 m	18,9 %	2,035 m	15,7 %	1,687 m	7,4 %	0,801 m
			70 %	10,1 %	1,087 m	16,8 %	1,811 m	14,7 %	1,583 m	7,6 %	0,815 m
			80 %	11,4 %	1,229 m	16,0 %	1,725 m	14,0 %	1,511 m	7,3 %	0,783 m

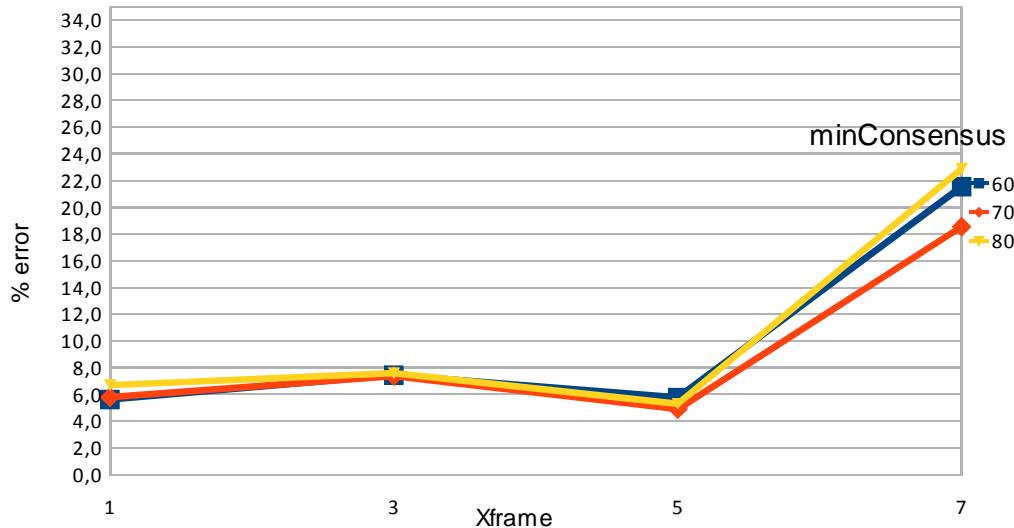
	minCons	Xframe	1	3	5	7
numMin=2	60 %		6,5 %	5,1 %	6,3 %	1,9 %
SELEC= 1	70 %		6,1 %	5,9 %	5,8 %	4,4 %
	80 %		4,3 %	7,0 %	5,9 %	5,1 %



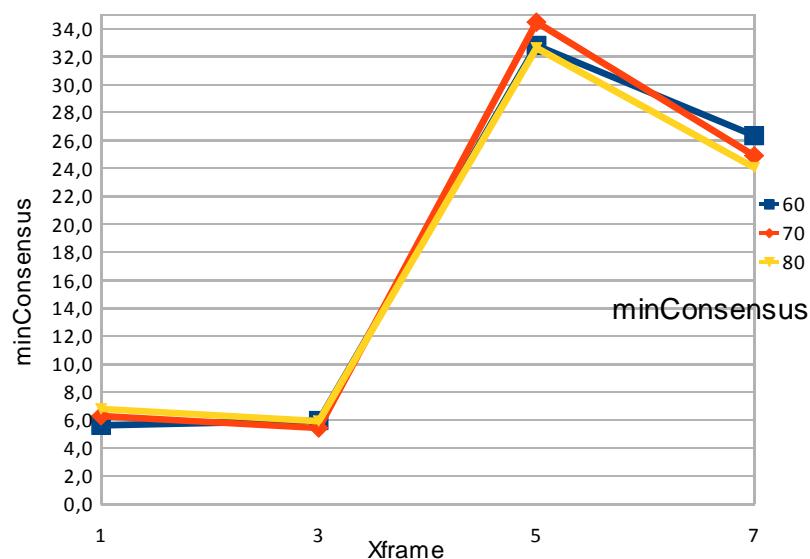
	minCons	Xframe	1	3	5	7
numMin=2	60 %		5,7 %	5,2 %	7,6 %	39,2 %
SELEC= 2	70 %		5,3 %	6,2 %	7,2 %	35,9 %
	80 %		4,8 %	6,8 %	7,2 %	31,8 %



	minCons	Xframe	1	3	5	7
	numMin=4	60 %	5,6 %	7,4 %	5,8 %	21,5 %
	SELEC= 1	70 %	5,8 %	7,4 %	4,9 %	18,6 %
	80 %	6,7 %	7,6 %	5,3 %	22,9 %	



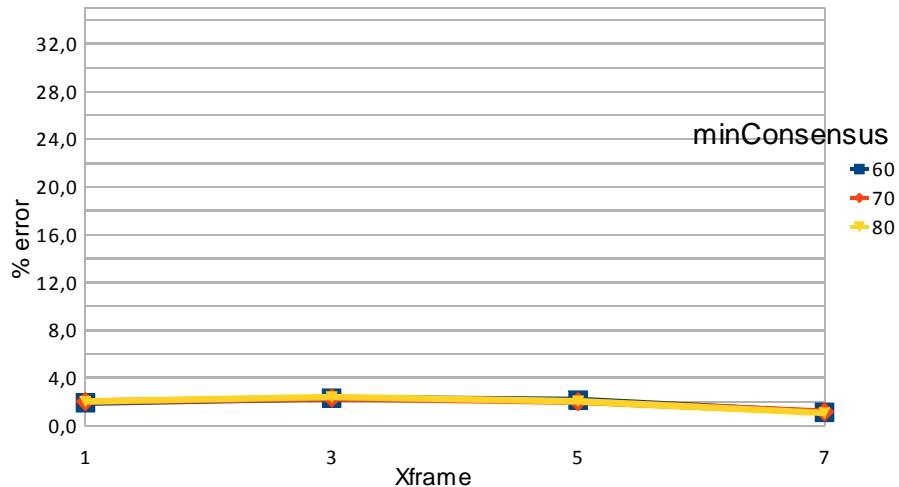
	minCons	Xframe	1	3	5	7
	numMin=4	60 %	5,6 %	6,0 %	32,8 %	26,3 %
	SELEC= 2	70 %	6,3 %	5,4 %	34,5 %	24,9 %
	80 %	6,8 %	5,9 %	32,6 %	24,1 %	



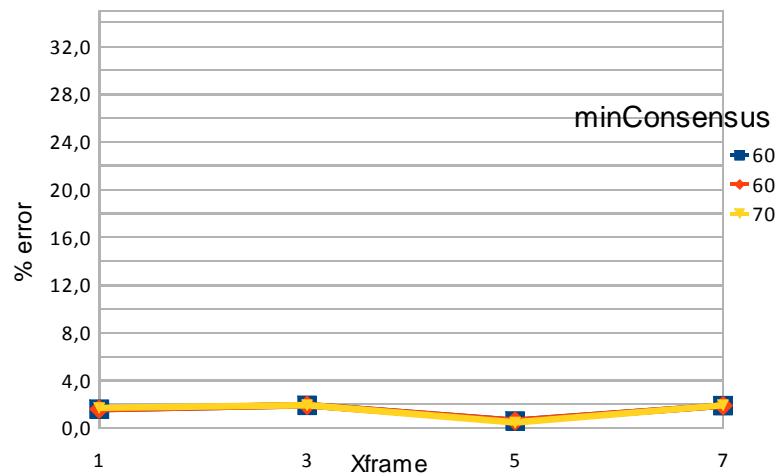
Aquesta taula ens mostra totes les proves realitzades en error de punt final d'odometria amb RANSAC del vídeo 5.

numMin	Xframe	Selec RANSAC	% Consensus	1		3		5		7	
				E. Relatiu	E. Absolut						
2	1	1	60 %	1,9 %	0,188 m	2,3 %	0,226 m	2,2 %	0,211 m	1,1 %	0,109 m
			70 %	2,0 %	0,198 m	2,3 %	0,226 m	2,0 %	0,197 m	1,2 %	0,113 m
			80 %	2,0 %	0,199 m	2,4 %	0,235 m	2,0 %	0,200 m	1,1 %	0,104 m
	2	2	60 %	1,6 %	0,157 m	1,9 %	0,189 m	0,6 %	0,060 m	1,9 %	0,185 m
			70 %	1,7 %	0,167 m	1,9 %	0,190 m	0,5 %	0,048 m	1,9 %	0,189 m
			80 %	1,7 %	0,168 m	2,0 %	0,199 m	0,5 %	0,050 m	1,8 %	0,180 m
4	1	1	60 %	1,6 %	0,158 m	1,9 %	0,182 m	0,6 %	0,063 m	1,0 %	0,093 m
			70 %	1,6 %	0,158 m	1,9 %	0,184 m	0,6 %	0,063 m	1,1 %	0,107 m
			80 %	1,6 %	0,157 m	1,9 %	0,186 m	0,3 %	0,032 m	1,3 %	0,127 m
	2	2	60 %	1,6 %	0,158 m	2,7 %	0,261 m	1,8 %	0,176 m	29,6 %	2,901 m
			70 %	1,6 %	0,158 m	2,7 %	0,262 m	1,8 %	0,175 m	28,5 %	2,797 m
			80 %	1,6 %	0,157 m	2,7 %	0,264 m	1,7 %	0,169 m	27,2 %	2,665 m
6	1	1	60 %	1,0 %	0,094 m	1,6 %	0,157 m	4,4 %	0,429 m	7,4 %	0,721 m
			70 %	0,9 %	0,092 m	1,4 %	0,134 m	5,4 %	0,529 m	8,0 %	0,780 m
			80 %	0,2 %	0,019 m	1,7 %	0,162 m	5,8 %	0,567 m	9,1 %	0,892 m
	2	2	60 %	2,0 %	0,191 m	0,6 %	0,063 m	3,0 %	0,295 m	8,2 %	0,800 m
			70 %	1,9 %	0,189 m	2,4 %	0,236 m	4,0 %	0,387 m	7,9 %	0,779 m
			80 %	2,3 %	0,226 m	2,3 %	0,228 m	4,0 %	0,393 m	8,3 %	0,811 m
8	1	1	60 %	0,4 %	0,039 m	1,2 %	0,116 m	10,1 %	0,988 m	10,5 %	1,028 m
			70 %	3,9 %	0,378 m	1,2 %	0,117 m	10,7 %	1,048 m	10,7 %	1,049 m
			80 %	6,0 %	0,591 m	0,4 %	0,038 m	10,2 %	0,999 m	9,6 %	0,946 m
	2	2	60 %	2,1 %	0,210 m	6,8 %	0,667 m	7,4 %	0,722 m	8,0 %	0,784 m
			70 %	2,6 %	0,251 m	7,2 %	0,701 m	7,7 %	0,751 m	6,6 %	0,650 m
			80 %	2,5 %	0,244 m	5,9 %	0,575 m	7,6 %	0,742 m	6,0 %	0,590 m

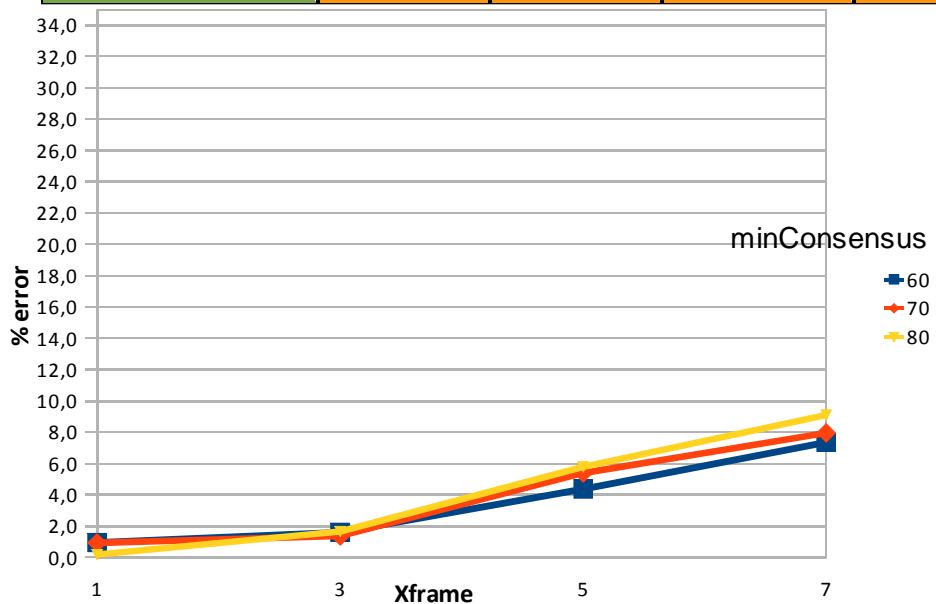
	minCons	Xframe	1	3	5	7
numMin=2	60 %		1,9 %	2,3 %	2,2 %	1,1 %
SELEC= 1	70 %		2,0 %	2,3 %	2,0 %	1,2 %
	80 %		2,0 %	2,4 %	2,0 %	1,1 %



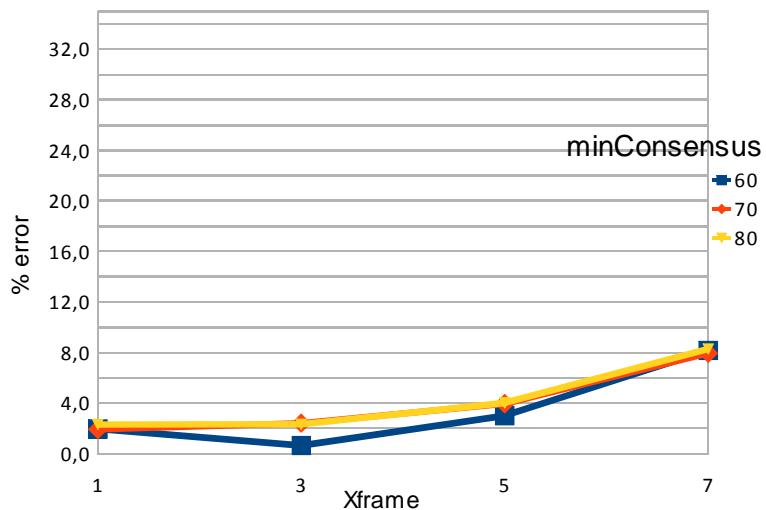
	minCons	Xframe	1	3	5	7
numMin=2	60 %		1,6 %	1,9 %	0,6 %	1,9 %
SELEC= 1	70 %		1,7 %	1,9 %	0,5 %	1,9 %
	80 %		1,7 %	2,0 %	0,5 %	1,8 %



	minCons	Xframe	1	3	5	7
numMin=6	60 %		1,0 %	1,6 %	4,4 %	7,4 %
SELEC= 2	70 %		0,9 %	1,4 %	5,4 %	8,0 %
	80 %		0,2 %	1,7 %	5,8 %	9,1 %



	minCons	Xframe	1	3	5	7
numMin=6	60 %		2,0 %	0,6 %	3,0 %	8,2 %
SELEC= 2	70 %		1,9 %	2,4 %	4,0 %	7,9 %
	80 %		2,3 %	2,3 %	4,0 %	8,3 %



A continuació es mostren totes les proves realitzades amb RANSAC per acumulació d'error:

- SELEC = 1:

Vídeo 1	1	3	5	7	Xframe	NumMin
Dist real: 2,5 m	2,78 m	2,77 m	2,75 m	2,74 m		2
	2,79 m	2,77 m	2,75 m	2,75 m		4
	2,80 m	2,77 m	2,76 m	2,76 m		6
	2,80 m	2,77 m	2,76 m	2,64 m		8
	2,8 m	2,8 m	2,8 m	2,7 m		Mitja
	0,0 m	0,0 m	0,0 m	0,1 m		Desviació tipica
	11,7 %	10,8 %	10,1 %	8,9 %		Error %

Video 2	1	3	5	7	Xframe	NumMin
Dist real: 6 m	7,069 m	7,091 m	7,137 m	6,962 m		2
	7,177 m	7,094 m	6,927 m	6,890 m		4
	7,111 m	7,078 m	6,553 m	6,854 m		6
	6,435 m	7,047 m	6,152 m	5,839 m		8
	6,9 m	7,1 m	6,7 m	6,6 m		Mitja
	0,6 m	0,0 m	0,8 m	0,9 m		Desviació tipica
	15,8 %	18,0 %	11,5 %	10,6 %		Error %

Vídeo 3	1	3	5	7	Xframe	NumMin
Dist real: 7,854 m	8,62 m	8,45 m	8,46 m	8,38 m		2
	8,62 m	8,47 m	8,44 m	8,45 m		4
	8,60 m	8,49 m	8,23 m	8,02 m		6
	8,48 m	8,45 m	7,90 m	7,37 m		8
	8,6 m	8,5 m	8,3 m	8,1 m		Mitja
	0,1 m	0,0 m	0,5 m	0,9 m		Desviació tipica
	9,2 %	7,8 %	5,1 %	2,6 %		Error %

Vídeo 5	1	3	5	7	Xframe	NumMin
Dist real: 9,801 m	10,38 m	10,39 m	10,36 m	10,25 m	2	
	10,37 m	10,37 m	10,23 m	10,05 m	4	
	10,30 m	10,35 m	9,63 m	9,17 m	6	
	9,83 m	10,35 m	8,85 m	8,50 m	8	
	10,2 m	10,4 m	9,8 m	9,5 m	Mitja	
	0,5 m	0,0 m	1,2 m	1,4 m	Desviació tipica	
	4,3 %	5,7 %	0,3 %	3,2 %	Error %	

- SELEC = 2:

Vídeo1	1	3	5	7	Xframe	NumMin
Dist real: 2,5 m	2,79 m	2,77 m	2,75 m	2,75 m	2	
	2,79 m	2,77 m	2,75 m	2,75 m	4	
	2,80 m	2,77 m	2,76 m	2,76 m	6	
	2,80 m	2,77 m	2,76 m	2,52 m	8	
	2,796 m	2,770 m	2,754 m	2,695 m	Mitja	
	0,004 m	0,004 m	0,004 m	0,206 m	Desviació tipica	
	11,8 %	10,8 %	10,2 %	7,8 %	Error %	

Vídeo 2	1	3	5	7	Xframe	NumMin
Dist real: 6 m	7,18 m	7,09 m	7,03 m	7,02 m	2	
	7,18 m	7,06 m	6,83 m	6,59 m	4	
	7,10 m	6,80 m	5,99 m	5,12 m	6	
	6,72 m	6,03 m	4,64 m	3,32 m	8	
	7,0 m	6,7 m	6,1 m	5,5 m	Mitja	
	0,4 m	0,9 m	1,9 m	2,9 m	Desviació tipica	
	17,4 %	12,4 %	2,0 %	8,1 %	Error %	

Vídeo 3	1	3	5	7	Xframe	NumMin
Dist real: 7,854 m	8,61 m	8,47 m	8,43 m	8,37 m	2	
	8,59 m	8,41 m	8,44 m	8,16 m	4	
	8,60 m	8,34 m	7,80 m	6,55 m	6	
	8,41 m	7,56 m	5,85 m	4,13 m	8	
	8,6 m	8,2 m	7,6 m	6,8 m	Mitja	
	0,2 m	0,7 m	2,1 m	3,4 m	Desviació tipica	
	8,9 %	4,3 %	2,8 %	13,4 %	Error %	

Vídeo 4	1	3	5	7	Xframe	NumMin
Dist: real: 32,28 m	34,28 m	34,19 m	34,14 m	31,28 m	2	
	34,38 m	33,87 m	28,39 m	17,82 m	4	
	33,73 m	28,68 m	16,61 m	10,51 m	6	
	29,30 m	19,14 m	11,62 m	8,03 m	8	
	32,9 m	29,0 m	22,7 m	16,9 m	Mitja	
	4,2 m	12,2 m	18,0 m	18,1 m	Desviació tipica	
	2,0 %	10,3 %	29,7 %	47,6 %	Error %	

Vídeo 5	1	3	5	7	Xframe	NumMin
Dist Real: 9,801 m	10,38 m	10,39 m	10,36 m	10,25 m	2	
	10,37 m	10,37 m	10,23 m	10,05 m	4	
	10,30 m	10,35 m	9,63 m	9,17 m	6	
	9,83 m	10,35 m	8,85 m	8,50 m	8	
	10,2 m	10,4 m	9,8 m	9,5 m	Mitja	
	0,5 m	0,0 m	1,2 m	1,4 m	Desviació tipica	
	4,3 %	5,7 %	0,3 %	3,2 %	Error %	

A continuació es mostraran totes les taules de temps amb RANSAC:

- SELEC = 1:

Vídeo 1	1	3	5	7	Xframe	NumMin
Temps real: 18 s	133 s	44 s	26 s	19 s	2	
	141 s	49 s	30 s	23 s	4	
	153 s	49 s	33 s	22 s	6	
	156 s	50 s	36 s	23 s	8	
	146 s	48 s	31 s	22 s	Mitja	

Vídeo 2	1	3	5	7	Xframe	NumMin
Temps real: 35 s	255 s	77 s	45 s	34 s	2	
	267 s	87 s	54 s	37 s	4	
	297 s	89 s	55 s	38 s	6	
	285 s	93 s	57 s	39 s	8	
	276 s	87 s	53 s	37 s	Mitja	

Vídeo 3	1	3	5	7	Xframe	NumMin
Temps real: 43 s	293 s	100 s	63 s	43 s	2	
	310 s	121 s	60 s	43 s	4	
	323 s	169 s	66 s	48 s	6	
	372 s	112 s	82 s	49 s	8	
	325 s	125 s	68 s	45 s	Mitja	

Vídeo 4	1	3	5	7	Xframe	NumMin
Temps real: 100 s	586 s	179 s	98 s	74 s	2	
	629 s	198 s	115 s	81 s	4	
	708 s	212 s	128 s	87 s	6	
	649 s	203 s	136 s	98 s	8	
	643 s	198 s	120 s	85 s	Mitja	

Vídeo 5	1	3	5	7	Xframe	NumMin
Temps real: 32 s	196 s	60 s	49 s	27 s	2	
	239 s	76 s	51 s	31 s	4	
	236 s	79 s	45 s	32 s	6	
	267 s	85 s	48 s	33 s	8	
	234 s	75 s	48 s	31 s	Mitja	

- SELEC = 2:

Vídeo 1	1	3	5	7	Xframe	NumMin
Temps real: 18 s	133 s	42 s	26 s	19 s	2	
	140 s	49 s	30 s	23 s	4	
	155 s	49 s	30 s	21 s	6	
	153 s	50 s	35 s	24 s	8	
	145 s	48 s	30 s	22 s	Mitja	

Vídeo 2	1	3	5	7	Xframe	NumMin
Temps real: 35 s	249 s	76 s	44 s	33 s	2	
	264 s	87 s	53 s	37 s	4	
	285 s	89 s	55 s	40 s	6	
	287 s	93 s	57 s	39 s	8	
	271 s	86 s	52 s	37 s	Mitja	

Vídeo 3	1	3	5	7	Xframe	NumMin
Temps real: 43 s	317 s	100 s	62 s	43 s	2	
	307 s	132 s	60 s	43 s	4	
	323 s	140 s	69 s	47 s	6	
	373 s	112 s	69 s	51 s	8	
	330 s	121 s	65 s	46 s	Mitja	

Vídeo 4	1	3	5	7	Xframe	NumMin
Temps real: 100 s	567 s	178 s	110 s	77 s	2	
	644 s	195 s	126 s	81 s	4	
	716 s	190 s	126 s	91 s	6	
	642 s	204 s	137 s	95 s	8	
	642 s	192 s	125 s	86 s	Mitja	

Vídeo 5	1	3	5	7	Xframe	NumMin
Temps real: 32 s	196 s	60 s	49 s	27 s	2	
	239 s	76 s	51 s	31 s	4	
	236 s	79 s	45 s	32 s	6	
	267 s	85 s	48 s	33 s	8	
	234 s	75 s	48 s	31 s	Mitja	

Bibliografia

- [1] F. Devernay i O. Faugeras, *Straight lines have to be straight: automatic calibration and removal of distortion from scenes of structured environments*. Machine Vision and Applications Journal. Springer-Verlag New York. Secaucus, 2001. pp. 14-24.
- [2] SONY, *DSC-P10* [En línia]. <<http://www.sony.es/support/es/product/dsc-p10>>. [Consulta 04/05/2015].
- [3] Caltech, *Camera Calibration Toolbox for Matlab* [En línia]. <http://www.vision.caltech.edu/bouguetj/calib_doc/>. [Consulta: 13-7-2014].
- [4] M.Muja i D. Lowe, *Distinctive image features from scale-invariant keypoints*. *International Journal of Computer Vision*. 60, 2 ,2004. pp. 91-110.
- [5] M.Muja i D. Lowe, *Fast approximate nearest neighbors with automatic algorithm configuration*. International Conference on Computer Vision Theory and Applications (VISAPP). Lisboa, Portugal 2009.
- [6] A. Moore, D. Hill i M. Johnson, An Empirical Investigation of Brute Force to choose Features, Smoothers and Function Approximators. En: R. Greiner, T. Petsche i S. Hanson. *Computational Learning Theory and Natural Learning Systems, Volume 4*. MIT Press. 1995. pp. 361-379.
- [7] F. Lu i Milios, *Robot pose estimation in unknown environements by matching 2D range scans*. Computer Society conference on (IEEE). Seattle, 1994. pp. 935-938.
- [8] Wilcox. R. R, *Robust Regression Methods:Achieving small standard errors when there is heteroscedasticity*. Understanding Statistics. Erlbaum. 2004. pp. 349-364.
- [9] A. Burguera, F. Bonin-Font i G. Oliver, *Towards Robust Image Registration for Underwater Visual SLAM*. International Conference on Computer Vision Theory and Applications (VISSAP). Lisboa, Portugal, 2014. pp. 539-544.
- [10] A. Burguera i Y. González, *RANSAC based data association for underwater visual SLAM*. First Iberian Robotics Conference. Madrid, España, 2013.